МАТЕРИАЛЫ ЗАДАНИЙ
Всероссийской междисциплинарной олимпиады школьников 8–11 класса «Национальная технологическая олимпиада»

по профилю «Интеллектуальные энергетические системы»

2024/25 учебный год

ntcontest.ru

УДК 373.5.016:[620.9:004.8]
ББК 74.263.2
И73

Авторы:
С. Е. Бойченко, И. А. Воронцов, А. А. Гаврилюк, Е. Н. Горечин, И. С. Давыдов, О. В. Зубков, О. Д. Иванова, И. Ф. Лацимирский, И. Б. Мамай, И. Г. Просекина, М. Ю. Просекин, А. В. Резников, В. В. Широков, М. А. Чекан, Д. М. Цивилева

И73 Всероссийская междисциплинарная олимпиада школьников 8–11 класса «Национальная технологическая олимпиада». Учебно-методическое пособие
Том 11 **Интеллектуальные энергетические системы**
— М.: Ассоциация участников технологических кружков, 2025. — 199 с.

ISBN 978-5-908021-10-4

Данное пособие разработано коллективом авторов на основе опыта проведения всероссийской междисциплинарной олимпиады школьников 8–11 класса «Национальная технологическая олимпиада» в 2024/25 учебном году, а также многолетнего опыта проведения инженерных соревнований для школьников. В пособии собраны основные материалы, необходимые как для подготовки к олимпиаде, так и для углубления знаний и приобретения навыков решения инженерных задач.

В издании приведены варианты заданий по профилю Национальной технологической олимпиады за 2024/25 учебный год с ответами, подробными решениями и комментариями. Пособие адресовано учащимся 8–11 классов, абитуриентам, школьным учителям, наставникам и преподавателям учреждений дополнительного образования, центров молодежного и инновационного творчества и детских технопарков.

Методические материалы также могут быть полезны студентам и преподавателям направлений, относящихся к группам:
01.00.00 Математика и механика
02.00.00 Компьютерные и информационные науки
09.00.00 Информатика и вычислительная техника
10.00.00 Информационная безопасность
11.00.00 Электроника, радиотехника и системы связи
13.00.00 Электро- и теплоэнергетика
27.00.00 Управление в технических системах
38.00.00 Экономика и управление

ISBN 978-5-908021-10-4

9 785908 021104 >

УДК 373.5.016:[620.9:004.8]
ББК 74.263.2

Оглавление

1. Введение

1.1. Национальная технологическая олимпиада

Всероссийская междисциплинарная олимпиада школьников 8–11 класса «Национальная технологическая олимпиада» (далее — Олимпиада, НТО) проводится в соответствии с распоряжением Правительства Российской Федерации от 10.02.2022 № 211-р при координации Министерства науки и высшего образования Российской Федерации и при содействии Министерства просвещения Российской Федерации, Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации, Министерства промышленности и торговли Российской Федерации, Ассоциации участников технологических кружков, Агентства стратегических инициатив по продвижению новых проектов, АНО «Россия — страна возможностей», АНО «Платформа Национальной технологической инициативы» и Российского движения детей и молодежи «Движение Первых».

Проектное управление Олимпиадой осуществляет структурное подразделение Национального исследовательского университета «Высшая школа экономики» — Центр Национальной технологической олимпиады. Организационный комитет по подготовке и проведению Национальной технологической олимпиады возглавляют первый заместитель Руководителя Администрации Президента Российской Федерации С. В. Кириенко и заместитель Председателя Правительства Российской Федерации Д. Н. Чернышенко.

Национальная технологическая олимпиада — это командная инженерная Олимпиада, позволяющая школьникам работать в самых передовых инженерных направлениях. Она базируется на опыте Олимпиады Кружкового движения НТИ и проводится с 2015 года, а с 2016 года входит в перечень Российского совета олимпиад школьников и дает победителям и призерам льготы при поступлении в университеты.

Всего заявки на участие в десятом юбилейном сезоне (2024–25 гг.) самых масштабных в России командных инженерных соревнованиях подали более 140 тысяч школьников. Общий охват олимпиады с 2015 года превысил 880 тысяч участников.

НТО способствует формированию профессиональной траектории школьников, увлеченных научно-техническим творчеством и помогает им:

- определить свой интерес в мире современных технологий;
- получить опыт решения комплексных инженерных задач;
- осознанно выбрать вуз для продолжения обучения и поступить в него на льготных условиях.

Кроме того, НТО позволяет каждому участнику познакомиться с перспективными направлениями технологического развития, ведущими экспертами и найти единомышленников.

Ценности НТО

Национальная технологическая олимпиада — командные инженерные соревнования для школьников и студентов. Олимпиада создает уникальное пространство, основанное на общих ценностях и смыслах, которыми делятся все участники процесса: школьники, студенты, организаторы, наставники и эксперты. В основе Олимпиады лежит представление о современном технологическом образовании как новом укладе жизни в быстро меняющемся мире. Эта модель предполагает:

- доступность качественного обучения для всех, кто стремится к знаниям;
- возможность непрерывного развития;
- совместное формирование среды, где гуманитарные знания и новые технологии взаимно усиливают друг друга.

Это — образ общества будущего, в котором участники Олимпиады оказываются уже сегодня.

Решать прикладные задачи, нацеленные на умножение общественного блага

В заданиях Олимпиады используются актуальные вызовы науки и технологий, адаптированные под уровень школьников. Они имеют прикладной характер и отражают реальные потребности общества, а системное и профессиональное решение подобных задач способствует развитию общего блага. Олимпиада предоставляет возможность попробовать себя в этом направлении уже сегодня и найти единомышленников.

Создавать, а не только потреблять

Стремление к созданию нового ценится выше потребления готового, а ориентация на общественную пользу — выше личной выгоды. Это не исключает заботу о собственных интересах, но подчеркивает: творчество приносит больше удовлетворения, чем пассивное потребление. Олимпиада — совместный труд организаторов, партнеров и участников, в котором важнее стремление решать общие задачи, чем критика чужих усилий.

Работать в команде

Командная работа рассматривается не только как эффективный способ достижения целей, но и как основа для формирования сообщества, объединенного общими ценностями. Команда помогает раскрыть индивидуальность каждого, при этом сохраняя уважение к другим. Такие горизонтальные связи необходимы для реализации амбициозных технологических проектов. Олимпиада способствует формированию подобного сообщества и приглашает к его созданию всех заинтересованных.

Осваивать и ответственно развивать новые технологии

Сообщество Национальной технологической олимпиады — часть Кружкового движения НТИ, объединенные интересом к современным технологиям, стремлением

к их пониманию и созданию нового. Возможности технологий постоянно расширяются, однако развитие должно сопровождаться ответственностью. Этика инженера и ученого предполагает осознание последствий своих решений. Главное правило — создавая новое, не навредить.

Играть честно и пробовать себя

Ценится честная победа, достигнутая в рамках установленных правил. Это предполагает отказ от списывания, давления и манипуляций. Честная игра означает уважение к себе, команде и соперникам. Олимпиада поддерживается как безопасное пространство, где каждый может пробовать новое, не опасаясь ошибок, и постепенно становиться сильнее и увереннее в себе.

Быть человеком

Соревнования — это сложный и эмоционально насыщенный процесс, в котором особенно важны порядочность, вежливость и чуткость. Эмпатия, уважение и забота делают участие полезным и комфортным. Высоко ценится бережное отношение к людям и их труду, отказ от токсичной критики и готовность нести ответственность за слова и поступки. Участие в общем деле помогает не только окружающим, но и самому человеку.

Организационная структура НТО

НТО — межпредметная олимпиада. Спектр соревновательных направлений (профилей НТО) сформирован на основе актуального технологического пакета и связан с решением современных проблем в различных технологических отраслях. С полным перечнем направлений (профилей) можно ознакомиться на сайте НТО: https://ntcontest.ru/tracks/nto-school/.

Соревнования в рамках НТО проводятся по четырем трекам:

1. НТО Junior для школьников (5–7 классы).
2. НТО школьников (8–11 классы).
3. НТО студентов.
4. Конкурс цифровых портфолио «Талант НТО».

В 2024/25 учебном году 21 профиль НТО включен в Перечень олимпиад школьников, ежегодно утверждаемый Приказом Министерства науки и высшего образования Российской Федерации, а также в Перечень олимпиад и иных интеллектуальных и (или) творческих конкурсов, утверждаемый приказом Министерства просвещения Российской Федерации. Это дает право победителям и призерам профилей НТО поступать в вузы страны без вступительных испытаний (БВИ), получить 100 баллов ЕГЭ или дополнительные 10 баллов за индивидуальные достижения. Преимущества при поступлении победителям и призерам НТО предлагают более 100 российских вузов.

НТО для школьников 8–11 классов проводится в три этапа:

- Первый отборочный этап — заочный индивидуальный. Участникам предлагаются предметный тур, состоящий из задач по двум предметам, связанным

с выбранным профилем, а также инженерный тур, задания которого погружают участников в тематику профиля; образовательный модуль формирует теоретические знания и представления.

- Второй отборочный этап — заочный командный. На этом этапе участники выполняют как индивидуальные задания на проверку компетенций, так и командные задачи, соответствующие выбранному профилю.
- Заключительный этап — очный командный. В течение 5–6 дней команды участников со всей страны, успешно прошедшие оба отборочных этапа, соревнуются в решении комплексных прикладных инженерных задач.

Профили НТО 2024/25 учебного года и соответствующий уровень РСОШ

Профили II уровня РСОШ:

- Автоматизация бизнес-процессов.
- Автономные транспортные системы.
- Беспилотные авиационные системы.
- Водные робототехнические системы.
- Инженерные биологические системы.
- Наносистемы и наноинженерия.
- Нейротехнологии и когнитивные науки.
- Технологии беспроводной связи.
- Цифровые технологии в архитектуре.
- Ядерные технологии.

Профили III уровня РСОШ:

- Анализ космических снимков и геопространственных данных.
- Аэрокосмические системы.
- Большие данные и машинное обучение.
- Геномное редактирование.
- Интеллектуальные робототехнические системы.
- Интеллектуальные энергетические системы.
- Информационная безопасность.
- Искусственный интеллект.
- Летающая робототехника.
- Спутниковые системы.
- Кластер «Виртуальные миры»:
    - ◇ Разработка компьютерных игр.
    - ◇ Технологии виртуальной реальности.
    - ◇ Технологии дополненной реальности.

Профили без уровня РСОШ:

- Инфохимия.
- Квантовый инжиниринг.
- Новые материалы.
- Программная инженерия в финансовых технологиях.

- Современная пищевая инженерия.
- Умный город.
- Урбанистика.
- Цифровые сенсорные системы.
- Разработка мобильных приложений.

Обратите внимание на то, что в олимпиаде 2025/26 учебного года список профилей, в т. ч. входящих в РСОШ, и уровни РСОШ могут поменяться.

Участие в НТО старшеклассников может принять любой школьник, обучающийся в 8–11 классе. Чаще всего Олимпиада привлекает:

- учащихся технологических кружков, интересующихся инженерными и робототехническими соревнованиями;
- школьников, увлеченных олимпиадами и предпочитающих межпредметный подход;
- энтузиастов передовых технологий;
- активных участников хакатонов, проектных конкурсов и профильных школ;
- будущих предпринимателей, ищущих команду для реализации стартап-идей;
- любознательных школьников, стремящихся выйти за рамки школьной программы.

Познакомить школьников с НТО и ее направлениями, а также мотивировать их на участие в Олимпиаде можно с помощью специальных мероприятий — Урока НТО и Дней НТО. Методические рекомендации для педагогов по проведению Урока НТО и организации Дня НТО в образовательной организации размещены на сайте: https://nti-lesson.ru. Здесь можно подобрать и скачать готовые сценарии занятий и подборки материалов по различным направлениям Олимпиады.

Участвуя в НТО, школьники получают возможность работать с практико-ориентированными задачами в области прорывных технологий, собирать команды единомышленников, погружаться в профессиональное сообщество, а также заработать льготы для поступления в вузы.

По всей стране работают площадки подготовки к НТО, которые помогают привлекать участников и проводят мероприятия по подготовке к этапам Олимпиады. Такие площадки могут быть открыты на базе:

- школ и учреждений дополнительного образования;
- частных кружков по программированию, робототехнике и другим технологическим направлениям;
- вузов;
- технопарков и других образовательных и научно-технических организаций.

Любое образовательное учреждение, ученики которого участвуют в НТО или НТО Junior, может стать площадкой подготовки к Олимпиаде и присоединиться к Кружковому движению НТИ. Подробные инструкции о том, как стать площадкой подготовки, размещены на сайте: https://ntcontest.ru. Условия регистрации и требования к ним актуализируются с развитием Олимпиады, а обновленная информация публикуется перед началом каждого нового цикла.

Наставники НТО

В Национальной технологической олимпиаде большое внимание уделяется работе с **наставниками** — людьми, сопровождающими участников на всех этапах подготовки и участия в Олимпиаде. Наставник оказывает поддержку как в решении организационных вопросов, так и в развитии технических и социальных навыков школьников, включая умение работать в команде.

Наставником НТО может стать любой взрослый, готовый помогать школьникам развиваться и готовиться к участию в инженерных соревнованиях. Это может быть:

- учитель школы или преподаватель вуза;
- педагог дополнительного образования;
- руководитель кружка;
- родитель школьника;
- специалист из технологической области или представитель бизнеса.

Даже если наставник сам не обладает достаточными знаниями в определенной области, он может привлекать к подготовке коллег и экспертов, а также оказывать поддержку и организовывать процесс обучения для самостоятельных учеников. Сегодня сообщество наставников НТО насчитывает более **7 000 человек** по всей стране.

Главная цель наставника — **организовать системную подготовку к Олимпиаде в течение всего учебного года**, поддерживать интерес и мотивацию участников, а также помочь им справляться с возникающими трудностями. Также наставник фиксирует цели команды и каждого участника, чтобы в дальнейшем можно было проанализировать развитие профессиональных и личных компетенций.

Основные направления работы наставника

Организационные задачи:

- Информирование и мотивация: наставник рассказывает учащимся об НТО, ее этапах и преимуществах, помогает с выбором подходящего профиля, ориентируясь на интересы и способности школьников.
- Составление программы подготовки: формируется расписание и план занятий, организуется работа по освоению необходимых знаний и навыков.
- Контроль сроков: наставник следит за календарем Олимпиады и напоминает участникам о сроках решения заданий отборочных этапов.

Содержательная подготовка:

- Оценка компетенций участников: наставник помогает определить сильные и слабые стороны учеников и подбирает задания и материалы для устранения пробелов.
- Подготовка к отборочным этапам: помощь в изучении рекомендованных материалов, заданий прошлых лет, онлайн-курсы по профилям.
- Подготовка к заключительному этапу: разбираются задачи заключительных этапов прошлых лет, отслеживаются подготовительные мероприятия (очные и дистанционные), в которых наставник рекомендует ученикам участвовать.

Развитие личных и командных навыков:

- Формирование команд: наставник помогает сформировать сбалансированные команды для второго отборочного и финального этапов, распределить роли, при необходимости ищет участников из других регионов и организует онлайн-коммуникацию.
- Анализ прогресса и опыта: после каждого этапа проводится совместная рефлексия, обсуждаются успехи и трудности, выявляются зоны роста и направления для дальнейшего развития.
- Поддержка и мотивация: наставник поддерживает интерес и энтузиазм участников (особенно в случае неудачных результатов), помогает справиться с разочарованием и сохранить настрой на дальнейшее участие.
- Построение индивидуальной образовательной траектории: наставник помогает школьникам осознанно планировать дальнейшее обучение: выбирать курсы, участвовать в конкурсах, определяться с вузами и направлениями подготовки.

Поддержка наставников НТО

Работе наставников посвящен отдельный раздел на сайте НТО: https://ntcontest.ru/mentors/.

Для систематизации знаний и подходов к работе наставников в рамках инженерных соревнований разработан курс «Дао начинающего наставника: как сопровождать инженерные команды»: https://stepik.org/course/124633/. Курс формирует общие представления об их работе в области подготовки участников к инженерным соревнованиям.

Для совершенствования профессиональных компетенций по направлениям профилей создан курс «Дао начинающего наставника: как развивать технологические компетенции»: https://stepik.org/course/186928/.

Для организации занятий с учениками педагогам предлагаются образовательные программы, разработанные на основе многолетнего опыта организации подготовки к НТО. В настоящий момент они представлены по передовым технологическим направлениям:

- компьютерное зрение;
- геномное редактирование;
- водная, летающая и интеллектуальная робототехника;
- машинное обучение и искусственный интеллект;
- нейротехнологии;
- беспроводная связь, дополненная реальность.

Программы доступны на сайте: https://ntcontest.ru/mentors/education-programs/.

Регистрируясь на платформе НТО, наставники получают доступ к личному кабинету, в котором отображается расписание отборочных соревнований и мероприятий по подготовке, требования к знаниям и компетенциям при решении задач отборочных этапов.

Сообщество наставников НТО существует и развивается. Ежегодно Кружко-

вое движение НТИ проводит Всероссийский конкурс технологических кружков: https://konkurs.kruzhok.org/. Принять участие в конкурсе может каждый наставник.

В 2022 году было выпущено пособие «Технологическая подготовка инженерных команд. Методические рекомендации для наставников». Методические рекомендации предназначены для учителей технологий, а также наставников и педагогов кружков и центров дополнительного образования. Рекомендации направлены на помощь в процессе преподавания технологий в школе или в кружке. Пособие построено на примерах из реального опыта работы со школьниками, состоит из теоретических положений, посвященных популярным взглядам в педагогике на тему подготовки инженерных команд к соревнованиям. Электронное издание доступно по ссылке: https://journal.kruzhok.org/tpost/pggs3bp7y1-tehnologicheskaya-podgotovka-inzhenernih.

В нем рассмотрены особенности подготовки к пяти направлениям:

- Большие данные.
- Машинное обучение.
- Искусственный интеллект.
- Спутниковые системы.
- Летающая робототехника.

Для наставников НТО разработана и постоянно пополняется страница с материалами для профессионального развития: https://nto-forever.notion.site/c9b9cbd21542479b97a3fa562d15e32a.

1.2. Интеллектуальные энергетические системы

В современном мире развития цифровых технологий увеличивает необходимость внимания к критическим инфраструктурам, в том числе энергетике. Энергетика — сложная существующая система, и ее можно преобразовать, используя новые технологии, но нельзя новым технологиям подчинить. Это требует одновременно глубокого понимания технического и технологического устройства существующих энергосистем, понимания принципов и возможностей новых технологий. Навыки нужно не просто совместить, а тщательно синтезировать, чтобы проектировать системы более эффективные, чем существующие, обладающие большим модернизационным потенциалом и устойчивые в течение длительного времени — технически, технологически, финансово.

Энергосистемы будущего должны не просто существовать, но и стабильно работать — вот главное требование к критическим инфраструктурам. В совокупности это сложнейшая открытая задача, для решения которой важно выделить основные технологии и способы их взаимодействия. Это является ключевым моментом в подготовке принципиально нового поколения специалистов, обучение которых должно идти с опорой на современные способы обучения.

Внедрение цифровых технологий и развитие отечественных ИТ-решений в энергетике — это часть цифровой трансформации, одной из национальных целей развития Российской Федерации. Возможность познакомить школьников во время соревнований и подготовки к ним с современной инженерной задачей — управлением киберфизическими системами — позволяет не просто рассказать им об определенной области технологий, но и предоставить возможность приобретения личного практического опыта решения небольших, но реальных задач в данной области. Это и есть вход в деятельностную профориентацию, обеспечивающую осознанный выбор образовательной траектории и профессии в будущем.

Профиль посвящен моделированию энергетических систем ближайшего будущего. Реализация энергосети в архитектуре интернета энергии требует проектирования и строительства многочисленных надежных гибких энергосистем, эффективно распределяющих электроэнергию, в том числе за счет использования альтернативных источников и взаимодействия с внешним рынком мощностей.

Важное место в профиле занимают экономические модели, которые в настоящее время пока не распространены. В частности, биржа экономических микроконтрактов, составляющая один из главных компонентов технологии Smart Grid. Успешное участие в ее работе предполагает применение автоматизации, создание оптимальных стратегий и алгоритмов анализа параметров энергосети, в том числе с применением информационных технологий.

Получить реальный опыт командного управления сложными киберфизическими системами и написания программ для их управления, на практике познакомиться с понятиями гибкости и баланса, динамического ценообразования в проектах «Интернет энергии» участники могут в заключительном этапе профиля. Представленные задачи требуют знания математики школьного уровня, а именно, теории вероятно-

стей, геометрии и основ анализа.

Помимо школьных знаний и навыков требуется самостоятельное освоение таких тем, как теория игр, теория аукционов, программирование на языке Python, основы численных методов в решении математических задач. Большинство задач профиля требует реализацию их решения в программном виде, будь то компонент существующей системы (управляющий скрипт энергосистемы) или самостоятельная программа (инструменты для принятия решений).

Отборочные этапы и подготовительные мероприятия по профилю проектируются таким образом, чтобы в течение года постепенно ввести школьников в контекст и культуру профиля, а также подготовить их к финальной задаче — от этапа к этапу увеличивается их сложность и специфика. По мере продвижения команд к заключительному этапу проводятся вебинары, предоставляются дополнительные методические материалы по сложным темам.

Знакомство с профилем начинается с «Урока НТО» по профилю Интеллектуальные энергетические системы, который проводится в общеобразовательных учреждениях. Материалы для его проведения находятся на сайте https://nto-lesson.ru/.

Первый шаг в понимании тематики профиля — видеокурс от разработчиков профиля. Курс знакомит с базовыми понятиями энергетики, архитектурой интернета энергии и разработан для погружения в тему. Помимо этого сформированы еще два курса с видеоразбором задач второго отборочного этапа за прошлые годы.

Первый отборочный этап (дистанционный индивидуальный) состоит из предметных задач по информатике, математике и инженерного тура.

Предметный тур определяет уровень подготовки школьников по математике и информатике, устанавливая необходимую планку для участия во втором этапе и работы с его обучающими материалами. Задачи инженерного тура первого этапа сформулированы таким образом, чтобы отразить специфику профиля. Задачи проще, чем на втором туре, и, в отличие от них, могут быть решены индивидуально, а не только командой.

На втором отборочном этапе (дистанционном) участники создают команды и загружают общее решение. Функций у этого этапа несколько:

- формирование команд;
- отбор команд участников, способных решать сложные задачи;
- ознакомление с математическими и физическими концепциями, на которых будет построена задача заключительного этапа;
- информирование участников о требованиях к уровню программирования (идентичных тем, что будут предъявлены им на заключительном этапе).

Командная работа начинается именно во время второго тура. Помимо общих механизмов НТО, в профиле Интеллектуальные энергетические системы существует условие для командного взаимодействия — ограниченное число попыток на каждую задачу. Это стимулирует участников уже на начальном этапе договариваться об общей стратегии и обозначать свою позицию по отношению к каждой задаче внутри команды.

Структура профиля выстроена таким образом, чтобы не просто провести соревнования, а выстроить годичную образовательную программу. Для этого второй

и заключительный этапы связаны между собой (на рис. 1.2.1 представлена эта взаимосвязь). Так, во время второго отборочного этапа участники выполняют задания, отражающие часть большой финальной командной задачи, и знакомятся с базовыми концепциями — работают с вероятностными прогнозами, графами, аукционами, задачами на оптимизацию.

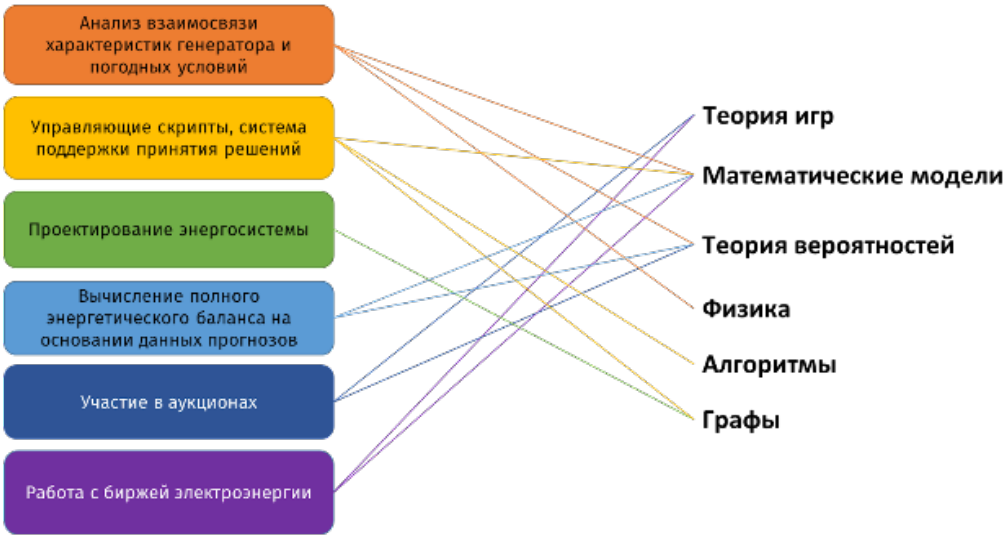


Рис. 1.2.1. Связь финальных тем и тем задач второго тура

В заключительном практическом туре перед участниками стоит задача моделирования собственных энергосистем и написание алгоритмов управления энергообеспечением. Главная цель — разработать стратегию оптимального построения умной энергосети и управления ею в условиях локальной конкуренции.

Участники проектируют экономические стратегии, анализируют прогнозы погоды и потребления, пишут скрипты по управлению энергообъектами. Побеждает та команда, чья энергосеть оказывается самой эффективной с точки зрения производства, накопления и расхода энергии, и самой «самостоятельной», способной выбрать оптимальный подход с учетом меняющихся условий.

Ежегодно сеть кардинально меняет свои законы и заготовки команд не работают, поэтому задача моделирует реальную инженерную ситуацию. Все базовые технологии известны, но новые объекты, экономические протоколы, балансы меняют механику и требуют создания обновленных продуктов и стратегий взаимодействия внутри сложной киберфизической системы.

Командная задача заключительного этапа представляет собой комплексную интегральную задачу, полное оптимальное решение которой чрезвычайно сложно, однако участники способны найти приближенное решение. Качество и сложность используемых приближений отражает глубину понимания, знаний и уровень их способностей.

Система оценки полностью автоматизирована и спроектирована таким образом, чтобы однозначно и численно оценить качество найденных и реализованных участниками приближенных решений.

Соревнования заключительного этапа профиля проводятся на программно-аппаратных комплексах лаборатории «Интеллектуальные энергетические системы», разработанных компанией «Полюс-НТ». Основой лаборатории являются стенды-трена-

жеры ИЭС — это модель поселения с потребителями энергии (жилые дома, больницы, заводы), электростанциями на альтернативных источниках энергии.

Стенды воссоздают погодные условия (ветер и освещенность) и рынок электроэнергии (многоуровневая биржа микроконтрактов). В 2024–25 гг. физические задачи определения параметров генераторов модифицированы:

- светильники на столах горят несинхронно, имитируя движение солнца с востока на запад;
- изменен характер прогнозов на силу ветра;
- прогнозы различаются на каждом из трех столов (но на соответствующих столах каждой из двух площадок они одинаковы);
- киберфизические солнечные электростанции (киберСЭС), введенные в 2023–24 гг., сохранены, но изменена их механика, а именно, скорость поворота и диапазон значений угла;
- ведены теплоэлектростанции (ТЭС), которые были на седьмом финале ИЭС, проходившем дистанционно;
- в игре изменился характер экономической модели: аукцион ведется по принципу all-pay (платят все) на пакеты объектов с заранее выставленными тарифами, а в рамках моделирования начисляются экологические баллы, которые в конце игры конвертируются в баллы итогового счета;
- банк для итоговой выплаты формируется из налогов, взимаемых с команд на протяжении моделирования;
- введена механика дискретного износа сетей: в течение моделирования ветки подстанций накапливают износ, после превышения которого ветка отключается на один такт и автоматически включается;
- с помощью управляющего скрипта команды могут вручную отключать ветки для сброса накопленного износа.

Помимо командной задачи (практический тур) на заключительном этапе участникам решают индивидуальные задачи по предметам: информатика и математика (предметный тур). Это необходимо для того, чтобы объективно проверить индивидуальные знания участников, косвенно оценить индивидуальный вклад участников в результат командной работы.

Таким образом, за время заключительного этапа, наряду с предметными знаниями, участники получают опыт командного решения с возможностью выявления индивидуального вклада каждого в решение общей задачи. Это дает возможность выявить наиболее одаренных в научно-техническом отношении школьников.

Участие в данном профиле позволяет им в будущем сделать осознанный выбор профессии, а также расширяет популярность знаний в области энергетики и управления сложными системами. Собственный реальный опыт управления сложной системой дает возможность уже в школьном возрасте понять свое отношение и выбрать целевой профильный вуз, а значит, выбрать эффективную образовательную траекторию.

2. Первый отборочный этап

2.1. Работа наставника НТО на этапе

Педагог-наставник играет важную роль в подготовке участника к первому отборочному этапу Национальной технологической олимпиады. На этом этапе школьникам предстоит справиться как с предметными задачами, соответствующими профилю, так и с заданиями инженерного тура, погружающими в выбранную технологическую область.

Наставник может организовать подготовку участника, используя разнообразные форматы и ресурсы:

- Разбор заданий прошлых лет. Совместный анализ задач отборочного этапа предыдущих лет позволяет понять структуру, уровень сложности и типичные подходы к решению. Это формирует у школьника устойчивые стратегии работы с олимпиадными заданиями.
- Мини-соревнования. Проведение тренировочных турниров с заданиями предметных олимпиад муниципального уровня помогает развить соревновательный навык, тренирует скорость и уверенность при решении задач в ограниченное время.
- Углубленные занятия. Наставник может выстроить образовательную траекторию, опираясь на рекомендации разработчиков профиля, и провести занятия по ключевым темам. Это особенно важно для системного понимания предметной области.
- Использование онлайн-курсов. Для самостоятельной подготовки и проверки знаний участник может использовать предметные курсы НТО, размещенные на платформах Степик и Яндекс Контест. Наставник может также организовать занятия с использованием этих материалов в рамках групповой или индивидуальной подготовки.
- Привлечение внешних экспертов. Если у наставника нет достаточной экспертизы в какой-либо предметной области, он может пригласить других педагогов или специалистов для проведения тематических занятий.
- Поддержка в инженерном туре. Инженерный тур включает теоретические материалы и задания, помогающие глубже погрузиться в тематику профиля. Наставник может сопровождать изучение курса, помогать в разборе теоретических вопросов и тренировать участника на практических задачах.

Таким образом, наставник не только помогает систематизировать подготовку, но и мотивирует участника, создавая для него комфортную и продуктивную образовательную среду.

2.2. Предметный тур. Информатика

2.2.1. Первая волна. Задачи 8–11 класса

Задачи первой волны предметного тура по информатике открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63452/enter/.

Задача 2.2.1.1. Ускорение ускорения (10 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Рассмотрим модель движения тела. Будем фиксировать такие параметры, как координата, скорость, ускорение и ускорение ускорения (рывок). Если некоторый параметр равен a и имеет скорость изменения v, то в следующий момент времени этот параметр будет равен $a + v$.

Например, если тело имело координату, равную 10, скорость, равную 20, ускорение, равное 30 и ускорение ускорения, равное 40, то в следующий момент оно будет иметь координату 30, скорость 50 и ускорение 70. Ускорение ускорения будем считать в этой задаче постоянной величиной.

Задача довольно проста: тело в начальный момент времени 0 находится в точке с координатой 0, скоростью 0 и ускорением 0. На это тело действует постоянное ускорение ускорения, равное 6. Требуется определить, в точке с какой координатой окажется это тело в момент времени t.

Формат входных данных

В единственной строке находится одно число t, где $0 \leqslant t \leqslant 10^6$.

Формат выходных данных

Вывести одно число — координату, в которой окажется тело в момент времени t.

Примеры

Пример №1

Стандартный ввод
6
Стандартный вывод
120

Пример №2

Стандартный ввод
2
Стандартный вывод
0

Пример №3

Стандартный ввод
1000000
Стандартный вывод
999997000002000000

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int t;
    cin >> t;
    cout << ((t * (t - 1)) * (t - 2)) << endl;
}
```

Задача 2.2.1.2. Двойное остекление (15 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

У деда Василия есть два прямоугольных куска стекла. Один из них имеет размеры $a \times b$, другой — $c \times d$. Дед собирается из этих кусков сделать окно с двойным остеклением. Он хочет, чтобы окно было обязательно квадратным и как можно большим по размеру. Дед должен вырезать из имеющихся у него прямоугольников два одинаковых квадрата максимально возможного размера. Нужно написать программу, которая по заданным a, b, c, d найдет максимальные размеры квадратного окна. Имейте ввиду, что оба квадрата могут быть вырезаны и из одного прямоугольного куска стекла.

Формат входных данных

На вход подаются две строки. В первой строке находятся размеры первого прямоугольника a, b через пробел, во второй — размеры второго прямоугольника c, d через пробел, где $1 \leqslant a, b, c, d \leqslant 10^9$.

Формат выходных данных

Вывести одно число — максимальную сторону квадратного двойного окна, которое можно вырезать из заданных на входе прямоугольных кусков стекла. Ответ может быть нецелым, требуется вывести его с точностью 1 знак после десятичной точки.

Примеры

Пример №1

Стандартный ввод
5 10 9 6
Стандартный вывод
5

Пример №2

Стандартный ввод
4 10 9 6
Стандартный вывод
4.5

Комментарий

Второй пример показывает, что иногда лучше вырезать оба квадрата из одного и того же куска стекла.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    double a, b, c, d;
    cin >> a >> b >> c >> d;
    double a0 = min({a, b, c, d});
    double a1 = min(max(a, b) / 2.0, min(a, b));
    double a2 = min(max(c, d) / 2.0, min(c, d));
    double ans =  max({a0, a1, a2});
    if( (int)ans == ans ){
        int ians = ans;
        cout << ians << endl;
        return 0;
    }
    cout.precision(1);
    cout << fixed<< ans << endl;
}
```

Задача 2.2.1.3. О золотой рыбке и... досках (20 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

После событий известной сказки А. С. Пушкина старик решил принципиально не пользоваться услугами золотой рыбки. Поэтому для того чтобы изготовить новое корыто, он честно заготовил n одинаковых досок.

Но гостивший в это время у старика со старухой внук решил, что ему нужно научиться пилить. И, не сказав ничего своему деду, внук быстро распилил каждую из досок на две части. В итоге у старика оказались $2n$ кусков досок. Самое интересное, что все эти куски оказались разными по длине, но имели целочисленные размеры. К сожалению, старик забыл, какова была исходная длина целых досок.

Формат входных данных

В первой строке задается целое число n — исходное количество целых досок, где $1 \leqslant n \leqslant 10^5$.

Во второй строке заданы $2n$ целых чисел d_i — длины всех кусков, которые получились после «тренировки» внука, где $1 \leqslant d_i \leqslant 10^9$. Гарантируется, что эти числа попарно различны, и их можно разбить на пары одинаковых по сумме чисел.

Все эти части досок пронумерованы от 1 до $2n$ в том порядке, в котором они заданы на входе.

Формат выходных данных

В первую строку вывести одно число — исходную длину целых досок.

В следующих n строках вывести пары номеров кусков досок, которые составляют по длине целые доски. Номера выводить через один пробел, внутри пары сначала должен идти меньший номер, затем больший. Пары должны быть выведены в порядке возрастания первых номеров в парах.

Примеры

Пример №1

Стандартный ввод
3 4 8 2 3 6 7

Стандартный вывод
10 1 5 2 3 4 6

Комментарий

Отсортируем куски и далее будем брать один из начала и второй к нему из конца.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int n;
    cin >> n;
    vector<pair<int, int> > v(2 * n);
     for(int i = 0; i < 2 * n; i++){
        int d;
        cin >> d;
        v[i] = {d, i + 1};
    }
    sort(v.begin(), v.end());
    vector<pair<int, int> > ans(n);
    for(int i = 0; i < n; i++){
```

```
16          ans[i] = {v[i].second, v[2 * n - i - 1].second};
17          if(ans[i].first > ans[i].second){
18              swap(ans[i].first, ans[i].second);
19          }
20      }
21      sort(ans.begin(), ans.end());
22      cout << v[0].first + v.back().first<< endl;
23      for(int i = 0; i < n; i++){
24          cout << ans[i].first<<' '<< ans[i].second<< endl;
25      }
26  }
```

Задача 2.2.1.4. Бонусы и экономия (25 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Технология производства некоторой металлической детали предполагает вытачивание ее из металлической заготовки. При этом образуются стружки, которые не стоит выкидывать. Ведь из a комплектов стружек (оставшихся после обработки a заготовок) можно бесплатно выплавить еще одну заготовку, которую снова можно использовать для выточки детали и создания еще одного комплекта стружек.

Заготовки можно купить на оптовом складе, при этом в целях привлечения клиентов, проводится акция «купи b заготовок, тогда еще одну получишь бесплатно».

Требуется изготовить c деталей. Нужно определить минимальное число заготовок, которые нужно купить за деньги, чтобы с учетом бонусных заготовок и экономии на стружках можно было изготовить требуемое число деталей.

Формат входных данных

В одной строке через пробел заданы три целых числа a, b, и c такие, что $2 \leqslant$ $\leqslant a \leqslant 10^{18}$, $1 \leqslant b, c \leqslant 10^{18}$.

Формат выходных данных

Вывести одно целое число — минимальное количество заготовок, которые нужно купить, чтобы с учетом всех бонусов и экономии выточить c конечных деталей.

Примеры

Пример №1

Стандартный ввод
4 5 41
Стандартный вывод
26

Примечания

В примере из условия нужно закупить 26 заготовок. Тогда за каждые пять купленных заготовок будет предоставлена одна бесплатная, итого по акции добавится еще пять заготовок, то есть получится 31 заготовка. Далее из 31 заготовки выточится 31 деталь, останется 31 комплект стружек. Из каждых четырех комплектов выплавится дополнительная заготовка, получится семь заготовок и три комплекта стружек. Из семи заготовок выточится семь деталей и останется семь комплектов стружек, три комплекта стружек осталось с первого шага, итого 10 комплектов стружек. Из них выплавится еще две заготовки, дающие две детали и два комплекта стружек. Собрав эти два комплекта с двумя, оставшимися от 10, получим еще одну заготовку, из которой выточится еще одна деталь. Останется один комплект стружек, который уже никак не получится использовать. Итого будет произведена $31 + 7 + 2 + 1 = 41$ деталь.

Комментарий

Методом бинарного поиска можно подобрать минимальное необходимое количество исходных заготовок.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
int f1(int M, int a){
    int res = 0, z = 0;
    while(1){
        if(M == 0 && z < a){
            return res;
        }
        res += M;
        M = M + z;
        z = M % a;
        M = M / a;
    }
}
```

```
int f2(int M, int b){
    return M + M / b;
}
signed main(){
    int a, b, c;
    cin >> a >> b >> c;
    int L = 0, R = 1;
    while(f1(R, a) <= c){
        R *= 2;
    }
    while(R - L > 1){
        int M = (R + L) / 2;
        if(f1(M, a) < c){
            L = M;
        }
        else{
            R = M;
        }
    }
    int z = R;
    L = 0, R = 1;
    while(f2(R, b) <= z){
        R *= 2;
    }
    while(R - L > 1){
        int M = (R + L) / 2;
        if(f2(M, b) < z){
            L = M;
        }
        else{
            R = M;
        }
    }
    cout << R << endl;
}
```

Задача 2.2.1.5. Сон таксиста (30 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Одному таксисту приснился красочный сон. Во сне он живет и работает в некотором городе, где абсолютно все улицы с односторонним движением. Эти улицы устроены так, что невозможно проехать с какого-либо перекрестка так, чтобы вернуться обратно на этот же перекресток, то есть в дорожной сети города нет циклов.

Таким образом, если с перекрестка A можно попасть по направлению движения улиц на перекресток B, то люди вызывают такси, иначе их везет специальный муниципальный подземный транспорт бесплатно.

В связи с такими странными правилами, таксистам в этом городе разрешено законом везти пассажира по любому маршруту, не нарушающему направления движения. Все в этом городе привыкли к такой ситуации и абсолютно спокойно относятся к тому, что таксисты везут их самым длинным путем. Разумеется, заработок таксиста за одну поездку прямо пропорционален ее длине. Для упрощения будем считать, что стоимость 1 км поездки составляет ровно 1 руб.

Схема дорог города задана. Перекрестки города пронумерованы числами от 1 до n. Таксист в своем сне находится на перекрестке номер S. Напишите программу, которая подскажет ему, сколько он максимально сможет заработать, когда ему придет заказ от клиента. Так как он не знает, куда попросит его везти клиент, нужно для каждого перекрестка от 1 до n указать максимальную стоимость поездки до этого перекрестка из пункта S на такси. Если по правилам на такси добраться из пункта S до какого-то перекрестка нельзя, вывести -1.

Формат входных данных

Дорожная сеть задана следующим образом: в первой строке находятся два числа через пробел n и m — число перекрестков и число улиц в городе, где $2 \leqslant n, m \leqslant 2 \cdot \cdot 10^5$.

В следующих m строках задана очередная односторонняя улица в виде трех чисел A, B, d через пробел, где A — начало улицы, B — конец улицы и d — ее длина. $1 \leqslant A, B \leqslant n$, $1 \leqslant d \leqslant 10^9$. Гарантируется, что в этой дорожной сети нет циклов. Некоторые пары перекрестков могут быть соединены двумя и более односторонними улицами. Дорожная сеть может быть неплоской за счет мостов и тоннелей.

В последней строке ввода содержится номер стартового перекрестка S, $1 \leqslant S \leqslant \leqslant n$.

Формат выходных данных

Вывести n чисел в одну строку через пробел. i-е число обозначает длину самого длинного пути с перекрестка номер S до перекрестка номер i. Если до перекрестка номер i от S нельзя доехать, не нарушая правила движения, вывести -1.

Примеры

Пример №1

Стандартный ввод
10 20 9 10 15 9 8 3 8 10 7 7 8 4 7 10 10 5 8 2 5 9 10

Стандартный ввод

```
5 6 5
7 6 5
4 6 8
3 6 4
3 4 6
5 3 2
2 5 2
2 3 3
3 1 5
1 4 2
2 1 7
4 7 4
6 8 1
5
```

Стандартный вывод

```
7 -1 2 9 0 18 13 19 10 26
```

Комментарий

Задача решается методом динамического программирования на ориентированном ациклическом графе.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
int n, m;
vector<vector<pair<int, int> > > G;
vector<int> order, used;
void dfs(int a){
    used[a] = 1;
    for(auto to : G[a]){
        if(!used[to.first]){
            dfs(to.first);
        }
    }
    order.push_back(a);
}
signed main(){
    cin >> n >> m;
    G.resize(n + 1);
    used.resize(n + 1, 0);
    for(int i = 0; i < m; i++){
        int a, b, d;
        cin >> a >> b >> d;
        G[a].push_back({b, d});
    }
```

```
25      int s;
26      cin >> s;
27      dfs(s);
28      reverse(order.begin(), order.end());
29      vector<int> dp(n + 1, -1);
30      dp[s] = 0;
31      for(auto el : order){
32          for(auto to : G[el]){
33              dp[to.first] = max(dp[to.first], dp[el] + to.second);
34          }
35      }
36      for(int i = 1; i <= n; i++){
37          cout << dp[i] <<' ';
38      }
39  }
```

2.2.2. Вторая волна. Задачи 8–11 класса

Задачи второй волны предметного тура по информатике открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63454/enter/.

Задача 2.2.2.1. Игра на планшете (10 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Маленький Андрей изучает геометрические фигуры при помощи игры на планшете. У него есть прямоугольные треугольники четырех цветов и ориентаций: желтые, зеленые, красные и синие. Для каждой разновидности треугольников есть заданное количество экземпляров этих треугольников. Более точно: у Андрея есть a желтых, b зеленых, c красных и d синих треугольников. Помимо этого у него есть прямоугольная таблица $n \times m$.

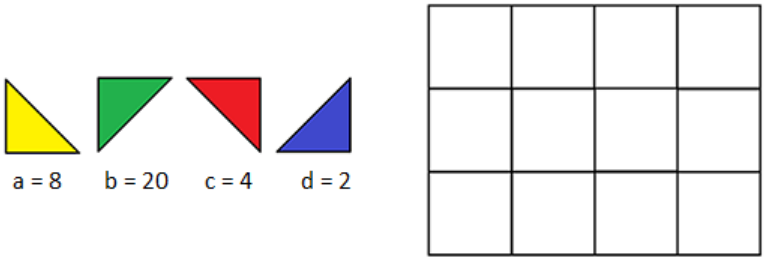


Рис. 2.2.1

Треугольники одного цвета имеют одну и ту же ориентацию, которую нельзя поменять. Андрей может только взять очередной треугольник и переместить его параллельным сдвигом в одну из ячеек этой прямоугольной таблицы. При этом в одну ячейку можно поместить либо вместе желтый и красный треугольники, либо вместе зеленый и синий, либо один любой треугольник из имеющихся.

Андрей хочет расположить в ячейках таблицы как можно больше треугольников из тех, что у него имеются. Нужно подсказать ему максимальное количество треугольников, которые получится разместить в таблице.

Формат входных данных

В первой строке содержатся четыре целых числа a, b, c и d через пробел — количество желтых, зеленых, красных и синих треугольников соответственно.

Во второй строке содержатся два целых числа n и m через пробел — размеры прямоугольной таблицы.

Все числа в пределах от 1 до 10^9.

Формат выходных данных

Вывести одно число — максимальное количество треугольников, которые можно при заданных условиях разместить в таблице.

Примеры

Пример №1

Стандартный ввод
`8 20 4 2` `3 4`

Стандартный вывод
`18`

Примечания

На рис. 2.2.2 представлен один из примеров размещения 18 треугольников из 34 заданных на входе.

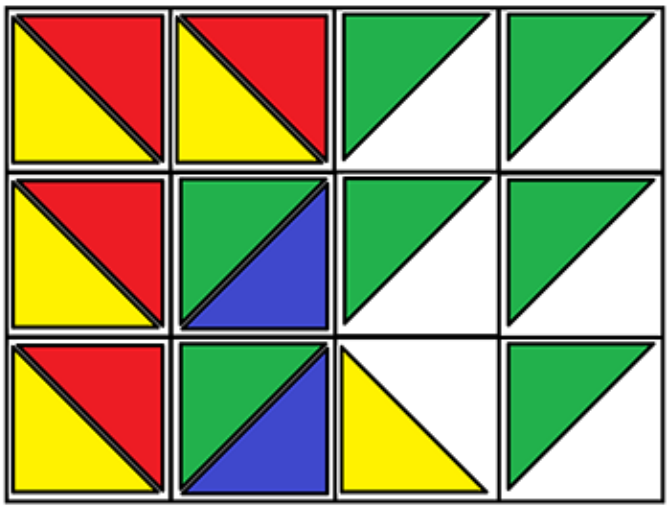


Рис. 2.2.2

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int a, b, c, d, n, m;
    cin >> a >> b >> c >> d >> n >> m;
    if(a > c){
        swap(a, c);
    }
    if(b > d){
        swap(b, d);
    }
    int f = a + b;
    int k = n * m;
    if(k <= f){
        cout << k * 2;
        return 0;
    }
    k -= f;
    c -= a;
    d -= b;
    cout << f * 2 + min(k, c + d) << endl;
}
```

Задача 2.2.2.2. Старая задача на новый лад (15 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Одна старая задача имеет следующий вид:

«Разбить число 45 на сумму четырех слагаемых так, что если к первому прибавить 2, из второго вычесть 2, третье умножить на 2, а четвертое разделить на 2, то получится одно и то же число».

Ответ к этой задаче — четыре числа 8, 12, 5 и 20. Можно убедиться, что в сумме они дают число 45, а если с каждым из них проделать соответствующую арифметическую операцию, то получится одно и то же число 10.

Необходимо решить чуть более общую задачу: даны числа n и k. Нужно представить число n в виде суммы четырех целых неотрицательных слагаемых $a + b + c + d$ таких, что $a + k = b - k = c \cdot k = d/k$. Гарантируется, что для заданных n и k такое разбиение существует.

Формат входных данных

В одной строке через пробел два числа n и k, где $1 \leqslant n \cdot k \leqslant 10^{18}$.

Формат выходных данных

Вывести через пробел в одну строку четыре целых неотрицательных числа a, b, c, d таких, что $a + b + c + d = n$ и $a + k = b - k = c \cdot k = d \,/\, k$.

Примеры

Пример №1

Стандартный ввод
`45 2`

Стандартный вывод
`8 12 5 20`

Пример №2

Стандартный ввод
`128 7`

Стандартный вывод
`7 21 2 98`

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int n, k;
    cin >> n >> k;
    int x = (k * n) / (k * k + 2 * k + 1);
    cout << x - k <<' '<< x + k <<' '<< x / k <<' '<< x * k << endl;
}
```

Задача 2.2.2.3. Ладья и обязательная клетка (20 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Шахматная ладья находится в левом верхнем углу прямоугольного поля, разбитого на клетки размером $n \times m$. n обозначает число строк, m — число столбцов. Она хочет попасть в правую нижнюю клетку этого поля кратчайшим путем. Ладья может передвигаться либо вправо, либо вниз на любое количество клеток. Ладья обязана посетить заданную клетку с координатами (x, y), где x — номер строки этой клетки, а y — номер ее столбца.

Требуется найти количество способов построить путь ладьи из левого верхнего угла в правый нижний, которые проходят через обязательную клетку с заданными координатами.

Формат входных данных

В первой строке находятся два числа через пробел: n — число строк и m — число столбцов прямоугольного поля, $2 \leqslant n, m \leqslant 25$. Во второй строке через пробел находятся координаты (x, y) обязательной для посещения клетки, где $1 \leqslant x \leqslant n$, $1 \leqslant y \leqslant m$. Координаты x и y не совпадают с координатами левой верхней и правой нижней клеток.

Формат выходных данных

Вывести одно число — количество кратчайших путей ладьи из верхней левой в правую нижнюю клетку, проходящих через заданную клетку.

Примеры

Стандартный ввод
3 4 2 3

Стандартный вывод
6

Примечания

На рис. 2.2.3 представлены шесть путей, которыми ладья может пройти по полю размером 3×4, обязательно посещая по пути клетку $(2, 3)$.

Комментарий

Задачу можно решить как комбинаторными методами (произведение биномиальных коэффициентов), так и динамическим программированием.

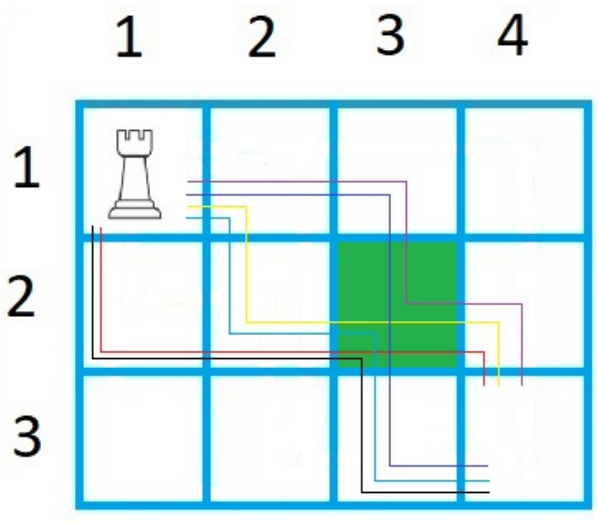


Рис. 2.2.3

Решение

Ниже представлено решение на языке C++.

C++

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    vector<vector<int> > bc(51, vector<int>(51, 0));
    bc[0][0] = 1;
    for(int i = 1; i <= 50; i++){
        for(int j = 0; j < 51; j++){
```

```
            bc[i][j] += bc[i - 1][j];
            if(j - 1 >= 0){
                bc[i][j] += bc[i - 1][j - 1];
            }
        }
    }
    int n, m, x, y;
    cin >> n >> m >> x >> y;
    int d1 = bc[x - 1 + y - 1][x - 1];
    int d2 = bc[n - x + m - y][n - x];
    int ans = d1 * d2;
    cout << ans << endl;
}
```

Задача 2.2.2.4. Танец с цифрами (25 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Десять танцоров репетируют на сцене новый танец. Каждый танцор одет в футболку, на которой написана одна из цифр от 1 до 9, цифры могут повторяться. Изначально они стоят в некотором порядке слева направо, и их цифры образуют некоторое десятизначное число A. Далее во время всего танца участники либо разбиваются на пять пар рядом стоящих танцоров и одновременно меняются местами внутри своих пар, либо самый левый танцор перемещается на самую правую позицию и становится самым правым танцором.

Сын постановщика танца от скуки на бумаге выписывает все получающиеся при каждом перемещении десятизначные числа. Так как танец длинный, то в итоге на бумаге окажутся все возможные числа, которые в принципе могут появится при этих условиях. Нужно найти разницу между самым большим и самым маленьким из этих чисел.

Формат входных данных

На вход подается одно десятизначное число A, обозначающее начальное расположение танцоров. В числе могут встречаться цифры от 1 до 9, некоторые из них могут повторяться.

Формат выходных данных

Вывести одно число, равное разности самого большого и самого маленького из чисел, которые могут быть получены во время танца.

Примеры

Пример №1

Стандартный ввод
1456531355
Стандартный вывод
5182160085

Примечания

Самое маленькое число, которое можно получить в примере, равно 1353155456, самое большое равно 6535315541.

Покажем, как получить эти числа из исходного числа 1456531355. Сначала получим самое большое следующим образом: две левых цифры, 1 и 4, переместим вправо, получим 5653135514, потом поменяем в парах цифры местами и получим самое большое — 6535315541. Далее опять поменяем порядок в парах и в числе 5653135514 переместим три левых цифры 5, 6 и 5 вправо, получим 3135514565 и здесь снова поменяем порядок в парах, получим самое маленькое — 1353155456. Таким образом, искомая разница равна 5182160085.

Решение

Ниже представлено решение на языке C++.

C++

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    string s;
    cin >> s;
    string mx = s, mn = s;

    for(int i = 0; i < 5; i++){
        for(int j = 0; j < 10; j++){
            mx = max(mx, s);
            mn = min(s, mn);
            if(j < 9){
                s = s.substr(1) + s[0];
            }
        }
        for(int j = 0; j < 5; j++){
            swap(s[2 * j], s[2 * j + 1]);
        }
    }
    stringstream ssmn;
    ssmn << mn;
    int imn;
    ssmn >> imn;
    stringstream ssmx;
```

```
26      ssmx << mx;
27      int imx;
28      ssmx >> imx;
29      cout << imx - imn << endl;
30  }
```

Задача 2.2.2.5. Трудная сортировка (30 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 3 с.

Ограничение по памяти: 64 Мбайт.

Условие

Иннокентий работает в отделе сортировки перестановок, подотделе сортировки вставками. Его задача заключается в сортировке перестановок, предоставленных заказчиками. Перестановкой длины n называется такая последовательность чисел, в которой встречаются все числа от 1 до n без повторений в некотором порядке.

Перестановка считается отсортированной, если в ней все числа расположены по возрастанию, то есть она имеет вид $1, \ldots, n$.

Иннокентий начинает рабочий день с пустой последовательности чисел. За день он сортирует вставками перестановку длины n. В начале каждой операции вставки он получает очередное число a_i из перестановки заказчика, после чего обрабатывает его, вставляя в отсортированную последовательность из ранее полученных чисел. После каждого такого добавления последовательность уже обработанных чисел должна быть отсортирована по возрастанию.

Перед тем как вставить число a_i в последовательность, он может выбрать, с какого края последовательности начать вставку. Далее он устанавливает число a_i с этого края и последовательно меняет вставляемое число с рядом стоящим числом b_j до тех пор, пока число a_i не встанет на свое место. На каждую перестановку вставляемого числа a_i с числом b_j Иннокентий тратит b_j единиц энергии.

Дана перестановка длины n из чисел a_i в том порядке, в котором Иннокентий их будет обрабатывать. Подскажите ему, какое минимальное количество энергии ему потребуется потратить, чтобы отсортировать всю перестановку.

Формат входных данных

В первой строке находится одно целое число n — длина перестановки, где $1 \leqslant \leqslant n \leqslant 2 \cdot 10^5$.

Во второй строке содержится n целых чисел a_i через пробел в том порядке, в котором они поступают на обработку Иннокентию. Гарантируется, что эти числа образуют перестановку длины n, то есть каждое число от 1 до n содержится в заданном наборе ровно один раз.

Формат выходных данных

Вывести одно число — минимальные суммарные энергозатраты Иннокентия для сортировки вставками заданной на входе перестановки.

Примеры

Пример №1

Стандартный ввод
9 2 9 1 5 6 4 3 8 7
Стандартный вывод
43

Примечания

Первым устанавливается число 2. Оно ни с чем не меняется местами, поэтому затрат нет.

Далее устанавливается число 9. Выбираем правый край и ставим его туда без потерь энергии.

Затем устанавливаем число 1. Выбираем левый край, ставим его туда и снова потерь нет.

Теперь нужно вставить число 5. Если его вставлять с правого края, придется менять местами с 9, а если с левого, то с 1 и 2, что суммарно явно лучше. Итого затраты на вставку 5 равны 3.

Число 6 снова лучше вставить слева, затраты на его вставку равны 8.

Число 4 вставим слева за 3.

Число 3 так же слева за 3.

А вот число 8 лучше вставить справа за 9.

И осталось число 7. Если вставлять слева, то затратим 21, а если справа, то всего 17.

Итого на сортировку заданной перестановки потратили: $0 + 0 + 0 + 3 + 8 + 3 + 3 + 9 + 17 = 43$.

Комментарий

Построим дерево отрезков на сумму, при обработке числа a будем находить, какая сумма на данный момент меньше: от 1 до $a - 1$ или от $a + 1$ до n. Прибавим ее к ответу и поместим в позицию a это число a.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
const int LG = 19;
int N = (1 << LG);
vector<int> tr(2 * N, 0);
void upd(int pos, int x){
    pos += N;
    tr[pos] = x;
    pos /= 2;
    while(pos){
        tr[pos] = {tr[2 * pos]+ tr[2 * pos + 1]};
        pos /= 2;
    }
}
int get(int l, int r){
    l += N;
    r += N;
    int res = 0;
    while(l <= r){
        if(l % 2 == 1){
            res += tr[l];
        }
        if(r % 2 == 0){
            res += tr[r];
        }
        l = (l + 1) / 2;
        r = (r - 1) / 2;
    }
    return res;
}
signed main(){
    int n, a;
    cin >> n;
    int ans = 0;
    for(int i = 0; i < n; i++){
        cin >> a;
        int sl = get(0, a - 1);
        int sr = get(a + 1, N - 1);
        ans += min(sl, sr);
        upd(a, a);
    }
    cout << ans << endl;
}
```

2.2.3. Третья волна. Задачи 8–11 класса

Задачи третьей волны предметного тура по информатике открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63456/enter/.

Задача 2.2.3.1. Туннель (10 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Рассмотрим классическую задачу прохождения группы с одним фонариком по туннелю. Есть четыре человека, и у них есть один фонарик. Нужно перевести всю группу на другой конец туннеля. По туннелю можно проходить только с фонариком и только либо вдвоем, либо в одиночку. По этой причине придется сделать пять рейсов по туннелю: три рейса туда и два рейса обратно. Туда идут двое, обратно — один, возвращая фонарик еще не прошедшей части группы. У каждого из четырех человек своя скорость передвижения по туннелю, но некоторые скорости могут совпадать. Двое идут со скоростью самого медленного в этой паре. Нужно найти минимальное время, за которое можно перевести группу по туннелю.

Здесь, в зависимости от скоростей персонажей, есть две стратегии. Проиллюстрируем их на примерах.

Пусть есть люди A, B, C, D. У A — время прохождения туннеля 1 мин, у B — 4 мин, у C — 5 мин, у D — 10 мин. Здесь работает наиболее очевидная стратегия: самый быстрый переводит текущего и возвращается с фонариком обратно за следующим. При этой стратегии нужно проходить так:

- A, B туда, затрачено 4 мин;
- A обратно, затрачена 1 мин;
- A, C туда, затрачено 5 мин;
- A обратно, затрачена 1 мин;
- A, D туда, затрачено 10 мин.

Общее время $4 + 1 + 5 + 1 + 10 = 21$ мин.

Но не всегда эта стратегия оптимальна. Уменьшим время прохождения туннеля персонажем B до 2 мин. По вышеопределенной стратегии будет 19 мин ($2 + 1 + 5 + + 1 + 10 = 19$), но имеется более быстрое решение:

- A, B туда, затрачено 2 мин;
- A обратно, затрачена 1 мин;
- C, D туда, затрачено 10 мин;
- B обратно, затрачено 2 мин;
- A, B туда, затрачено 2 мин.

Общее время $2 + 1 + 10 + 2 + 2 = 17$ мин.

Заметим, что для предыдущего примера такая стратегия не работает: $4+1+10+ + 4 + 4 = 23$ мин.

Если же персонаж B проходит туннель за 3 мин (а все остальные так же, как и в примерах), то независимо от стратегии будет затрачено 20 мин. В этом случае

считаем, что работает первая стратегия.

Поразмыслив, станет понятно, от какого условия зависит выбор стратегии. Далее будем всегда считать, что A движется не медленнее B, B движется не медленнее C, C движется не медленнее D.

Дано время прохождения туннеля персонажами A, C, D. Нужно найти границу `border` для B такую, что если определить для B время прохождения строго меньшее, чем `border`, то выгодна вторая стратегия, иначе — первая.

Формат входных данных

В одной строке задано три целых чисел через пробел — время прохождения туннеля персонажами A, C, D. Времена даны по неубыванию. Все числа на входе в пределах от 1 до 100.

Формат выходных данных

Вывести одно число — границу `border` для B такую, что если определить время прохождения им туннеля строго меньше, чем `border`, нужно использовать вторую стратегию, иначе — первую. Ответ может быть нецелым, поэтому вывести его нужно с одним знаком после десятичной точки.

Примеры

Пример №1

Стандартный ввод
`1 5 10`
Стандартный вывод
`3`

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int A, C, D;
    cin >> A >> C >> D;
    cout.precision(1);
    cout << fixed << (A + C) / 2.0 << endl;
}
```

Задача 2.2.3.2. Математический пазл (15 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

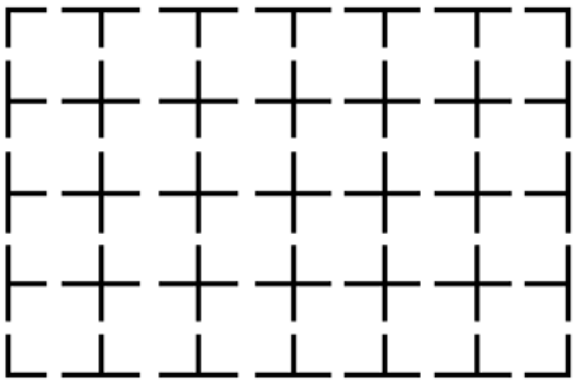


Рис. 2.2.4

Компания по производству пазлов решила освоить принципиально новый тип головоломок. Для этого берется прямоугольная решетка размера $n \times m$, каждый ее столбец и строка разрезаются посередине пополам. После этого образуются фигуры трех типов: четыре уголка, $2 \cdot (n+m-2)$ т-образных фигур и $(n-1) \cdot (m-1)$ крестиков.

Тому, кто решает головоломку, требуется сложить из этих фигур исходную прямоугольную решетку. При этом необходимо использовать абсолютно все имеющиеся в наличии фигуры.

Формат входных данных

В первой строке заданы через пробел два числа a — количество т-образных фигур и b — количество крестиков, которые находятся в одном из пазлов. При этом в наборе всегда есть еще четыре уголка. Известно, что этот комплект позволяет собрать прямоугольную решетку размера $n \times m$, где $1 \leqslant n,\ m \leqslant 10^9$.

Формат выходных данных

Требуется по числам a и b найти размеры исходной решетки n и m. Будем всегда считать, что $n \leqslant m$, то есть нужно вывести в одну строку через пробел два числа, первое из которых не превосходит второго, и вместе они задают размеры загаданной решетки.

Примеры

Пример №1

Стандартный ввод
16 15
Стандартный вывод
4 6

Пример №2

Стандартный ввод
0 0
Стандартный вывод
1 1

Комментарий

Задачу можно решить либо бинарным поиском, либо при помощи квадратного уравнения.

Решение

Ниже представлено решение на языке C++ при помощи бинпоиска.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int a, b;
    cin >> a >> b;
    int L = 0, R = a / 4 + 1;
    while(R - L > 1){
        int M = (R + L) / 2;
        int D = a / 2 - M;
        if(M * D <= b){
            L = M;
        }
        else{
            R = M;
        }
    }
    cout << L + 1 << ' ' << a / 2 - L + 1 << endl;
}
```

Задача 2.2.3.3. Восемь пирогов и одна свечка (20 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Мечта Карлсона наконец-то сбылась! Мама Малыша испекла восемь пирогов прямоугольной формы и в один из них воткнула свечку. После того как Карлсон съел семь пирогов, он решил-таки поделиться кусочком оставшегося восьмого пирога с Малышом. Но, будучи в хорошем настроении, он вынул из пирога свечу и предложил ему решить задачку.

«Так как я самый щедрый Карлсон в мире, то делить оставшийся пирог будешь ты. Но учти, ты должен разрезать пирог одним прямым разрезом так, чтобы линия прошла через один из углов и точку, где стояла свечка. После этого я выберу себе один из двух кусочков, а оставшийся, так и быть, достанется тебе».

Малыш не против этого замысла, однако считает, что разрезать пирог нужно как можно более справедливо, то есть так, чтобы разница между меньшим и большим кусками была как можно меньше. Подскажите Малышу, какой минимальной разницы между площадями кусков он сможет добиться.

Формат входных данных

В первой строке находятся два числа n и m через пробел — размеры прямоугольного пирога. Пирог размещен на координатной плоскости так, что его левый нижний угол находится в точке $(0, 0)$, а правый верхний — в точке (n, m), где $2 \leqslant n,\ m \leqslant 1\,000$.

Во второй строке находятся два числа x и y через пробел — координаты свечки, где $1 \leqslant x \leqslant n - 1$, $1 \leqslant y \leqslant m - 1$, то есть свечка находится строго внутри пирога.

Формат выходных данных

Вывести одно вещественное число с точностью не менее трех знаков после десятичной точки — минимальную разницу между площадями двух получающихся после разрезания кусков, которую сможет получить Малыш.

Примеры

Пример №1

Стандартный ввод
8 5 7 2
Стандартный вывод
12.571

Пример №2

Стандартный ввод
2 2 1 1
Стандартный вывод
0.000

Примечания

На рис. 2.2.5 представлены четыре варианта разделения пирога для первого примера из условия. Можно видеть, что самый близкий к справедливому способ разделения связан с разрезом из левого верхнего угла. Площадь треугольника в этом случае будет равна $96 / 7$, площадь четырехугольника равна $184 / 7$, и разница равна $88 / 7$, что при округлении до трех знаков равно 12,571.

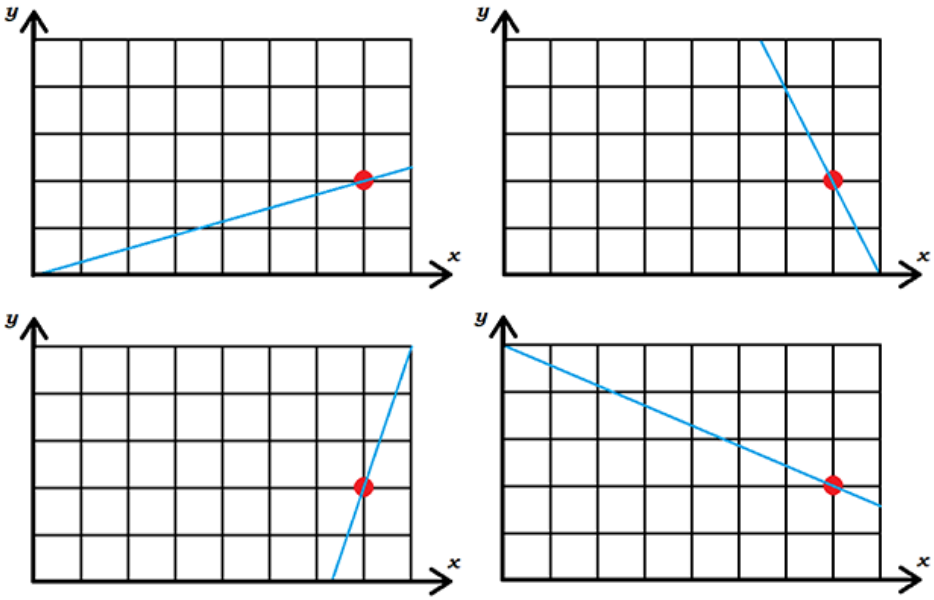


Рис. 2.2.5

Комментарий

Геометрия: для каждого из четырех случаев аккуратно находим катеты прямоугольного треугольника при помощи пропорции, затем находим площадь этого треугольника и, вычитая из всего прямоугольника эту площадь, находим площадь второго куска. Далее выбираем наиболее оптимальное отношение площадей.

Решение

Ниже представлено решение на языке C++.

C++

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
const int INF = 1e18;
double katy(double x, double y, double n){
    return n * y / x;
}
double n, m, x, y;
double ans = INF;
double k1, k2;
void upd(){
    if(k1 < m){
        double st =k1 * n / 2;
        ans = min(ans, n * m - 2 * st);
    }
    else{
        double st =k2 * m / 2;
        ans = min(ans, n * m - 2 * st);
    }}
signed main(){
    cin >> n >> m >> x >> y;
    k1 = katy(x, y, n);
    k2 = katy(y, x, m);
    upd();
    k1 = katy(n - x, y, n);
    k2 = katy(y, n - x, m);
    upd();
    k1 = katy(x, m - y, n);
    k2 = katy(m - y, x, m);
    upd();
    k1 = katy(n - x, m - y, n);
    k2 = katy(m - y, n - x, m);
    upd();
    cout.precision(3);
    cout << fixed << ans<< endl;
}
```

Задача 2.2.3.4. Плетенка (25 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

У Маши есть n полосок бумаги. i-я полоска имеет ширину 1 и длину a_i. Маша разделит эти полоски на две части и покрасит некоторые в желтый, а оставшиеся — в зеленый цвет. Она сама выберет, какие полоски как покрасить. Далее она хочет из этих полосок сплести максимально большую плетенку. Она расположит полоски одного цвета в некотором порядке горизонтально, а полоски другого цвета в некотором порядке вертикально. После этого она переплетет горизонтальные и вертикальные полоски так, что они будут чередоваться то сверху, то снизу, образуя в местах пересечения шахматную раскраску. Наконец, она обрежет выступающие края полосок так, что останется прямоугольная плетенка с ровными краями. Каждая клетка полученной плетенки должна иметь два слоя.

Маша хочет сплести максимально большую по площади прямоугольную плетенку. Подскажите ей, плетенку какой площади она сможет сделать. Заметим, что она может при создании плетенки использовать не все имеющиеся у нее полоски.

Формат входных данных

В первой строке на вход подается число n — количество полосок бумаги у Маши, где $2 \leqslant n \leqslant 2 \cdot 10^5$. Во второй строке через пробел заданы n целых чисел a_i через пробел — длины полосок, где $1 \leqslant a_i \leqslant 10^9$.

Формат выходных данных

Вывести одно число — площадь прямоугольника, форму которого может иметь самая большая плетенка Маши.

Примеры

Пример №1

Стандартный ввод
8 3 6 5 4 4 5 5 2

Стандартный вывод
12

Примечания

На рис. 2.2.6 представлен один из вариантов получения самой большой плетенки для полосок из примера. Синим обозначена граница полученной максимальной плетенки. Ее размер 3×4, и ее площадь 12. При ее создании Маша не должна использовать полоску номер 8, по этой причине неважно, как она раскрашена.

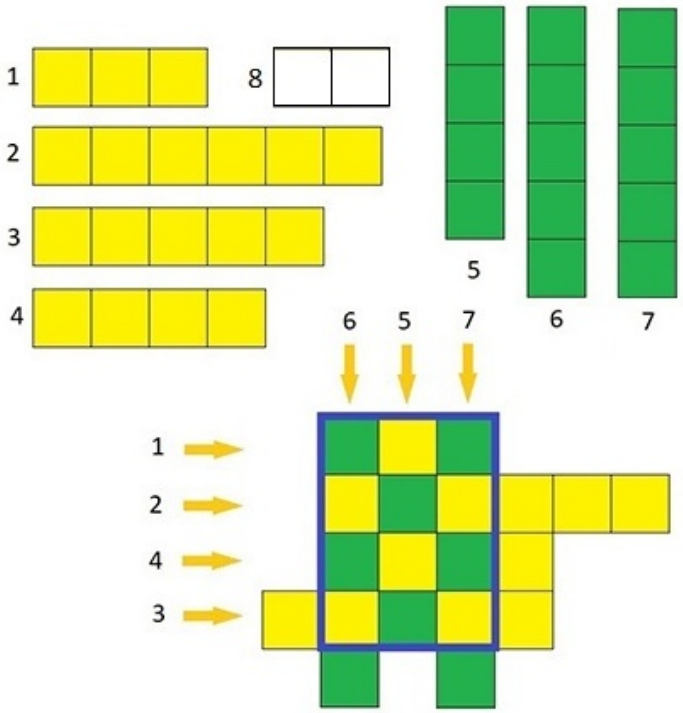


Рис. 2.2.6

Решение

Ниже представлено решение на языке C++.

C++

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int n;
    cin >> n;
    deque<int> v(n);
    for(int i = 0; i < n; i++){
        cin >> v[i];
    }
    sort(v.begin(), v.end());
    int ans = 0;
    int cnth = 0, minh;
    while(1){
        if(v.size() == 0){
            break;
        }
        cnth++;
        minh = v.back();
        v.pop_back();
        while(v.size() > 0 && v[0] < cnth){
            v.pop_front();
        }
        ans = max(ans, cnth * min(minh, (int)v.size()));
    }
    cout << ans << endl;
}
```

Задача 2.2.3.5. Английский в игровой форме (30 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 3 с.

Ограничение по памяти: 64 Мбайт.

Условие

Маша и Витя запоминают слова английского языка в оригинальной игровой форме. За день им нужно выучить n слов, где $20 \leqslant n \leqslant 100$, каждое из которых имеет длину от 5 до 8 символов. Маша выбирает из этого набора наугад несколько попарно различных слов (также от 5 до 8) и собирает их в одну строку без пробелов. Далее она переставляет буквы в этой строке так, что слова оказываются полностью перепутанными, и дает эту строку Вите. Теперь Витя должен восстановить все слова, которые выбрала Маша.

Но у Вити плохо получается, а Маша уже забыла, какие слова она выбрала. Нужно им помочь — написать программу, которая восстановит слова, выбранные Машей.

Формат входных данных

В первой строке находится строка, которую Маша предложила Вите. Во второй строке содержится число n — количество слов, которые нужно выучить детям, $20 \leqslant \leqslant n \leqslant 100$.

В следующих n строках содержатся эти слова по одному в строке. Все слова в этом наборе различны. Слова отсортированы в лексикографическом (алфавитном) порядке. Все слова состоят из маленьких букв от a до z. Обратите внимание, что в тестах к этой задаче все заданные слова реально существуют в английском языке и случайным образом выбраны из словаря.

Гарантируется, что длина каждого слова из предложенного набора (словаря) в пределах от 5 до 8, строка, которую получила Маша, может быть получена путем перестановки букв некоторых различных слов из предложенного словаря, причем, набор выбранных Машей слов определяется по ней однозначно. Количество слов, из которых составлена Машина строка, находится в пределах от 5 до 8.

Формат выходных данных

Вывести все слова, выбранные Машей, в алфавитном порядке по одному в строке.

Примеры

Пример №1

Стандартный ввод
stirbaexsudueoeidgomttcrnrwlunapntetacwri 24 bridge cranky document drawing farmer fighter figurine gravy havoc minimum reactant reply republic sonata soprano split subset tailor texture tomorrow trout vicinity wrist writer

Стандартный вывод
document drawing republic sonata texture wrist

Комментарий

В случае, выделенном в условии (слова являются случайными, взятыми из английского словаря), задача решается рекурсией с перебором вариантов.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
string frs;
int n;
vector<string> dict;
vector<int> msk(26, 0);
int cnt = 0;
vector<vector<int> > amsk;
vector<string> ans;
bool bigok = 0;
void p(int pos){
   if(!bigok){
        if(cnt == 0){
            sort(ans.begin(), ans.end());
            bigok = 1;
            return;
        }
        for(int i = pos; i < n; i++){
            string ts = dict[i];
            bool ok = 1;
            for(int j = 0; j < 26; j++){
                if(amsk[i][j] > msk[j]){
                    ok = 0;
                }
            }
            if(ok){
                ans.push_back(ts);
                for(int j = 0; j < 26; j++){
                    msk[j] -= amsk[i][j];
                    cnt -= amsk[i][j];
                }
                p(i + 1);
                if(!bigok){
                for(int j = 0; j < 26; j++){
                    msk[j] += amsk[i][j];
                    cnt += amsk[i][j];
                }
                ans.pop_back();
                }
            }
        }
   }
}
signed main(){
    cin >> frs;
    cin >> n;
    amsk.resize(n, vector<int>(26, 0));

    string ts;
    for(int i = 0; i < n; i++){
        cin >> ts;
        dict.push_back(ts);
    }
    for(int i = 0; i < n; i++){
        for(auto el : dict[i]){
            amsk[i][el - 'a']++;
        }
    }
```

```
60      for(auto el : frs){
61          msk[el - 'a']++;
62          cnt++;
63      }
64      p(0);
65      for(auto el : ans){
66          cout << el << endl;
67      }
68  }
```

2.2.4. Четвертая волна. Задачи 8–11 класса

Задачи четвертой волны предметного тура по информатике открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63457/enter/.

Задача 2.2.4.1. Квадратный флаг (10 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Одному портному заказали сделать одноцветный флаг. Особенность этого флага в том, что он должен быть квадратным. У портного есть два прямоугольных куска ткани заданного цвета. Один из них имеет размеры $a \times b$, другой — $c \times d$. Так как клиент будет платить пропорционально площади изготовленного флага, портной хочет сначала сшить имеющиеся у него прямоугольные куски, соединив их двумя какими-то сторонами, а затем из полученного полотна вырезать и сделать флаг с максимально большой стороной. Определить сторону получившегося у него флага.

Формат входных данных

На вход подаются две строки. В первой строке находятся размеры первого прямоугольника — целые числа a, b через пробел, во второй — размеры второго прямоугольника, также целые числа c, d через пробел, где $1 \leqslant a, b, c, d \leqslant 10^9$.

Формат выходных данных

Вывести одно число — сторону самого большого квадрата, который можно получить по условию задачи.

Примеры

Пример №1

Стандартный ввод
2 4 3 6

Стандартный вывод
4

Пример №2

Стандартный ввод
2 2 3 6

Стандартный вывод
3

Примечания

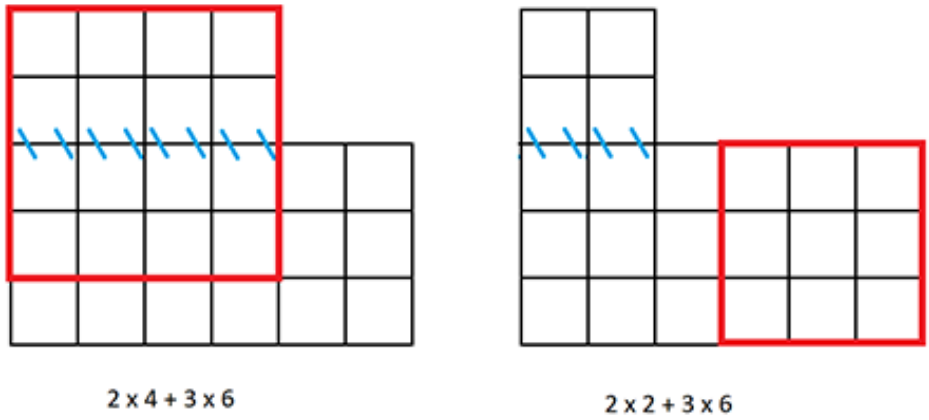


Рис. 2.2.7

На рис. 2.2.7 представлены иллюстрации для тестов из условия. Синими штрихами обозначено место сшивки двух кусков. Красный квадрат выделяет один из вариантов вырезания максимального квадрата.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int a, b, c, d;
    cin >> a >> b >> c >> d;
    int ans = max(min(a, b), min(c, d));
    int p1 = min(a + c, min(b, d));
    int p2 = min(a + d, min(b, c));
    int p3 = min(b + c, min(a, d));
    int p4 = min(b + d, min(a, c));
    ans = max({ans, p1, p2, p3, p4});
    cout << ans << endl;
}
```

Задача 2.2.4.2. Потерянная ДНК (15 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

В данной задаче будем упрощенно считать, что ДНК представляется строкой длины от 10 до 100, состоящей из букв A, C, G, T.

Пусть даны две ДНК D_1 и D_2 одной и той же длины n. Выберем некоторое произвольное число i от 1 до $n-1$ и поменяем местами префиксы (начала) этих ДНК длины i. Будем говорить, что полученные новые две строки образованы путем скрещивания двух исходных по префиксу длины i.

Например, пусть $D_1 =$ **AACGGTAGGT**, а $D_2 =$ TCCCGGAACA. Выберем $i = 4$ и поменяем местами префиксы длины 4. Получим две новые ДНК, одна из которых будет иметь вид **AACG**GGAACA, а вторая — TCCC**GTAGGT**. Для наглядности были выделены части первой из них.

Полученные новые ДНК снова могут быть скрещены по любому префиксу длины от 1 до $n-1$.

Теперь можно рассмотреть популяцию из нескольких ДНК. Выберем из них две, произведем их скрещивание по префиксу какой-либо длины и поместим две новые ДНК в исходную популяцию. В данной задаче будем считать, что количество ДНК не увеличивается, то есть старые две ДНК заменяются на новые две ДНК.

Дана исходная популяция из m ДНК, каждая имеет одну и ту же длину n. После некоторого количества попарных скрещиваний была получена новая популяция. Но при итоговой обработке данных сведения об одной ДНК из новой популяции были потеряны. Задача состоит в отыскании этой потерянной ДНК по оставшимся $m-1$ ДНК из новой популяции.

Формат входных данных

В первой строке через пробел даны два числа n — длина ДНК и m — количество ДНК в исходной популяции, где $10 \leqslant n \leqslant 100$, $2 \leqslant m \leqslant 100$.

В следующих m строках содержится описание исходной популяции ДНК, каждая задается строкой длины n, состоящей из символов A, C, G и T.

Далее следует разделяющая строка, содержащая n символов «$-$».

Далее следует еще $m - 1$ строк, описывающих новую (заключительную) популяцию без одной ДНК.

Гарантируется, что данные верны, то есть $m - 1$ последняя ДНК является некоторой новой популяцией ровно без одной ДНК, полученной из исходной популяции, заданной в m первых строках.

Формат выходных данных

Вывести недостающую утерянную ДНК.

Примеры

Пример №1

Стандартный ввод
10 2 AACGGTAGGT TCCCGGAACA ---------- TCCCGTAGGT

Стандартный вывод
AACGGGAACA

Пример №2

Стандартный ввод
10 4 AACCGGTTAA ACGTACGTAC AAACCCGGGT CATTACTGGA ---------- AAGCGCTTAA CCACACGTGC AACTAGGGGT

Стандартный вывод
AATTCCTGAA

Комментарий

Для каждой позиции нужно найти недостающую букву из первого набора ДНК. Для этого удобнее всего использовать функцию `xor`.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int n, m;
    cin >> n >> m;
    vector<string> v1(m);
    for(int i = 0; i < m; i++){
        cin >> v1[i];
    }
    string d;
    cin >> d;
    vector<string> v2(m - 1);
    for(int i = 0; i < m - 1; i++){
        cin >> v2[i];
    }
    for(int j = 0; j < n; j++){
        int ss = 0;
        for(int i = 0; i < m; i++){
            ss ^= (int)(v1[i][j]);
        }
        for(int i = 0; i < m - 1; i++){
            ss ^= (int)(v2[i][j]);
        }
        cout << (char)(ss);
    }
    cout << endl;
}
```

Задача 2.2.4.3. Утомленные туристы (20 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Рассмотрим следующий вариант известной задачи на перемещение по туннелю группы из четырех человек. В общем виде она выглядит так: четыре туриста хотят пройти по темному туннелю. Имеется один фонарик. По туннелю можно перемещаться либо вдвоем, либо по одному, при этом у тех, кто движется в туннеле,

должен быть фонарик в руках. По этой причине движение должно быть следующим: двое переходят туда, один возвращается обратно и приносит фонарик тем, кто еще не перешел. После этого указанный маневр повторяется снова.

У каждого участника своя скорость движения в туннеле. Пусть участники проходят туннель за A, B, C и D мин. Если идут двое, то они движутся со скоростью того, кто идет медленнее. Требуется по заданным временам прохождения туннеля каждого из участников перевести их максимально быстро через туннель.

Немного усложним данную задачу. Введем фактор усталости. А именно, любой участник, пройдя по туннелю, устает и в следующий раз идет уже медленнее. После каждого прохождения туннеля время прохождения любого участника увеличивается на E мин. Например, если участник до начала движения проходит туннель за 1 мин, а показатель усталости E равен 3 мин, то первый раз участник пройдет туннель за 1 мин, второй раз — за 4 мин, третий раз — за 7 мин и т. д.

По заданным A, B, C, D и E узнать, за какое минимальное время можно провести всю группу через туннель согласно указанным правилам.

Формат входных данных

На вход подаются пять чисел. В первой строке через пробел четыре числа A, B, C и D — время прохождения туннеля каждым из четырех участников до того, как они начали движение. Во второй строке содержится число E — величина, на которую увеличивается время прохождения туннеля каждым участником после каждого перемещения. При этом $1 \leqslant A, B, C, D \leqslant 1\,000$, $0 \leqslant E \leqslant 1\,000$.

Формат выходных данных

Вывести одно число — минимальное время прохождения туннеля всей группой.

Примеры

Пример №1

Стандартный ввод
8 9 10 1 3
Стандартный вывод
44

Пример №2

Стандартный ввод
8 9 10 1 0
Стандартный вывод
29

Примечания

В первом примере при прохождении туннеля каждый турист устает и движется медленнее на 3 мин. Покажем, как перевести группу при этом за 44 мин.

Каждую ситуацию будем обозначать следующим образом: слева от двоеточия находятся туристы, которые стоят в начале туннеля, а справа — те, что стоят в конце туннеля. Туриста будем обозначать при помощи числа, соответствующего его текущему времени прохождения туннеля.

Тогда исходная ситуация имеет вид 1, 8, 9, 10 :.

Сначала идут туристы 1 и 8, каждый после перехода устает на 3 мин, получим ситуацию 9, 10 : 4, 11, затрачено 8 мин.

Обратно возвращается турист 4, он устает еще на 3 мин. Ситуация становится 7, 9, 10 : 11, затрачено $8 + 4 = 12$ мин.

Теперь идут туристы 7 и 9, получится ситуация 10 : 10, 11, 12, затрачено $8 + 4 + 9 = 21$ мин.

Возвращается турист 10, получится 10, 13 : 11, 12, затрачено $8 + 4 + 9 + 10 = 31$ мин.

Наконец, оставшиеся двое туристов 10 и 13 за 13 мин переходят туннель, итого затрачено $8 + 4 + 9 + 10 + 13 = 44$ мин.

Комментарий

Задача решается рекурсивным перебором всех вариантов прохождения.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
const int INF = 1e18;
vector<int> v(4);
int e, ans = INF;
void p(vector<int> &vl, vector<int> &vr, int tv){
    if(vl.size() == 2){
        ans = min(ans, tv + *max_element(vl.begin(), vl.end()));
        return;
    }
    for(int i = 0; i < vl.size() - 1; i++){
        for(int j = i + 1; j < vl.size(); j++){
            vector<int> vl1;
            for(int k = 0; k < vl.size(); k++){
                if(k != i && k != j){
                    vl1.push_back(vl[k]);
                }
            }
            vector<int> vr1 = vr;
```

```
21                  vr1.push_back(vl[i] + e);
22                  vr1.push_back(vl[j] + e);
23                  int tmp = max(vl[i], vl[j]);
24                  sort(vr1.rbegin(), vr1.rend());
25                  vl1.push_back(vr1.back() + e);
26                  vr1.pop_back();
27                  p(vl1, vr1, tv + tmp + vl1.back() - e);
28              }
29          }
30      }
31      signed main(){
32          for(int i = 0; i < 4; i++){
33              cin >> v[i];
34          }
35          sort(v.begin(), v.end());
36          cin >> e;
37          vector<int> vl = v, vr;
38          p(vl, vr, 0);
39          cout << ans;
40      }
```

Задача 2.2.4.4. Проектируем мост (25 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

При постройке моста используются два типа пролетов: П-образные (они прочные, но дорогие) и Т-образные (они дешевле, но менее надежные). Мост должен начинаться и заканчиваться П-образными пролетами. Любой Т-образный пролет должен иметь хотя бы один П-образный пролет в качестве соседнего.

Длина проектируемого моста — n пролетов. Муниципалитет выделил средства на постройку a П-образных и b Т-образных пролетов. При этом $a + b = n$. Требуется выяснить, сколькими способами при этих условиях можно скомпоновать мост. Два способа компоновки моста отличаются, если в одной на некоторой позиции стоит П-образный пролет, а в другой на этой же позиции стоит Т-образный пролет.

Формат входных данных

В одной строке через пробел заданы два числа: a — число П-образных пролетов и b — число Т-образных пролетов, на постройку которых выделены средства, где $2 \leqslant a \leqslant 10^6$, $0 \leqslant b \leqslant 10^6$.

Формат выходных данных

Вывести одно число — количество вариантов компоновки моста. Так как ответ может быть очень большим, требуется вывести остаток от его деления на 1 000 000 007 ($10^9 + 7$).

Примеры

Пример №1

Стандартный ввод
4 3

Стандартный вывод
7

Примечания

Для примера из условия имеется 7 вариантов компоновки моста (пробелы добавлены для лучшего восприятия вариантов):

П Т Т П Т П П

П Т Т П П Т П

П Т П Т Т П П

П Т П П Т Т П

П П Т П Т Т П

П П Т Т П Т П

П Т П Т П Т П

Комментарий

При заданных ограничениях задача решается только при помощи комбинаторики с вычислениями по модулю.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
const int INF = 1e18;
const int MOD = 1e9 + 7;
vector<int> f(2e6 + 1, 1);
```

```
int binpow (int a, int n) {
        int res = 1;
        while (n > 0) {
                if (n % 2 == 1)
                        (res *= a) %= MOD;
                (a *= a) %= MOD;
                n /= 2;
        }
        return res;
}

int bc(int n, int k){
    int res = f[n];
    int p1 = binpow(f[k], MOD - 2);
    int p2 = binpow(f[n - k], MOD - 2);
    (res *= p1) %= MOD;
    (res *= p2) %= MOD;
    return res;
}
signed main(){
    for(int i = 1; i <= 2e6; i++){
        f[i] = (f[i - 1] * i) % MOD;
    }
    int a, b;
    int ans = 0;
    cin >> a >> b;
    a--;
    for(int i = 0; i < a + 1; i++){
        if(2 * i <= b){
            int d = bc(a, i);
            if(b - 2 * i <= a - i){
                (d *= bc(a - i, b - 2 * i) ) %= MOD;
                (ans += d) %= MOD;
            }
        }
    }
    cout << ans << endl;
}
```

Задача 2.2.4.5. Джентльмены на прогулке (30 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 8 с.

Ограничение по памяти: 64 Мбайт.

Условие

По прямому участку улицы, которую будем считать отрезком AB длины d, прогуливаются n джентльменов. i-й джентльмен движется со скоростью v_i. Скорости всех джентльменов попарно различны. Дойдя до любого конца улицы, каждый джентльмен поворачивает и идет в обратную сторону.

При каждой встрече два джентльмена приветствуют друг друга, приподнимая

головной убор. Приветствие происходит и в том случае, когда один джентльмен обгоняет другого. Если два джентльмена встречаются в момент их одновременного поворота, то происходит два приветствия: одно до поворота, другое — после поворота. Если происходит одновременная встреча трех и более джентльменов, то они приветствуют друг друга попарно, то есть каждый каждого. Допустим, если одновременно встретились четыре джентльмена где-то посреди улицы, произойдет шесть попарных приветствий. Если же эти четыре джентльмена встретились в момент их одновременного поворота, произойдет уже двенадцать приветствий.

В этой задаче считаем, что все действия происходят без остановок, то есть и повороты и приветствия происходят мгновенно. Джентльмены одновременно начинают свою прогулку из точки A в момент 0. В этот момент они уже производят свои первые попарные приветствия, то есть в момент 0 уже произведено $n \cdot (n-1)/2$ приветствий. Момент старта не считается моментом поворота, то есть на старте число приветствий не удваивается. Джентльмены гуляют достаточно долго, чтобы произошло любое заданное количество приветствий.

Требуется найти момент, в который было произведено k-е по порядку приветствие.

Формат входных данных

В первой строке ввода через пробел содержится два целых числа: d — длина отрезка AB и n — количество прогуливающихся джентльменов, где $1 \leqslant d \leqslant 200$, $2 \leqslant n \leqslant 2\,000$.

Во второй строке находятся n целых чисел v_i через пробел — скорости каждого джентльмена, где $1 \leqslant v_i \leqslant 2\,000$. Гарантируется, что все скорости попарно различны. Скорости даны в порядке возрастания, то есть $v_1 < v_2 < \ldots < v_n$.

В третьей строке содержится одно целое число k — номер требуемого приветствия, для которого нужно найти момент, когда оно произойдет, где $1 \leqslant k \leqslant 10^9$.

Формат выходных данных

Вывести одно вещественное число — время, когда произойдет k-е по порядку приветствие. Ответ вывести с точностью не менее двух знаков после десятичной точки.

Примеры

Пример №1

Стандартный ввод
5 4 2 5 8 10 6

Стандартный вывод
0.000

Пример №2

Стандартный ввод
5 4 2 5 8 10 7
Стандартный вывод
0.556

Пример №3

Стандартный ввод
5 4 2 5 8 10 11
Стандартный вывод
1.000

Пример №4

Стандартный ввод
5 4 2 5 8 10 15
Стандартный вывод
1.429

Пример №5

Стандартный ввод
5 4 2 5 8 10 17
Стандартный вывод
1.667

Пример №6

Стандартный ввод
5 4 2 5 8 10 19
Стандартный вывод
1.667

Пример №7

Стандартный ввод
5 4 2 5 8 10 21
Стандартный вывод
2.000

Примечания

На рис. 2.2.8 приведено положение джентльменов из примеров в моменты времени 0, 1 и 2. Джентльмены обозначены своими скоростями. Стрелками обозначены направления их движения в соответствующий момент. Перечислим и пронумеруем в порядке возрастания моменты попарных приветствий этих джентльменов до момента времени 2 включительно. Если два и более приветствия происходят одновременно, неважно какое из них конкретно имеет номер k, главное, что они происходят в один и тот же определенный момент времени.

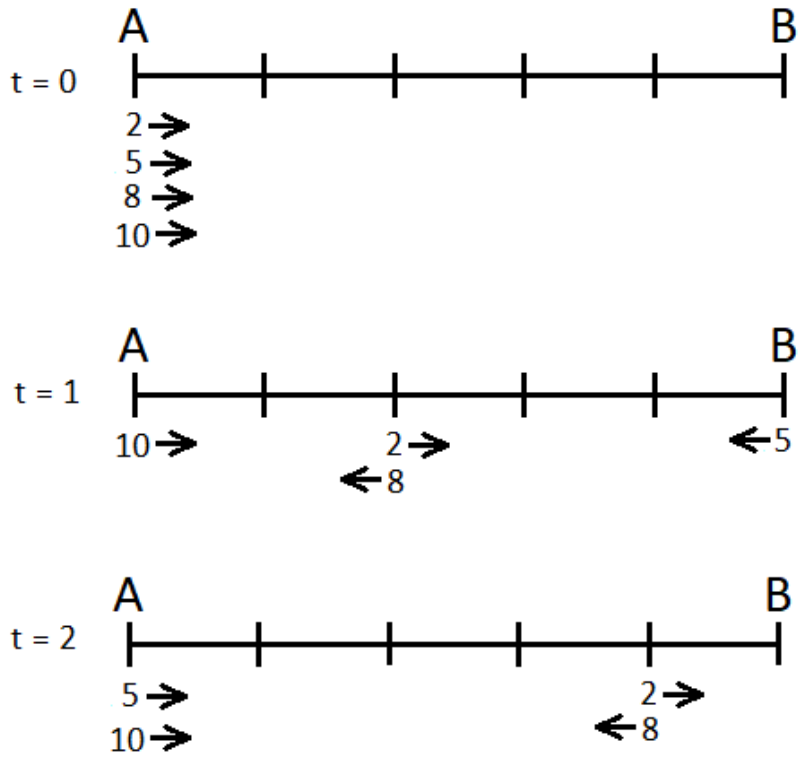


Рис. 2.2.8

1. 2 и 5 приветствуют друг друга в момент 0 (изображено на рис. 2.2.8).
2. 2 и 8 приветствуют друг друга в момент 0 (изображено на рис. 2.2.8).
3. 2 и 10 приветствуют друг друга в момент 0 (изображено на рис. 2.2.8).
4. 5 и 8 приветствуют друг друга в момент 0 (изображено на рис. 2.2.8).
5. 5 и 10 приветствуют друг друга в момент 0 (изображено на рис. 2.2.8).

6. 8 и 10 приветствуют друг друга в момент 0 (изображено на рис. 2.2.8).
7. 8 и 10 приветствуют друг друга в момент 0.556.
8. 5 и 10 приветствуют друг друга в момент 0.667.
9. 5 и 8 приветствуют друг друга в момент 0.769.
10. 2 и 10 приветствуют друг друга в момент 0.833.
11. 2 и 8 приветствуют друг друга в момент 1.000 (изображено на рис. 2.2.8).
12. 8 и 10 приветствуют друг друга в момент 1.111.
13. 2 и 10 приветствуют друг друга в момент 1.250.
14. 5 и 10 приветствуют друг друга в момент 1.333.
15. 2 и 5 приветствуют друг друга в момент 1.429.
16. 5 и 8 приветствуют друг друга в момент 1.538.
17. 2 и 8 приветствуют друг друга в момент 1.667.
18. 2 и 10 приветствуют друг друга в момент 1.667.
19. 8 и 10 приветствуют друг друга в момент 1.667 (в момент 1.667 встретятся одновременно три джентльмена 2, 8 и 10).
20. 2 и 8 приветствуют друг друга в момент 2.000 (изображено на рис. 2.2.8).
21. 5 и 10 приветствуют друг друга в момент 2.000 (до поворота).
22. 5 и 10 приветствуют друг друга в момент 2.000 (после поворота, изображено на рис. 2.2.8).

Комментарий

Задача решается при помощи бинпоиска с квадратичным нахождением ответа в каждой его итерации.

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
const double EPS = 1e-7;
double x(double M, int V, int d){
    double dst = V * M;
    int cnt = floor((dst + EPS) / d);
    double pin = dst - cnt * d;
    if(cnt % 2 == 0){
        return pin;
    }
    else{
        return d - pin;
    }
}
int F(double M, vector<int> &v, int d){
    int res = 0;
    for(int i = 0; i < v.size(); i++){
        double dst = v[i] * M;
```

```
        int cnt = floor((dst + EPS) / d);
        res += cnt * i;
        double tx = x(M, v[i], d);
        for(int j = 0; j < i; j++){
            double txj = x(M, v[j], d);
            if(cnt % 2 == 0){
                res += txj <= tx + EPS;
            }
            else{
                res += txj >= tx - EPS;
            }
        }
    }
    return res;
}
signed main(){
    int d, n;
    cin >> d >> n;
    vector<int> v(n);
    for(int i = 0; i < n; i++){
        cin >> v[i];
    }
    int k;
    cin >> k;
    double L = 0, R = 1;
    while(F(R, v, d) <= k){
        R *= 2;
    }
    R *= 2;
    while(R - L > 1e-4){
        double M = (R + L) / 2.0;
        if(F(M, v, d) < k){
            L = M;
        }
        else{
            R = M;
        }
    }
    cout.precision(10);
    cout << fixed << L << endl;
}
```

2.3. Предметный тур. Математика

2.3.1. Первая волна. Задачи 8–9 класса

Задачи первой волны предметного тура по математике за 8–9 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63459/enter/.

Задача 2.3.1.1. (15 баллов)

Тема: арифметика.

Условие

Оля в каждую клетку таблицы 3×3 записала по некоторому числу и с удивлением заметила, что сумма чисел в каждой строке и в каждом столбце таблицы равна 23. Внимательный же одноклассник Витя к ее размышлениям добавил информацию, что сумма чисел в каждом получившемся квадрате 2×2 равна 32. Какое число Оля записала в центральную клетку таблицы?

Решение

Проанализируем исходную таблицу и увидим, что при построении всех возможных квадратов 2×2:

- числа, стоящие в угловых клетках исходной таблицы, входят по одному разу;
- числа, стоящие во второй строке и во втором столбце — по два раза;
- центральное число — четыре раза.

Тогда если найдем сумму чисел во всех квадратах 2×2 и из нее вычтем сумму чисел всей таблицы, а также сумму чисел, стоящих во втором столбце и второй строке, то найдем центральное число, то есть $32 \cdot 4 - 23 \cdot 3 - 23 \cdot 2 = 13$.

Ответ: 13.

Задача 2.3.1.2. (15 баллов)

Тема: комбинаторика.

Условие

Нечетное восьмизначное число назовем «интересным», если оно состоит из простых цифр и одинаковые цифры не стоят рядом. Сколько существует таких «интересных чисел»?

Решение

Простые цифры — это 2, 3, 5 и 7. Тогда так как «интересное» число должно быть нечетным, то в разряде его единиц может стоять только 3, 5 или 7, то есть три варианта. В разряде десятков также может стоять только три варианта, т. к. одинаковые цифры не могут стоять рядом, и т. д. Таким образом, общее количество «интересных» чисел равно $3^8 = 6561$.

Ответ: 6561.

Задача 2.3.1.3. (20 баллов)

Тема: планиметрия.

Условие

В остроугольном треугольнике ABC провели высоты AA_1 и CC_1. Точки E и F — середины отрезков AC и A_1C_1 соответственно.

Найдите длину отрезка EF, если известно, что $AC = 30$ и $A_1C_1 = 24$.

Решение

В прямоугольном треугольнике AC_1C с гипотенузой AC: $C_1E = \frac{1}{2}AC = 15$. Аналогично в треугольнике A_1C: $A_1E = \frac{1}{2}AC = 15$.

Таким образом, треугольник A_1C_1E является равнобедренным, и его медиана EF является также и высотой.

Тогда по теореме Пифагора: $EF^2 = A_1E^2 - A_1F^2 = 15^2 - 12^2 = 81$, $EF = 9$.

Ответ: 9.

Задача 2.3.1.4. (25 баллов)

Темы: уравнения, формулы сокращенного умножения.

Условие

Найдите значение выражения $x + y + 3z$, если известно, что числа x, y, z удовлетворяют равенству:

$$5x^2 + 4y^2 + 9z^2 + 12z + 13 = 4xy + 12x.$$

Решение

Преобразуем равенство следующим образом:

$$(x^2 - 4xy + 4y^2) + (4x^2 - 12x + 9) + (9z^2 + 12z + 4) = 0,$$

то есть

$$(x-2y)^2+(2x-3)^2+(3z+2)^2=0.$$

Данное равенство будет выполняться при условии, что каждое слагаемое равно 0.

Отсюда получаем систему

$$\begin{cases} x-2y=0, \\ 2x-3=0, \\ 3z+2=0, \end{cases}$$

единственным решением которой будет

$$x=\frac{3}{2};\ y=\frac{3}{4};\ z=-\frac{2}{3}.$$

Тогда

$$x+y+3z=\frac{3}{2}+\frac{3}{4}+3\cdot\left(-\frac{2}{3}\right)=\frac{1}{4}=0{,}25.$$

Ответ: 0,25.

Задача 2.3.1.5. (25 баллов)

Тема: теория вероятностей.

Условие

Шестизначное число будем называть «замечательным», если оно составлено из цифр 1, 2, 3, 4, 5, 6 (каждая цифра используется в числе по одному разу) и кратно 12. Какая вероятность, что сгенерированное компьютером шестизначное число будет «замечательным»?

Ответ выразите в долях и округлите его до четвертого знака после запятой.

Решение

Для того чтобы «замечательное» число делилось на 12, оно должно делиться на три и на четыре. Заметим, что все рассматриваемые числа кратны трем, так как сумма их цифр равна 21.

Для того же чтобы число было кратно четырем, необходимо, чтобы две его последние цифры образовывали число, кратное четырем. В нашем случае это могут быть варианты: 12, 16, 24, 32, 36, 52, 56, 64, всего их восемь. К каждому из них нужно приписать впереди четырехзначное число, составленное из остальных четырех цифр, таких чисел $4!=24$. Значит, всего «интересных» чисел $24\cdot 8=192$.

Всего же шестизначных чисел $9\cdot 10^5=900\,000$.

Тогда вероятность, что сгенерированное компьютером число будет являться «замечательным», будет равна $\dfrac{192}{900\,000}\approx 0{,}0002$.

Ответ: 0,0002.

2.3.2. Первая волна. Задачи 10–11 класса

Задачи первой волны предметного тура по математике за 10–11 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63476/enter/.

Задача 2.3.2.1. (10 баллов)

Темы: комбинаторика, десятичная запись числа, цифры.

Условие

Двузначное число назовем подходящим, если оно состоит из четных цифр, расположенных по возрастанию (например, 26). Сколько существует таких подходящих чисел?

Решение

Число не может начинаться с нуля, так что можно использовать только цифры 2, 4, 6, 8. Выпишем все подходящие: 24, 26, 28, 46, 48, 68.

Ответ: 6.

Задача 2.3.2.2. (15 баллов)

Темы: текстовые задачи, пропорции, составление уравнений.

Условие

На Марсе планируется разместить колонию в 100 тысяч человек. Разные колонисты будут заняты на разных работах и важно, чтобы каждый вид работы выполняли группы из минимального количества человек. Одна из важных задач — обеспечение колонистов сбалансированным питанием. Нормы здорового рациона были рассчитаны таким образом, чтобы обеспечить для каждого человека 350 г картофеля в день. Полный цикл производства картофеля от посадки и до сбора составляет 60 дней, каждые 60 дней часть собранного урожая используется для выращивания нового. В той технологии, которую используют космонавты, с 1 га можно вырастить 250 т картофеля, а для посадки нужно 5 т/га. Специальная обработка почвы позволяет добиться сохранения постоянного уровня урожайности, причем можно засадить и обрабатывать произвольную долю гектара. Чтобы полностью обслуживать один гектар в условиях теплиц на Марсе, требуется труд четырех человек.

Какое минимальное количество человек должны трудиться на выращивании картофеля?

Решение

Один человек за 60 дней по плану должен съедать $60 \cdot 0{,}35 = 21$ кг картофеля. Следовательно, 100 тысяч человек по плану за это время съедят 2 100 000 кг.

С одного гектара получаем 250 т, но при этом из них 2 т нужно использовать для посадки. Это значит, что с каждого гектара люди получат в свой рацион 245 т картофеля. Если разделить количество картофеля, которое съест по плану колония за 60 дней, на количество картофеля, которое попадет к ним с 1 га, то получится, что требуется приблизительно 8,571 га. Так как каждый гектар должны обрабатывать четыре человека, то для обработки 8,571 га потребуется труд 34,286 человек. Это значит, что 34 человек недостаточно, требуется запланировать труд 35 человек.

Ответ: 35.

Задача 2.3.2.3. (20 баллов)

Темы: уравнение параболы, координаты вершины параболы.

Условие

Две параболы с различными вершинами пересекаются таким образом, что первая парабола проходит через вершину второй параболы, а вторая — проходит через вершину первой. Уравнение первой параболы имеет вид $y = x^2$, второй $y = a \cdot x^2 + b \cdot x + c$. Найдите, чему равна величина $10 \cdot a + c$.

Решение

Координаты вершины первой параболы имеют вид $(0; 0)$, следовательно, коэффициент $c = 0$. Координаты вершины второй параболы имеют вид

$$x = -\frac{b}{2a};$$
$$y = -\frac{b^2}{4a}.$$

Тогда, подставив их в уравнение первой параболы, получаем:

$$-\frac{b^2}{4a} = \frac{b^2}{2a}.$$

Отсюда $a = -1$.

Ответ: -10.

Задача 2.3.2.4. (25 баллов)

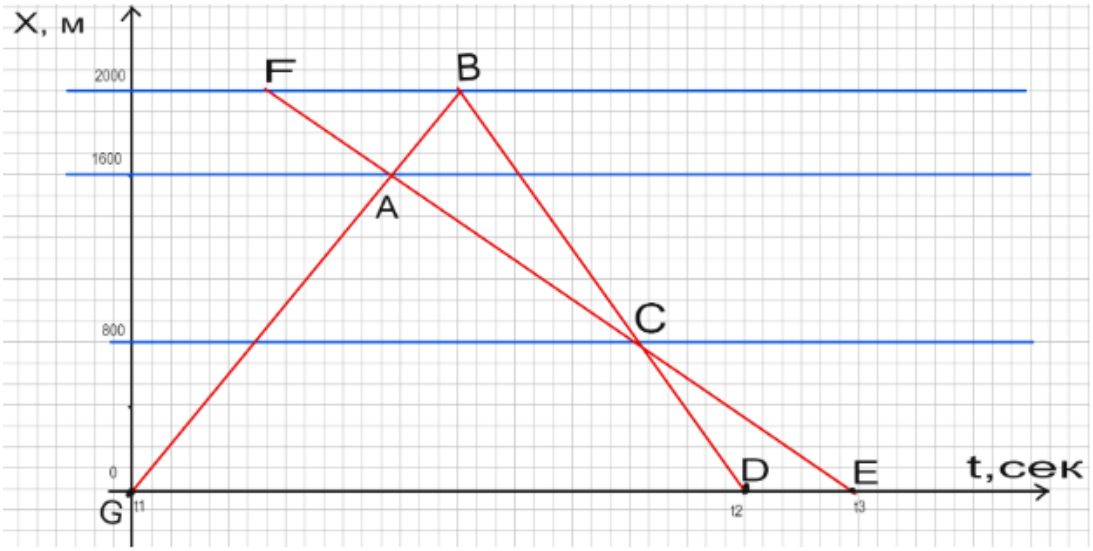
Темы: текстовые задачи и логика, графическое изображение движения, теорема Менелая.

Условие

В 6:00 со дна океана, находящегося на глубине 2 000 м, на поверхность, двигаясь с постоянной скоростью вертикально вверх, начала всплывать подводная лодка. Когда она поднялась до глубины 400 м, капитан заметил, что мимо них вниз плывет глубоководный батискаф. Ему что-то показалось странным. Когда подводная лодка поднялась на поверхность, капитан понял, что на оболочке батискафа были признаки повреждения. Чтобы предотвратить возможную трагедию, в тот же самый момент с подводной лодки вниз спустили спасательный глубоководный аппарат, который спускался с некоторой постоянной скоростью. Когда до дна оставалось 800 м, этот аппарат поравнялся с батискафом. Если бы спасательный аппарат не перехватил батискаф, то спасательный аппарат достиг бы дна к 11:00. Предполагая, что спасательный аппарат все время движения двигался равномерно, определите, в какой момент времени батискаф достиг бы дна, если бы он продолжил движение с той же постоянной скорость. Ответ введите в виде двух целых чисел, записанных подряд — количество часов и количество минут.

Решение

Самое изящное решение получается графическим методом.



Здесь данные из условия задачи можно обозначить следующим образом:

$h_1 = 2\,000 - 400 = 1\,600$, $h(2) = 800$, $H = 2\,000$, $t_1 = 6$, $t_2 = 12$.

По теореме Менелая получаем:

$$\frac{GA}{AB} \cdot \frac{BC}{CD} \cdot \frac{DE}{GE} = 1.$$

Выразим эти отношения из разных пар подобных треугольников:

$$\frac{GA}{AB} = \frac{h_1}{H - h_1};$$
$$\frac{BC}{CD} = \frac{H - h_2}{h_2};$$
$$\frac{DE}{GE} = \frac{t_3 - t_2}{t_3 - t_1};$$
$$\frac{h_1}{H - h_1} \cdot \frac{H - h_2}{h_2} \cdot \frac{t_3 - t_2}{t_3 - t_1} = 1.$$

Подставим числа:

$$\frac{1\,600}{400} \cdot \frac{1\,200}{800} \cdot \frac{t_3 - 11}{t_3 - 6} = 1;$$
$$6 \cdot (t_3 - 11) = t_3 - 6;$$
$$5 \cdot t_3 = 60;$$
$$t_3 = 12 \text{ ч.}$$

Ответ: 12.

Задача 2.3.2.5. (30 баллов)

Условие

Инженер-исследователь работает над созданием новой системы гиперпространственной навигации для космических кораблей, которая потребует меньших вычислительных ресурсов. Часть измерений гиперпространства скрыта от нас и устроена не так, как мы привыкли, а именно — являются дискретными (с конечным количеством позиций), позиции в которых следуют друг за другом циклически. Например, если это измерение, в котором 5 позиций, то их можно занумеровать числами от 0 до 4 так, что космический корабль, при прямолинейном движении вдоль этого измерения, будет пролетать позиции $0 - 1 - 2 - 3 - 4 - 0 - \ldots$ (конечно, корабль в любой момент может изменить направление своего движения на обратное или начать/продолжить изменять позиции и по другим измерениям гиперпространства).

Оказалось, что в гиперпространстве возможна быстрая (но не мгновенная) телепортация: для такого перемещения требуется особая последовательность перемещений в дискретных подпространствах с остановками лишь в выделенные моменты времени. Ранее для хранения таких сложных гипермаршрутов использовалась технология сплошного хранения всех промежуточных опорных точек пути. Однако из-за воздействия агрессивной космической радиации устройства хранения информации часто выходят из строя, что делает сплошное хранение информации очень дорогим, так как требует многократного резервного копирования,

Инженер корабля предложил хранить не сами последовательности позиций, а формулы для их вычисления (что хранить гораздо дешевле и надежнее). В частности, ему удалось запрограммировать движение в одном из измерений с 13 позициями следующим образом: начальное положение обозначается числом 0 и дальнейшие позиции для остановки вычисляются по формуле: x_{n+1} равно остатку от деления $(x_n^5 + 2)$ на 13.

Переход корабля из одной позиции в соседнюю по прямому или обратному ходу занимает 1 единицу времени, которую называют таймом. Корабль, используя эту формулу, прошел полный цикл по остановкам и вернулся в позицию с номером 0.

Какое минимальное количество таймов могло занимать все его движение между остановками в ходе этого цикла?

Решение

Запишем последовательность позиций, в которых останавливается корабль:

$$0 - 2 - 8 - 10 - 6 - 4 - 12 - 1 - 3 - 11 - 9 - 5 - 7 - 0.$$

Между каждыми двумя позициями корабль может двигаться либо прямым ходом, либо обратным. Нужно выбирать кратчайший из двух.

Тогда общая длительность промежутков будет:

$$T = 2 + 6 + 2 + 4 + 2 + 5 + 2 + 2 + 5 + 2 + 4 + 2 + 6 = 44.$$

Ответ: 44.

2.3.3. Вторая волна. Задачи 8–9 класса

Задачи второй волны предметного тура по математике за 8–9 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63460/enter/.

Задача 2.3.3.1. (15 баллов)

Тема: арифметика.

Условие

Первый поезд мимо телеграфного столба проезжает за 9 с, второй поезд мимо этого же столба — за 14 с, а, двигаясь навстречу мимо друг друга, они проезжают за 10 с (с момента, когда поравнялись их начала, и до момента, когда разминулись концы).

Во сколько раз скорость первого поезда больше скорости второго?

Решение

Пусть x м/с — скорость первого поезда, тогда из условия задачи его длина 9 м. Аналогично, если y м/с — скорость второго поезда, то его длина равна $14y$ м.

Зная, что, двигаясь навстречу мимо друг друга, они проезжают за 10 с, составим уравнение:

$$\frac{9x + 14y}{x + y} = 10.$$

Решив это уравнение, получим $x = 4y$. То есть скорость первого поезда в четыре раза больше скорости второго.

Ответ: 4.

Задача 2.3.3.2. (15 баллов)

Тема: комбинаторика.

Условие

Вася и Петя играют в разведчиков и для этого придумали свой язык шифрования, в котором используются только пять символов. При этом все «слова» в их сообщениях непустые, то есть содержат хотя бы один знак, и длиной не более пяти знаков.

Сколько различных «слов» они имеют в своем арсенале, чтобы передавать друг другу информацию?

Решение

«Слова», которые могут составлять Вася и Петя на своем языке, могут состоять из 1, 2, 3, 4 и 5 символов.

Тогда общее количество слов будет равно $5^1 + 5^2 + 5^3 + 5^4 + 5^5 = 3905$.

Ответ: 3905.

Задача 2.3.3.3. (20 баллов)

Тема: геометрия.

Условие

В треугольнике ABC длина биссектрисы AD равна длине отрезка DC и $AC = 2AB$. Найдите $\angle ABC$.

Решение

В равнобедренном треугольнике ADC из точки D проведем медиану DE на сторону AC, которая также будет являться и высотой.

Тогда $AE = \frac{1}{2}AC = AB$. Треугольники AED и ABD равны по двум сторонам и углу между ними: $AE = AB$, AD — общая сторона и $\angle DAE = \angle DAB$.

Следовательно, $\angle ABC = \angle ABD = \angle AED = 90°$.

Ответ: 90°.

Задача 2.3.3.4. (25 баллов)

Тема: десятичная запись натурального числа.

Условие

В натуральном двузначном числе a цифры поменяли местами и получили двузначное число b. Оказалось, что сумма чисел a и b делится на 5, а их разность — на 27.

Найдите все возможные значения числа a. В ответ запишите сумму всех полученных чисел.

Решение

Пусть $a = \overline{xy} = 10x + y$ и $b = \overline{yx} = 10y + x$.

Тогда

$$a + b = 11x + y = 11(x + y).$$

Так как по условию $a + b = 11(x + y) \vdots 5$ и числа 5 и 11 взаимно просты, то

$$(x + y) \vdots 5. \tag{2.3.1}$$

Далее из второго условия $a - b = 9x - 9y = 9(x - y) \vdots 27$, следует, что

$$(x - y) \vdots 3. \tag{2.3.2}$$

Осталось перебрать все возможные значения цифр x и y, удовлетворяющих условиям (2.3.1) и (2.3.2). Непосредственной проверкой можно убедиться, что этим условиям удовлетворяют пары (1; 4), (2; 8), (4; 1), (5; 5), (6; 9), (8; 2) и (9; 6).

Таким образом, получаем пять чисел, сумма которых равна $14 + 28 + 41 + 55 + + 69 + 82 + 96 = 385$.

Ответ: 385.

Задача 2.3.3.5. (25 баллов)

Тема: текстовая задача.

Условие

Команда «Математики» за последние три года, согласно протоколам, приняла участие в 111 матчах по мини-футболу (в это число вошли и игры, которые были отменены по техническим причинам). При анализе результатов было замечено:

- сколько-то игр было выиграно;
- ничьи составляют 45% от всех игр, в которых не были одержаны победы;
- количество матчей, в которых были допущены поражения, к количеству отмененных игр относится как $1:2$.

Какое количество матчей «Математики» проиграли?

Решение

Пусть было одержано x побед. Тогда количество игр, которые были сыграны вничью, проиграны или были отменены, равно $111-x$.

Тогда $\frac{9}{20}(111-x)$ — количество игр, сыгранных вничью.

Найдем количество игр, которые были проиграны, или отменены:

$$(111-x)-\frac{9}{20}(111-x)=\frac{1221-11x}{20}.$$

Тогда количество игр, в которых были поражения, равно

$$y=\frac{1221-11x}{60}\in Z.$$

Получили диофантово уравнение

$$11x+60y=1221.$$

Выразим x:

$$x=111-60\cdot\frac{y}{11}.$$

Таким образом, $y\vdots 11$ и $y>0$.

Рассмотрим различные случаи относительно y:

1. $y=11$. Тогда $x=111-60=51$.
2. $y=22$. Тогда $x=111-120=-9$. Количество игр не может быть отрицательным числом. Следовательно, данный случай, как и все последующие, не подходит.

Таким образом, количество игр, в которых были получены поражения, равно 11.

Ответ: 11.

2.3.4. Вторая волна. Задачи 10–11 класса

Задачи второй волны предметного тура по математике за 10–11 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63477/enter/.

Задача 2.3.4.1. (10 баллов)

Темы: стереометрия, центральная симметрия.

Условие

Прямоугольный параллелепипед имеет объем, равный 30. Его рассекли на две части, проведя плоскость через точку пересечения всех трех его диагоналей.

Чему равно максимальное значение объема одной из этих двух частей?

Решение

При центральной симметрии плоскости переходят в параллельные им плоскости, а прямые — в параллельные им прямые. Диагонали параллелепипеда делятся точкой пересечения пополам, поэтому он имеет центр симметрии. При центральной симметрии любая точка параллелепипеда, не находящаяся на секущей плоскости, перейдет в точку, которая находится с другой стороны от любой плоскости, проходящей через этот центр, так как эти две точки и центр симметрии находятся на одной прямой, которая пересекает эту плоскость.

Таким образом, плоскость делит параллелепипед на две части, которые переходят друг в друга при центральной симметрии. Следовательно, их объемы должны быть равны. Это означает, что часть параллелепипеда имеет объем, равный половине объема параллелепипеда

Ответ: 15.

Задача 2.3.4.2. (15 баллов)

Темы: теорема Виета, многочлены.

Условие

Путешественник достал древнюю карту спрятанных сокровищ на острове Пасхи. Путь к пещере, в которой пиратами был закопан клад, был зашифрован с помощью квадратного уравнения. К сожалению, с течением времени запись одного из коэффициентов стерлась, и поэтому путешественник не смог его точно восстановить.

Оказалось, что оно имеет следующий вид: $x^2 + 6x + a = 0$.

Здесь буквой a обозначен неизвестный коэффициент.

Уравнение использовалось для того, чтобы можно было разделить инструкцию по поиску сокровища на несколько частей таким образом, чтобы совершенно невозможно было бы понять, что и где искать, если хотя бы одной части недостает.

У путешественника были все части инструкции, поэтому он смог понять, что нужно от нужной точки на побережье идти ровно P км на юг вдоль единственной тропы, затем $Q = \dfrac{P}{2}$ км на запад, а потом повернуться на северо-восток и идти прямо, пока вершина вулкана Теревака, кратер Рано-Арои, не станет виден под углом

ровно $10R^\circ$ над уровнем горизонта. Рядом с этим местом и находится пещера. Здесь P, Q — корни данного квадратного уравнения, упорядоченные по возрастанию, $R = 2P + Q$.

Может ли путешественник, исходя из данных условий, однозначно найти два этих корня?

Если может, напишите в ответ число R. Если не может, напишите в ответ число 0.

Решение

Запишем теорему Виета для квадратного уравнения:

$$\begin{cases} P + Q = -6, \\ PQ = a. \end{cases}$$

В условии указано, что $Q = \dfrac{P}{2}$. Подставив в первое уравнение, получаем, что $P = -4$, $Q = -2$.

Ответ: -10.

Задача 2.3.4.3. (20 баллов)

Темы: арифметическая задача, симметрия.

Условие

Исследователи выращивают экспериментальную культуру грибов. Эти грибы размножаются почкованием. Гриб порождает два новых гриба каждые 4 ч. Только что появившийся гриб слишком маленький, и поэтому он должен еще 6 ч расти, прежде чем размножаться, таким образом, первое потомство от нового гриба возникает лишь через 10 ч после его появления из почки.

Сколько грибов, включая только что появившихся, будет в лаборатории через 28 ч, если изначально там был один гриб, который породит два новых гриба только через 4 ч.

Решение

Самый первый гриб за 28 ч успеет породить только три поколения грибов, так как для появления четвертого поколения нужно 30 ч. Поэтому чтобы ответить на вопрос задачи, нужно посчитать, сколько грибов успеют отпочковаться от грибов, которые породил первый гриб, а потом посчитать также третье поколение.

Первые два гриба, отпочковавшиеся через 4 ч, создадут еще четыре гриба в 14 ч, еще четыре — в 18 ч, еще четыре — в 22 ч и еще четыре в — 26 ч. Всего они породят 16 грибов.

Вторые два гриба, появившиеся через 8 ч, создадут еще четыре гриба в 18 ч, еще четыре — в 22 ч и еще четыре гриба — в 26 ч. Всего они породят 12 грибов.

Третьи два гриба, появившиеся через 12 ч, создадут еще четыре гриба в 22 ч, и еще четыре гриба — в 26 ч. Всего они породят восемь грибов.

Четвертые два гриба, появившиеся через 16 ч, создадут еще четыре гриба в 26 ч.

Пятые два гриба — в 20 ч, шестые два гриба — в 24 ч, а седьмые два гриба в 28 ч не успеют породить никаких новых грибов — это еще шесть грибов.

Таким образом, можно посчитать количество грибов первого и второго поколения:

$$N_1 = 7 \cdot 2 = 14;$$
$$N_2 = 16 + 12 + 8 + 4 = 40.$$

Осталось посчитать третье поколение. Оно образуется в 24 ч и 28 ч из первых четырех грибов из первых двух грибов, в 28 ч из вторых четырех грибов из первых двух грибов и в 28 ч из первых четырех грибов из вторых двух грибов. То есть еще восемь грибов:

$$N_3 = 2 \cdot 2 \cdot 2 + 2 \cdot 2 \cdot 2 + 2 \cdot 2 \cdot 2 + 2 \cdot 2 \cdot 2 = 32.$$

Суммарно получаем:

$$N = 1 + N_1 + N_2 + N_3 = 1 + 14 + 40 + 32 = 87.$$

Ответ: 87.

Задача 2.3.4.4. (25 баллов)

Темы: прямоугольный треугольник, теорема Пифагора, теорема косинусов.

Условие

В прямоугольном равнобедренном треугольнике ABC с прямым углом C проведена биссектриса AL. Из точки L к стороне BC проведен перпендикуляр, который пересек сторону AB в точке M. Перпендикуляр, построенный к стороне AB в точке M, пересекает сторону AC в точке N.

Чему равен угол ANL? Ответ приведите в градусах.

Решение

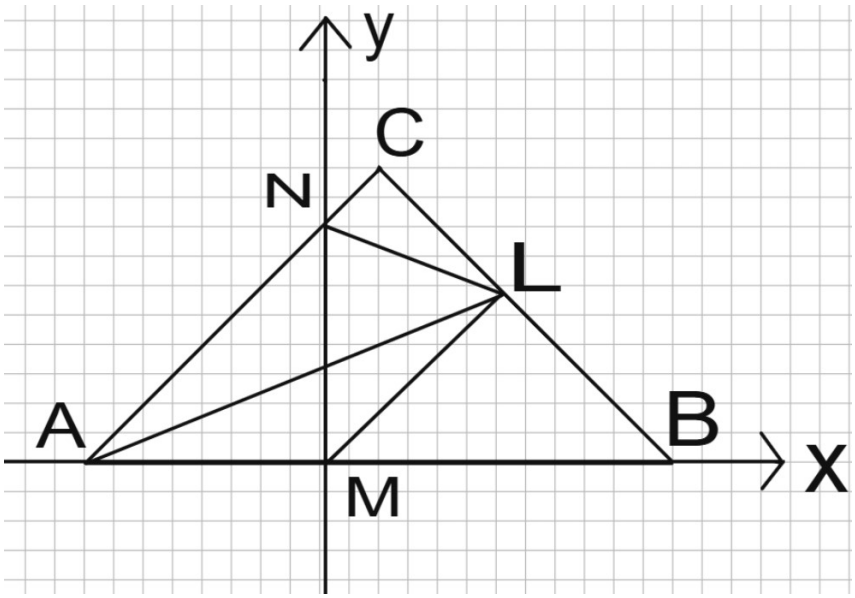
$MN = AM$, значит, угол ANM тоже равен 45° и NM перпендикулярно AB.

Тогда углы NML и BML тоже равны по 45°.

Пусть $AM = 1$ (в условии никаких длин нет, поэтому можем за единицу длины взять любой отрезок). Тогда

$$AN = \sqrt{2};$$

$$AL = 2 \cdot AM \cdot \cos 22{,}5^\circ = 2 \cdot \sqrt{\frac{\sqrt{2}+2}{4}} = \sqrt{\sqrt{2}+2}.$$



Чтобы найти NL, используем метод координат. Проведем горизонтальную ось через AB, вертикальную ось через MN. Тогда точка N имеет координаты (0; 1). Что же касается точки L, то ее координаты x и y совпадают, а длина ML равна 1. Следовательно, они равны $L\left(\frac{\sqrt{2}}{2}; \frac{\sqrt{2}}{2}\right)$.

Используя формулу расстояния между двумя точками, получаем:

$$NL^2 = \frac{1}{2} + \left(1 - \frac{\sqrt{2}}{2}\right)^2 = 2 - \sqrt{2}.$$

Обозначив угол ALN за x, применим теорему косинусов:

$$2 = 2 - \sqrt{2} + \sqrt{2} + 2 - 2 \cdot \sqrt{\sqrt{2}+2} \cdot \sqrt{2-\sqrt{2}} \cdot \cos x.$$

Отсюда получаем, что:

$$\cos x = \frac{\sqrt{2}}{2},\ x = 45^\circ.$$

Тогда угол ANL равен 180° минус угол ALN и угол NAL:

$$ANL = 180 - 45 - 22{,}5 = 112{,}5.$$

Ответ: 112,5.

Задача 2.3.4.5. (30 баллов)

Темы: уравнение параболы, уравнение касательной, угловой коэффициент наклона прямой.

Условие

Для разработки оптической системы на основе параболических отражателей света потребовалось исследовать оптические свойства парабол. Пусть парабола задана уравнением $y = 16x^2$. Требуется на плоскости найти такую точку O, что все проекции этой точки на касательные к параболе лежат на оси абсцисс. Найдите координаты точки O и запишите их в ответ.

Уравнение касательной прямой к параболе (в заданной точке (x_0, y_0)) однозначно устанавливается как уравнение невертикальной прямой, проходящей через (x_0, y_0) и имеющей единственную точку пересечения с параболой.

Решение

Рассмотрим точку с абсциссой x_0 на параболе. Уравнение прямой, проходящей через эту точку, в общем виде имеет вид:

$$y = a \cdot (x - x_0) + 16 \cdot (x_0)^2.$$

Приравняем его к уравнению параболы и найдем, при каком значении a они будут иметь ровно одну точку пересечения:

$$16 \cdot x^2 = a \cdot (x - x_0) + 16 \cdot (x_0)^2;$$

$$16 \cdot x^2 - a \cdot x + x_0 \cdot a - 16 \cdot (x_0)^2 = 0;$$

$$D = a^2 - 4 \cdot 16 \cdot (x_0 \cdot a - 16 \cdot x_0^2) = (a - 32 \cdot x_0)^2 = 0;$$

$$a = 32 \cdot x_0.$$

Итак, запишем уравнение касательной в этой точке к параболе в виде

$$y = 32 \cdot x_0 \cdot (x - x_0) + 16 \cdot (x_0)^2.$$

Эта прямая пересечет ось абсцисс в точке с координатой $x_1 = \dfrac{x_0}{2}$.

Уравнение прямой, проходящей через эту точку перпендикулярно касательной:

$$y = -\frac{2 \cdot x - x_0}{64 \cdot x_0}.$$

Эта прямая пересечет ось ординат в точке с координатами $(0; 0{,}015625)$. Координаты этой точки не зависят от значения x_0, а значит, все такие прямые пройдут через эту точку.

Ответ: $(0; 0{,}015625)$.

2.3.5. Третья волна. Задачи 8–9 класса

Задачи третьей волны предметного тура по математике за 8–9 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63461/enter/.

Задача 2.3.5.1. (15 баллов)

Тема: текстовая задача.

Условие

Начинающий предприниматель Петров закупил 1 000 единиц некоторого товара и попытался его продать с наценкой 20% за единицу продукции. Однако ожидания предпринимателя не совпали с реальностью, и он смог продать только 40% от своего объема, после чего вынужден был снизить цену на товар на 10%. В результате снижения единица товара стала стоить 5 832 руб. за штуку.

Какую чистую прибыль, то есть разность между деньгами, полученными за продажу товара и затратами на его закупку, получил Петров?

Решение

Пусть x руб. — цена за единицу товара, по которой совершена закупка предпринимателем Петровым. Тогда он первоначально планировал осуществить продажи по цене

$$x + 0{,}2x = 1{,}2x.$$

После снижения же цены товар стал стоить

$$1{,}2x - 0{,}1 \cdot 1{,}2x = 1{,}08x.$$

Так как известно, что после снижения единица товара стала стоить 5 832 руб. за штуку, то

$$1{,}08x = 5\,832x = 5\,400.$$

Таким образом, товар был закуплен 5 400 руб. за штуку, и общие затраты на его покупку составили 5 400 000 руб.

Согласно условию задачи 400 единиц товара было продано по цене $1{,}2 \cdot 5\,400 = 6\,480$ руб., и всего было получено за них $6\,480 \cdot 400 = 2\,592\,000$ руб.

Оставшиеся же 600 единиц были проданы по цене 5 832 руб. и получено за них $5\,832 \cdot 600 = 3\,499\,200$ руб.

Тогда чистая прибыль предпринимателя Петрова будет равна

$$2\,592\,000 + 3\,499\,200 - 5\,400\,000 = 691\,200 \text{ руб.}$$

Ответ: 691 200.

Задача 2.3.5.2. (15 баллов)

Тема: комбинаторика.

Условие

Сколько существует нечетных пятизначных чисел, в которых есть хотя бы одна цифра 5?

Решение

Для того чтобы найти количество требуемых чисел, достаточно из общего количества пятизначных нечетных чисел вычесть количество чисел, в которых отсутствует цифра 5.

В десятичной записи нечетного пятизначного числа на последнюю позицию претендует пять вариантов (цифры 1, 3, 5, 7 и 9), на первую — девять вариантов (все цифры, кроме нуля), а на все остальные позиции — по 10 вариантов. Тогда общее количество пятизначных нечетных чисел будет равно

$$9 \cdot 10 \cdot 10 \cdot 10 \cdot 5 = 45\,000.$$

Для записи нечетного пятизначного числа, в десятичной записи которого отсутствует цифра 5, на каждую соответствующую позицию будет на один вариант меньше, тогда общее количество таких чисел будет равно

$$8 \cdot 9 \cdot 9 \cdot 9 \cdot 4 = 23\,328.$$

Тогда количество пятизначных нечетных чисел, в которых присутствует хотя бы одна цифра 5, равно

$$45\,000 - 23\,328 = 21\,672.$$

Ответ: 21 672.

Задача 2.3.5.3. (20 баллов)

Темы: алгебра, система уравнений.

Условие

Наблюдательный Витя для некоторых двух различных чисел заметил интересную особенность: первое число, увеличенное на 4, будет равно квадрату второго числа, уменьшенного на 2; и наоборот, если ко второму числу прибавить 4, то результат будет равен квадрату первого числа, уменьшенного на 2. Найдите сумму квадратов данных двух чисел.

Решение

Пусть x, y — два исходных различных числа. Тогда согласно условиям задачи будем иметь систему уравнений:

$$\begin{cases} x + 4 = (y - 2)^2, \\ y + 4 = (x - 2)^2. \end{cases}$$

Вычитая из первого равенства второе, получим:

$$x - y = (y - 2)^2 - (x - 2)^2 = (y - x)(x + y - 4).$$

Так как числа x, y различны, то отсюда получаем, что $x + y = 3$.

Складывая же уравнения полученной системы, получим

$$x + y + 8 = (y - 2)^2 + (x - 2)^2 = y^2 - 4y + 4 + x^2 - 4x + 4.$$

Из последнего равенства получаем, что

$$x^2 + y^2 = 5(x + y) = 15.$$

Ответ: 15.

Задача 2.3.5.4. (25 баллов)

Темы: теория чисел, остатки.

Условие

Петя записал на доске три числа 391, 604, 888 и задумчиво сказал Васе: «Если я сейчас эти три числа разделю на одно и то же натуральное число, отличное от единицы, то в результате получу один и тот же остаток».

На какое натуральное число Петя планирует произвести деление исходных чисел?

Решение

Обозначим число, на которое производится деление, через x, а остаток через y.

Тогда каждое из записанных Петей чисел можно представить в виде:

$$391 = xm + y,$$
$$604 = xk + y,$$
$$888 = xn + y,$$

где m, k и n — неполные частные, возникающие при делении.

Вычитая из третьего равенства второе, а из второго — первое, получим:

$$284 = x(n - k),$$
$$213 = x(k - m).$$

Вычтем из верхнего равенства нижнее:

$$71 = x(n - 2k + m).$$

Так как 71 — это простое число, то $71 = 71 \cdot 1$, и, по условию задачи $x \neq 1$, то единственный возможный вариант для делителя Пети равен 71.

Ответ: 71.

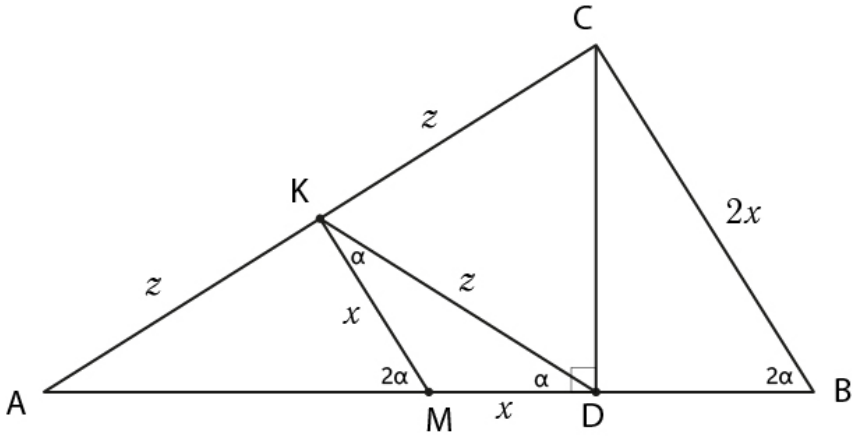
Задача 2.3.5.5. (25 баллов)

Тема: планиметрия.

Условие

CD — высота остроугольного треугольника ABC, M — середина стороны AB и $\angle ABC = 2\angle BAC$. Найдите отношение $BC : MD$.

Решение



На стороне AC отметим ее середину — точку K.

Тогда $AK = KC$ и $AM = MB$ (по условию задачи), следовательно, MK — средняя линия треугольника ABC и $BC = 2MK$.

Докажем, что $MK = MD$.

По свойству медианы прямоугольного треугольника, проведенной из вершины прямого угла, в треугольнике ADC: $DK = \dfrac{1}{2}AC = AK$.

Таким образом, треугольник AKD — равнобедренный и $\angle KAD = \angle KDA$ как углы при основании KD.

Так как MK — средняя линия треугольника ABC, то $MK \parallel BC$ и $\angle AMK = \angle ABC = 2\angle BAC = 2\angle KAD = 2\angle KDA = 2\angle KDM$.

По теореме о внешнем угле для треугольника MKD

$$\angle AMK = \angle KDM + \angle MKD.$$

Тогда из последних двух равенств следует, что $\angle KDM = \angle MKD$ и треугольник MKD — равнобедренный.

Следовательно, $MK = MD$, и так как $BC = 2MK = 2MD$, то $BC : MD = 2 : 1$.

Ответ: 2.

2.3.6. Третья волна. Задачи 10–11 класса

Задачи третьей волны предметного тура по математике за 10–11 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63478/enter/.

Задача 2.3.6.1. (10 баллов)

Темы: осевая симметрия, равнобедренный треугольник, движение.

Условие

Известно, что выпуклая фигура Ф на плоскости устроена таким образом, что она симметрична относительно любой прямой, которая проходит через точку O на этой плоскости. Самое большое расстояние между двумя точками, принадлежащими фигуре Ф, равно дроби, в числителе которой шесть, а в знаменателе квадратный корень из числа π.

Чему равна площадь фигуры Ф?

Решение

1. Докажем, что все точки на границе фигуры равноудалены от центра симметрии O.
   - Возьмем на границе фигуры произвольную точку A. Пусть расстояние $OA = R$.
   - Возьмем любую другую точку B на границе фигуры.
   - Построим прямую I, проходящую через точку O и являющуюся биссектрисой угла AOB.
   - По свойству осевой симметрии, точка A', симметричная точке A относительно прямой I, также принадлежит фигуре Ф.
   - Поскольку I — биссектриса, точка A' попадет на луч OB. Так как при симметрии расстояние до центра сохраняется ($OA' = OA = R$), точка A' совпадет с точкой B только если $OB = R$.
   - Предположим, что $OB > R$. Тогда точка A лежит внутри отрезка OB. Но поскольку фигура выпуклая, весь отрезок OB должен принадлежать фигуре, а значит, и точка A не может быть граничной. Пришли к противоречию.
   - Предположим, что $OB < R$. Тогда точка B лежит внутри отрезка OA. Это также противоречит тому, что B — граничная точка.
   - Следовательно, единственно возможный вариант — $OB = R$.
   - Поскольку точка B была выбрана на границе произвольно, получается, что все точки границы фигуры Ф находятся на одинаковом расстоянии R от точки O. По определению, это окружность.
2. Так как границей является окружность, то сама фигура — круг.
3. Используя данное в условии значение диаметра ($6/\sqrt{\pi}$), находим радиус ($3/\sqrt{\pi}$) и вычисляем площадь, которая равна 9.

Ответ: 9.

Задача 2.3.6.2. (15 баллов)

Темы: составление уравнений, составление пропорций, проценты.

Условие

Находясь на борту космического корабля, главный двигатель за первый час израсходовал 40% всего запаса анобтаниума, а вспомогательные двигатели вместе за это же время израсходовали лишь 300 г анобтаниума. За следующий час главный двигатель израсходовал 80% оставшегося топлива, а вспомогательные двигатели израсходовали 100 г топлива на двоих. В итоге на борту корабля осталось 800 г топлива. Сколько килограммов фантастического топлива было на борту до начала полета?

Решение

Найдем массу анобтаниума, оставшегося к концу первого часа.

Не было израсходовано главным двигателем к этому моменту $100 + 800 = 900$ г.

Это составляет $100 - 80 = 20\%$.

Составим пропорцию и решим ее:

$$\begin{aligned} 20\% &- 900, \\ 100\% &- ? \end{aligned}$$

Значит, к концу первого часа оставалось $900 : 0{,}2 = 4\,500$ г анобтаниума.

Найдем массу топлива к началу первого часа.

Не было израсходовано главным двигателем к этому моменту $4\,500 + 300 = 4\,800$ г, что составляет $100 - 40 = 60\%$.

Составим пропорцию и решим ее:

$$\begin{aligned} 60\% &- 4\,800, \\ 100\% &- ? \end{aligned}$$

Значит, к началу первого часа было $4\,800 : 0{,}6 = 8\,000$ г, что составляет 8 кг.

Ответ: 8.

Задача 2.3.6.3. (20 баллов)

Темы: уравнение параболы, уравнение прямой.

Условие

Известно, что три различные точки $A(2;4)$, $B(x;6)$, $C(6;y)$ расположены на координатной плоскости таким образом, что через них нельзя провести параболу

с вертикальной осью. При этом также известно, что x — минимальное натуральное подходящее число, неравное единице.

Найдите величину $x + y$.

Решение

Через три точки нельзя провести параболу тогда и только тогда, когда они расположены на одной прямой. Действительно, прямая не может пересекать параболу в трех точках, так как квадратное уравнение имеет не больше двух корней. С другой стороны, если три точки не лежат на одной прямой, то через них всегда можно провести параболу. Покажем это.

Пусть на числовой прямой есть три точки с координатами (x_1, y_1), (x_2, y_2), (x_3, y_3). Запишем уравнение параболы в следующем виде:

$$y = y_1 \frac{(x - x_2) \cdot (x - x_3)}{(x_1 - x_2) \cdot (x_1 - x_3)} + y_2 \frac{(x - x_1) \cdot (x - x_3)}{(x_2 - x_1) \cdot (x_2 - x_3)} + y_3 \frac{(x - x_1) \cdot (x - x_2)}{(x_3 - x_1) \cdot (x_3 - x_2)}.$$

Первое слагаемое равно нулю во второй и третьей точке, и равно y_1 в первой, аналогичным образом устроены второе и третье слагаемые, так что это уравнение задает функцию, проходящую через три точки. Однако надо еще проверить, что уравнение задает именно параболу. Для этого нужно, чтобы коэффициент при x_2 не равнялся нулю.

$$\frac{y_1}{(x_1 - x_2) \cdot (x_1 - x_3)} + \frac{y_2}{(x_2 - x_1) \cdot (x_2 - x_3)} + \frac{y_3}{(x_3 - x_1) \cdot (x_3 - x_2)} \neq 0;$$
$$y_1 \cdot (x_2 - x_3) - y_2 \cdot (x_1 - x_3) + y_3 \cdot (x_1 - x_2) \neq 0.$$

Можно убедиться, что это условие означает, что три точки не лежат на одной прямой. А именно, нахождение трех точек на одной прямой можно записать следующим образом:

$$\frac{y_1 - y_2}{x_1 - x_2} = \frac{y_1 - y_3}{x_1 - x_3};$$
$$(y_1 - y_2) \cdot (x_1 - x_3) = (y_1 - y_3) \cdot (x_1 - x_2);$$
$$y_1 \cdot (x_1 - x_3) - y_1 \cdot (x_1 - x_2) = y_2 \cdot (x_1 - x_3) - y_3 \cdot (x_1 - x_2);$$
$$y_1 \cdot (x_2 - x_3) + y_3 \cdot (x_1 - x_2) - y_2 \cdot (x_1 - x_3) = 0.$$

Таким образом, если это условие выполнено, то через три точки проходит прямая, и не проходит никакая парабола. А если оно не выполнено, то проходит единственная парабола, и нельзя провести никакую прямую.

Тогда выразим угловой коэффициент этой прямой тремя разными способами:

$$k = \frac{2}{x - 2} = \frac{y - 6}{6 - x} = \frac{y - 4}{4}.$$

Отсюда получаем, что

$$y = \frac{4 \cdot x}{x - 2}.$$

Минимальное натуральное x, не равное единице, которое подходит — это $x = 3$. Тогда $y = 12$.

Значит, $x + y = 15$.

Ответ: 15.

Задача 2.3.6.4. (25 баллов)

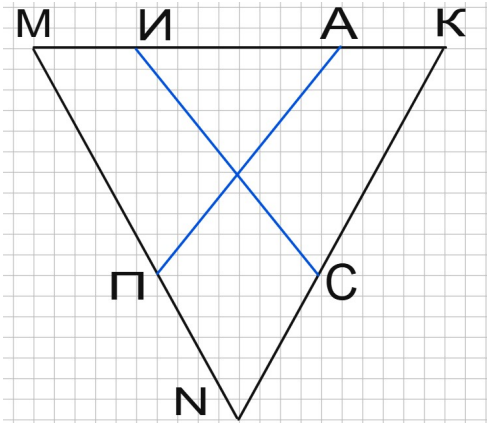
Темы: равносторонний треугольник, первый признак равенства треугольников.

Условие

Три прямые дороги образуют треугольник с равными сторонами, длина которых равна 1 000 м. У этих дорог стоят четыре человека, каждый на обочине одной из трех дорог. Иван и Александр стоят возле одной дороги в 500 м друг от друга. Сергей и Петр стоят у обочин двух других дорог. Сергею идти до Ивана 1 500 м по дорогам кратчайшим путем, Александру до Петра тоже. Между дорогами расположено поле. Какая величина получится, если к расстоянию от Сергея до Ивана по прямой (то есть по полю, а не по дорогам) добавить половину расстояния от Ивана до Александра вдоль дороги, возле которой они стоят, и вычесть расстояние от Александра до Петра по прямой (по полю)?

Решение

Нарисуем расположение всех этих четырех человек. Расположение Сергея и Петра здесь определяется из того условия, что путь до Ивана и Александра соответственно должен занимать 1 500 м, в то время как расстояние по одной стороне не больше 1 000 м, а по другой не больше 500 м.



Используя указанные расстояния, можем записать:

МА + МП = КИ + КС, а значит, МИ + МП = КА + КС = 1 000 м.

МИ + КА = ИА = 500 м. Кроме того, длина КМ равна 1 000 м.

Отсюда выходит, что КА = 1 000 − КС = 1000 − МА, а значит, КС = МА. Аналогично выходит, что КИ = МП.

Тогда треугольники КИС и МПА равны друг другу по двум сторонам и углу между ними.

Следовательно, АП $=$ СИ, а значит, АП $-$ СИ $+ 0{,}5 \cdot$ ИА $= 250$ м.

Ответ: 250.

Задача 2.3.6.5. (30 баллов)

Темы: делители числа, произведение делителей, разложение на множители.

Условие

Количество четных делителей натурального числа в 5 раз больше всех остальных его делителей (рассматриваются все делители, включая само число и единицу). Третья часть всех делителей не делится на 3. Половина четных делителей делится на 5. Само число при этом не превосходит 10 000. Напишите в ответ максимальное число, которое подходит под этим условия.

Решение

Количество четных делителей натурального числа в 5 раз больше всех остальных его делителей.

Это значит, что оно делится на 2^5 степени, но не делится на 2^6.

Третья часть всех делителей не делится на 3.

Это значит, что оно делится на 3^2, но не делится на 3^3.

Половина четных делителей делится на 5.

Это значит, что оно делится на 5, но не делится на 25.

Если перемножим 2^5 на 3^2 и на 5, то получим 1440. Минимальное число, подходящее под условия выше, но большее этого числа, равно $7 \cdot 1\,440 = 10\,080 > 10\,000$.

Следовательно, под все условия подходит только число 1 440.

Ответ: 1 440.

2.3.7. Четвертая волна. Задачи 8–9 класса

Задачи четвертой волны предметного тура по математике за 8–9 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63462/enter/.

Задача 2.3.7.1. (15 баллов)

Темы: теория чисел, признаки делимости.

Условие

На доске записано число 202420252026. Танечка хочет убрать несколько цифр из исходного числа так, чтобы получившийся результат делился на 45 и являлся наибольшим из всех возможных. Какое число запишет на доске Танечка?

Решение

Для того чтобы число Танечки было бы кратно 45, необходимо выполнение условий делимости на 5 и 9. Следовательно, число должно заканчиваться на 0 или на 5. В данном случае первым делом Танечка должна убрать последние две цифры и получить 2024202520.

Для выполнения условия делимости на 9 необходимо, чтобы сумма цифр числа была бы 9. Сумма цифр сейчас равна 19. Ближайшая сумма, кратная 9, равна 18, но 1 в числе нет, следовательно, следующий вариант — 9. Для этого из оставшегося числа ей нужно вычеркнуть цифры, дающие в сумме 10. Тогда наибольшее число, которое может получить Танечка, — 202050.

Ответ: 202050.

Задача 2.3.7.2. (15 баллов)

Тема: десятичная запись натурального числа.

Условие

Найдите все трехзначные натуральные числа $\overline{abc}$, удовлетворяющие условию

$$\overline{abc} = \overline{ab} + \overline{bc} + \overline{ca}.$$

В ответ запишите сумму всех найденных чисел.

Решение

Распишем равенство, заданное в условии задач

$$\overline{abc} = \overline{ab} + \overline{bc} + \overline{ca};$$
$$100a + 10b + c = 10a + b + 10b + c + 10c + a;$$
$$100a + 10b + c = 11a + 11b + 11c;$$
$$89a = 10c + b.$$

Так как a, b, c — цифры, то единственным решением данного уравнения является набор $a = 1$, $b = 9$, $c = 8$. Следовательно, единственное число, удовлетворяющее условию задачи, это 198.

Ответ: 198.

Задача 2.3.7.3. (20 баллов)

Темы: алгебра, квадратный трехчлен.

Условие

Найдите количество значений параметра b, при которых все корни уравнения $x^2 + bx + 2026 = 0$ целые.

Решение

Пусть x_1 и x_2 — целые корни данного уравнения. Тогда согласно теореме Виета:

$$x_1 \cdot x_2 = 2026.$$

Так как 2026 раскладывается на множители

$$2026 = 1 \cdot 2026 = 2 \cdot 1013,$$

то получаем четыре набора для значений корней

$$(1; 2026),\ (-1; -2026),\ (2; 1013),\ (-2; -1013).$$

Зная значения корней, также по теореме Виета найдем значения параметра b:

$$b = -(x_1 + x_2).$$

Таким образом, всего существует четыре значения параметра $b = \{-2027;\ 2027;\ -2015;\ 2015\}$, при каждом из которых уравнение имеет целые корни.

Ответ: 4.

Задача 2.3.7.4. (25 баллов)

Тема: геометрическая вероятность.

Условие

В треугольнике ABC на биссектрисе BD отмечена точка E так, что $BE = ED$. Найти вероятность, что точка, брошенная в треугольник ABC, попадет в треугольник AED, если $AB = 3$ и $BC = 5$.

Ответ выразите в долях и при необходимости округлите его до четвертого знака после запятой.

Решение

Согласно определению геометрической вероятности, требуемая вероятность будет равна отношению площадей треугольников AED и ABC. AE — медиана треугольника ABE, следовательно, $S_{ABD} = 2S_{AED}$.

Площади треугольников ABD и BDC относятся как длины их оснований AD и DC, то есть

$$\frac{S_{ABD}}{S_{BDC}} = \frac{AD}{DC} = \frac{AB}{BC} = \frac{3}{5}.$$

Последнее равенство выполняется согласно свойству биссектрисы BD в треугольнике ABC. Тогда

$$S_{ABC} = S_{ABD} + S_{BDC} = S_{ABD} + \frac{5}{3}S_{ABD} = \frac{8}{3}S_{ABD} = \frac{16}{3}S_{AED}.$$

Из последнего равенства следует отношение

$$\frac{S_{AED}}{S_{ABC}} = \frac{3}{16} = 0{,}1875.$$

Таким образом, вероятность того, что точка брошенная в треугольник ABC, попадет в треугольник AED, равна 0,1875.

Ответ: 0,1875.

Задача 2.3.7.5. (25 баллов)

Тема: алгебра.

Условие

При каком значении числа a сумма квадратов чисел x и y будет принимать наибольшее значение, если известно, что сумма этих чисел равна $2a+1$, а произведение равно $4a^2 + 8a - 4$?

Решение

По условию задачи $x + y = 2a + 1$ и $xy = 4a^2 + 8a - 4$.

Воспользуемся формулой квадрата суммы двух чисел

$$(x + y)^2 = x^2 + 2xy + y^2.$$

Отсюда

$$x^2 + y^2 = (x + y)^2 - 2xy = (2a + 1)^2 - 2(4a^2 + 8a - 4) = 4a^2 + 4a + 1 - 8a^2 - 16a + 8 =$$

$$= -4a^2 - 12a + 9 = -(4a^2 + 12a + 9) + 18 = -(2a + 3)^2 + 18.$$

В полученном выражении первое слагаемое принимает неположительные значения при любом a. Следовательно, сумма квадратов чисел x и y будет максимальной при $2a + 3 = 0$ или $a = -1{,}5$.

Проверим, что при данном значении параметрам $a = -1{,}5$ числа x и y действительно существуют. В этом случае $x + y = -2$ и $xy = -7$.

Выразив из первого равенства $y = -x - 2$ и подставив его во второе, после преобразований получим уравнение $x^2 + 2x - 7 = 0$. Дискриминант данного уравнения равен 32, следовательно, корни уравнения существуют, по которым однозначным образом восстанавливаются решения построенной системы. Откуда и следует существования чисел x и y, заданных в условии задачи.

Ответ: $-1{,}5$.

2.3.8. Четвертая волна. Задачи 10–11 класса

Задачи четвертой волны предметного тура по математике за 10–11 класс открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/63479/enter/.

Задача 2.3.8.1. (10 баллов)

Темы: кратчайший путь, параллельный перенос.

Условие

Склад находится в месте, отмеченном на карте точкой A. Нужно проложить дорогу до берега реки, затем построить мост, перпендикулярный течению реки, и от другого берега проложить дорогу до деревни, отмеченной на карте точкой B. Пример подобного построения на рисунке.

Берега реки здесь нарисованы как параллельные прямые. Координатная ось Ox на рисунке отсчитывает положения моста относительно реки в километрах. В примере, приведенном на рисунке, мост проходит через метку 1 км.

Через какую метку должен проходить мост, чтобы сумма длин пути от склада A до реки по дороге и от противоположного берега реки до деревни B была наименьшей? Ответ дайте в километрах.

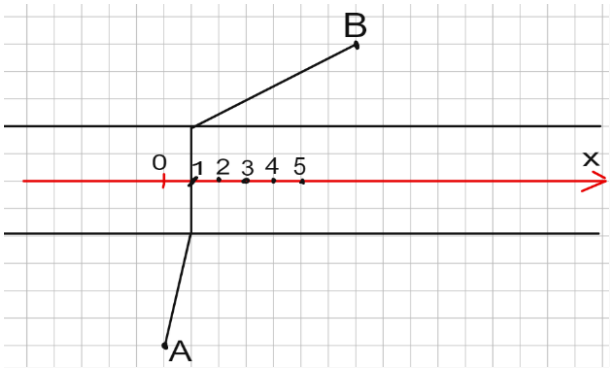


Рис. 2.3.1

Решение

Если вырезать с карты реку и соединить точки A и B прямой, то это и будет кратчайший путь, их соединяющий. Чтобы получить путь до реки, нужно после этого вновь вставить реку. Продемонстрируем эти операции с помощью рис. 2.3.2–2.3.3.

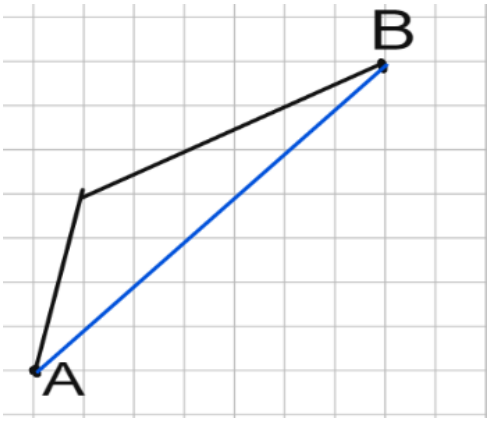


Рис. 2.3.2

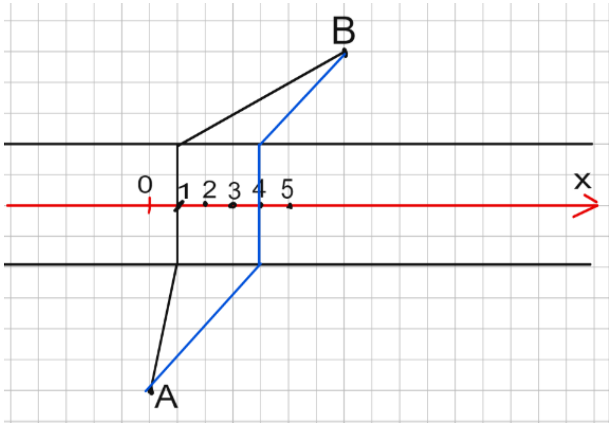


Рис. 2.3.3

Ответ: 4.

Задача 2.3.8.2. (15 баллов)

Темы: составление уравнений, составление пропорций, решение уравнений.

Условие

Два космических корабля стартуют одновременно с одной планеты и направляются к Альфе Центавра, расстояние до которой составляет 4,37 св. лет. Один корабль движется со скоростью 0,1 св. год в год, а другой — со скоростью 0,2 св. год в год.

Через сколько лет расстояние до Альфы Центавра для более быстрого корабля будет в три раза меньше, чем для более медленного корабля? Ответ приведите с точностью до сотых.

Решение

Обозначим время, прошедшее с начала пути, как t лет.

Расстояние, пройденное медленным кораблем, равно $0{,}1t$ св. год, и оставшееся расстояние до Альфы Центавра для медленного корабля $4{,}37 - 0{,}1t$ св. год.

Расстояние, пройденное быстрым кораблем, равно $0{,}2t$ св. год, и оставшееся расстояние до Альфы Центавра для быстрого корабля $4{,}37 - 0{,}2t$ св. год.

По условию задачи, остаток пути для быстрого корабля в 3 раза меньше, чем остаток пути для медленного корабля:

$$4{,}37 - 0{,}2t = \frac{1}{3}(4{,}37 - 0{,}1t).$$

Умножим обе стороны на 3:

$$3(4{,}37 - 0{,}2t) = 4{,}37 - 0{,}1t;$$

$$13{,}11 - 0{,}6t = 4{,}37 - 0{,}1t.$$

Переносим все t в одну сторону и постоянные в другую:

$$13{,}11 - 4{,}37 = 0{,}6t - 0{,}1t;$$

$$8{,}74 = 0{,}5t.$$

Делим обе стороны на $0{,}5$:

$$t = \frac{8{,}74}{0{,}5} = 17{,}48.$$

Таким образом, искомое время равно 17,48 лет.

Ответ: 17,48.

Задача 2.3.8.3. (20 баллов)

Темы: квадратный трехчлен, функции, неопределенные коэффициенты.

Условие

Функция $f(x)$ является квадратным трехчленом и может быть описана следующим образом:

$$f(x) = (f(1) + f(-1) + f(0)) \cdot x^2 + (f(1) + 2 \cdot f(0)) \cdot x - 1.$$

В то же время квадратный трехчлен в общем виде может быть записан так:

$$f(x) = a \cdot x^2 + b \cdot x + c.$$

Найдите минимальное значение величины $a^2 + 2b^2 + 3c^2$ при данных условиях.

Решение

Подставим $f(x)$ в общем виде в первую формулу из условия:

$$a \cdot x^2 + b \cdot x + c = (a + b + c + a - b + c + c) \cdot x^2 + (a + b + c + 2 \cdot c) \cdot x - 1;$$

$$\begin{cases} a = 2 \cdot a + 3 \cdot c, \\ b = a + b + 3 \cdot c, \\ c = -1. \end{cases}$$

Отсюда получаем, что $a = 3$, $c = -1$. Чтобы искомая величина была минимальной, нужно, чтобы коэффициент $b = 0$.

Ответ: 12.

Задача 2.3.8.4. (25 баллов)

Темы: делители числа, произведение делителей, разложение на множители.

Условие

Произведение всех делителей числа 1 000, включая само это число и единицу, равно 10^k.

Чему равно k?

Решение

$$1\,000 = 2^3 \cdot 5^3.$$

Комбинируя все возможные способы выбрать степень двойки и степень пятерки, входящие в делитель, получаем все $(3+1)\cdot(3+1) = 16$ вариантов, каждый из которых соответствует делителю числа. При этом эти 16 делителей можно разбить на пары, произведение в каждой дает 1 000:

$$1\,000 = 1 \cdot 1\,000 = 2 \cdot 500 = 4 \cdot 250 = 8 \cdot 125 = 5 \cdot 200 = 10 \cdot 100 = 20 \cdot 50 = 25 \cdot 40.$$

Тогда выходит, что это будет число $1\,000^8$, а значит, 10^{24}.

Ответ: 24.

Задача 2.3.8.5. (30 баллов)

Темы: равносторонний треугольник, первый признак равенства треугольников.

Условие

Дан квадрат $ABCD$. На сторонах CB и CD отмечены точки L и K соответственно такие, что $CL = CK$. Из точки C на отрезок LD опущен перпендикуляр в точку E.

Пусть $AE = 60$, $EK = 91$. Найдите длину AK.

Решение

Сделаем рис.2.3.4.

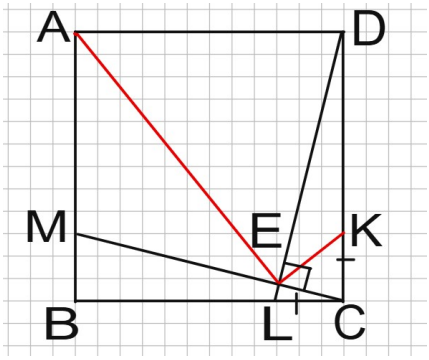


Рис. 2.3.4

Одно из возможных решений заключается в использовании метода координат. Обозначим длину квадрата за единицу, а $CL = LK = x$.

Тогда

$$L(1-x;0);\ K(1;x);\ \overrightarrow{LD} = (x;1);\ \overrightarrow{CE} = t\cdot(1;-x);\ E(1+t;-x\cdot t);\ \overrightarrow{LE} = (t+x;-x\cdot t).$$

Так как вектора LE и LD должны быть сонаправлены, то

$$\frac{t+x}{x} = -x\cdot t;\ t = -\frac{x}{1+x^2};\ E(1-\frac{x}{1+x^2};\ 1-\frac{1}{1+x^2});$$

$$\overrightarrow{AE} = (1-\frac{x}{1+x^2};\ -\frac{1}{1+x^2});\ \overrightarrow{EK} = (\frac{x}{1+x^2};\ x-\frac{x^2}{1+x^2}).$$

Посчитаем скалярное произведение: $\overrightarrow{AE}\cdot\overrightarrow{EK} = 0$. Это значит, что треугольник AEK прямоугольный.

Тогда AK можно найти по теореме Пифагора: $60^2 + 91^2 = 109^2$.

Ответ: 109.

2.4. Инженерный тур

Задачи первого этапа инженерного тура открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/66693/enter/.

Задача 2.4.1. Истинный вкус (10 баллов)

Тема: аукционы.

Условие

На аукционе с пятью участниками представлено десять лотов. Все участники являются покупателями. У каждого лота есть своя реальная стоимость, а каждый участник имеет свое представление о ней. Выгода участника заключается в разнице между реальной стоимостью лота и ставкой, которую заплатил за него участник. Каждый лот выставляется на аукцион по принципу «первой цены» на повышение ставки. Пусть участники делают ставки, равные их представлению о стоимости. Кто из участников получит наибольшую выгоду от участия в аукционе?

В ответ напишите только номер участника.

Ценности лотов: 1,5798; 1,6064; 1,9614; 1,8850; 1,1010; 1,8911; 1,0226; 1,3908; 1,2414; 1,4331.

Ставки:

Участник 1: 1,4677; 1,6054; 2,1281; 2,0558; 1,0331; 1,9565; 1,0470; 1,3317; 1,1777; 1,3046.

Участник 2: 1,7049; 1,5426; 1,8581; 1,9503; 1,2074; 1,8291; 1,0876; 1,5199; 1,2901; 1,5358.

Участник 3: 1,6117; 1,5865; 2,1466; 2,0016; 1,0121; 2,0230; 0,9819; 1,3106; 1,3429; 1,3134.

Участник 4: 1,6150; 1,7188; 1,7872; 1,7634; 1,1143; 1,9733; 1,0473; 1,3088; 1,3005; 1,4478.

Участник 5: 1,5031; 1,4801; 1,9936; 2,0135; 1,0584; 1,7534; 1,0649; 1,3442; 1,1376; 1,5654.

Эта задача проверяет базовые знания теории аукционов, что потребуется на заключительном этапе.

Решение

Принцип аукциона «первой цены» прост: кто сделал максимальную ставку — тот и забирает лот. Следовательно, для каждого лота (столбца таблицы) определяем участника-победителя по наибольшей ставке и прибавляем ему выгоду как разницу между ценностью лота и ставкой победителя.

После вычисления всех лотов получаем следующие итоговые выгоды:

Участник 1: $-0{,}1708$.
Участник 2: $-0{,}4256$.
Участник 3: $-0{,}4186$.
Участник 4: $-0{,}1124$.
Участник 5: $-0{,}1322$.

Наибольшую выгоду (наименьший убыток) получил участник под номером 4.

Ответ: 4.

Задача 2.4.2. Трехмерное путешествие (10 баллов)

Тема: графы.

Условие

Дан куб, изображенный на рис. 2.4.1.

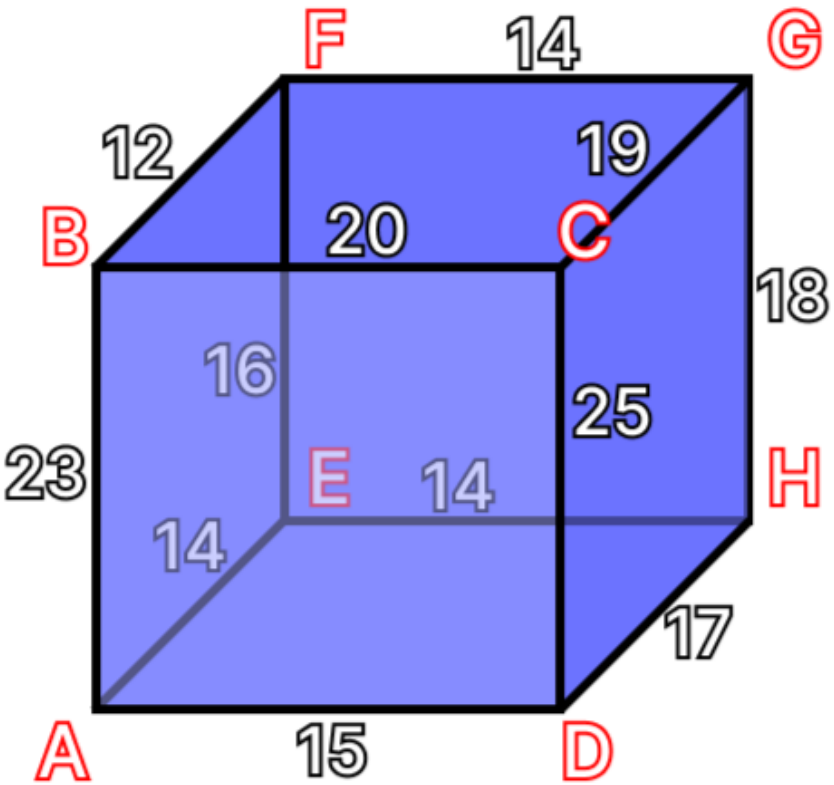


Рис. 2.4.1. Трехмерное путешествие

Каждое ребро куба, без диагоналей, обладает собственными фиксированными затратами на перемещение по ним. Найдите путь по ребрам из узла A в узел G, требующий наименьших затрат. В ответе укажите число — суммарные затраты на этом пути. Задача проверяет навык работы с графовыми моделями, что является частью заключительного этапа.

Решение

Выпишем все возможные пути.

Таблица 2.4.1

Откуда	Куда	Затраты	Откуда	Куда	Затраты
A	B	23	D	H	17
B	F	12	D	C	25
F	G	14	C	B	20
A	E	14	C	G	19
E	H	14	F	E	16
A	D	15	G	H	18

Решение имеет несколько способов, в том числе динамическое программирование (которое можно выполнить даже на бумаге) и реализацию алгоритма Дейкстры. Оптимальный путь, который по суммарным затратам равен 44, выглядит следующим образом: $A \to E \to F \to G$.

Ответ: 44.

Задача 2.4.3. Разрядки на подзарядке (30 баллов)

Тема: теория игр.

Условие

Роботы-уборщики А и Б исправно обслуживают марсианскую солнечную электростанцию уже так долго, что научились испытывать нечеловеческую скуку. В качестве одного из способов скрасить рутину А и Б подключаются к энергонакопителю и играют в разрядки. Правила просты: по очереди каждый робот разряжает накопитель на целое ненулевое число ампер-часов. Максимальную величину разряда за ход роботы определяют случайно перед каждой партией. Кто в свой ход остается без заряда — проиграл. После этого роботы возвращаются к работе, пока накопитель подзаряжается обратно.

Напишите бота для игры в разрядки. Бот должен сыграть 10 партий с проверочной системой. Балл за эту задачу составит отношение выигранных партий к их общему числу.

В каждой партии первый ход за вашей программой. Проверочная система играет идеально, и чтобы все было честно, в случае гарантированного поражения вы можете отказаться от партии в первый ход. Если это действительно так, отказ от партии засчитывается в качестве победы.

Это интерактивная задача, а значит, программа взаимодействует с проверочной системой посредством стандартных потоков ввода и вывода. Иными словами, после отправки сообщения программа должна очистить буфер (выполнить `flush`) и считать ответ от системы (прочесть строку). В случае некорректного ответа проверка прерывается с вердиктом `PE` (`Presentation Error`).

В начале работы система отправляет вашему боту очередное положение и ждет ответа.

Эта задача проверяет базовые навыки программирования и теории игр, что потребуется на заключительном этапе.

Примечания

- Каждое входящее и исходящее сообщение должно сопровождаться переносом строки.
- Очистка буфера делается функцией `sys.stdout.flush()`.

Формат входных данных

Строки, сообщения следующего вида:

- `START 15 3` — начало партии, первое число — заряд накопителя (например, 15 А·ч), далее максимальный разряд за ход;
- `NEXT 12` — очередной ход, числом передается текущий заряд накопителя;
- `ISHOULDGO` — игра окончена, завершите программу штатным образом;
- остальные сообщения содержат сообщение об ошибке, и за ними следует вердикт `PE`.

Формат выходных данных

Строки, сообщения следующего вида:

- `3` — разрядить накопитель на указанное число ампер-часов (например, на 3);
- `X` (латинская «икс») — отказаться от партии (только в первый ход).

Примеры

Пример №1

Стандартный ввод
15 3 16 3 9 5 100 9 100 10 500 101 150 7 27 8 39 4 40 2

Стандартный вывод
3 3 3 3 X 3 5 X 1 10 ...

Тесты

https://disk.yandex.ru/d/m6s19zm25-d5Sw.

Решение

Задача моделирует игру Баше, частный случай игры в Ним. Оптимальная стратегия без обращения к функции Шпрага – Гранди и в целом теории игры Ним сводится к простой формулировке: при максимальном разряде M разряжайте накопитель так, чтобы после вашего хода заряд был кратен $(M + 1)$. С другой стороны, если заряд уже в первый ход кратен $M + 1$, необходимо отказаться от партии.

Авторское решение задачи на языке `Python 3.12` представлено в файле «Разрядки на подзарядке Решение.py» (https://disk.360.yandex.ru/d/2EzK88CZzHPP8A).

Задача 2.4.4. Холодная оптимизация (20 баллов)

Тема: анализ данных.

Условие

Развитие энергосистемы иногда сводится не к строительству новых объектов, а совсем наоборот. Чтобы развить что-то нужное, приходится избавляться от чего-то ненужного. И хорошо, если можно решить эту задачу однозначно.

Представлена энергосистема небольшого микрорайона. В файле https://disk.yandex.ru/d/P-tvXXZzsjFFFQ содержится аналитика по потребителям и генераторам этой системы за последние 10 суток. В первой строке приводятся типы объектов (C — потребитель, G — генератор), во второй строке — тариф в условных единицах, в следующих строках — влияние на энергобаланс в мегаваттах (положительное — генерация, отрицательное — потребление). Тариф потребителя — выплачивается участнику за каждый потребленный мегаватт, тариф генератора — выплачивает участник за каждые сутки. Недостающая или излишняя энергия в каждых сутках покупается или продается на внешнем рынке, тариф продажи — 2 у. е./МВт, тариф покупки — 5 у. е./МВт.

Определите, какой объект нужно убрать из системы, чтобы достичь максимального общего экономического баланса (доходы минус расходы, в условных единицах). В качестве ответа введите вещественное число — значение баланса после удаления объекта.

Обратите внимание на то, что в ответе в качестве разделителя используется точка. Ответ засчитывается, если отличается от эталонного не более, чем на 10^{-2}.

Задача проверяет навыки работы с данными и математическими моделями, что является частью заключительного этапа.

Решение

Заметим, что в модели отличаются цена покупки и продажи на внешнем рынке, а также характер влияния на экономику у обоих типов объектов (потребители платят пропорционально, генераторы — фиксировано). Следовательно, задачу нельзя решить «в лоб», посчитав индивидуальные экономические балансы и выбрав минимальный. Необходимо поочередно убирать каждый объект и пересчитывать энергетический и экономический балансы всей системы. Этот процесс можно ускорить применением Excel или программирования. Общий принцип следующий:

1. Для каждых суток складываем потребление и генерацию, получая суточный энергобаланс. В зависимости от знака, умножаем на соответствующий тариф и получаем изменение экономического баланса. Суммируем все изменения и складываем в общий экономический баланс.
2. Для каждого потребителя перемножаем суммарное потребление на его тариф и прибавляем к общему экономическому балансу.
3. Для каждого генератора умножаем его тариф на 10 (число суток в данных) и вычитаем из общего экономического баланса.

Выполнив эти вычисления с поочередным удалением каждого объекта, получаем ряд значений, среди которых выбираем максимальный. Это и будет ответ.

Решение представлено в таблице: https://disk.yandex.ru/i/3F7YfUQ8VzL2cQ.

Ответ: 2741,83.

Задача 2.4.5. Сомнительная логистика (30 баллов)

Тема: графы.

Условие

Энергоснабжение в отдаленных местностях, где не прокладывают линии электропередач, держится на собственных источниках электроэнергии. В том числе (как ни странно) теплоэнергоцентралях (ТЭЦ), использующих для работы газ, нефть, мазут и даже уголь. Своевременная поставка топлива, в отличие от погодных условий, более контролируема. Особенно если есть возможность обмениваться топливом с соседями. Только бы не запутаться во всех этих дорогах...

Дана сеть двусторонних дорог между населенными пунктами с расстояниям

(в км) по каждому ребру, два населенных пункта, откуда и куда нужно перевезти грузовик с углем, а также фактический маршрут, по которому предлагается его отправить. Рассчитайте, насколько фактический маршрут отличается от кратчайшего.

Задача проверяет навык программной работы с графовыми моделями, что является частью заключительного этапа.

Формат входных данных

В первой строке два числа N (до 100) и M (до 200) — соответственно число населенных пунктов и дорог между ними. Далее идет M строк, в каждой три числа, первые два — соединяемые узлы (нумеруются с нуля!), третье вещественное число — расстояние между ними (до 50). В последней строке приводится предлагаемый маршрут как последовательность номеров узлов через пробел. Последовательность начинается и заканчивается соответственно искомыми начальным и конечными пунктами. Например.

```
5 5
0 1 10.123123
1 2 20.456456
2 3 30.789789
3 4 20.456456
4 0 10.123123
0 1 2 3
```

Формат выходных данных

Единственное вещественное число — разница между длительностью фактического и кратчайшего маршрутов. Ответ засчитывается, если отличается от эталонного не более, чем на 10^{-3} км.

Примеры

Стандартный ввод
33 66 0 15 1.8048578387308658 0 12 27.827199865885426 0 4 11.087019215553134 0 28 2.8397283978530856 0 19 22.484731859522654 1 11 1.7776693647962352 1 8 47.0822317682757 1 27 21.764679873312986 2 32 6.532395422128161 ...

Стандартный вывод
37.952397149109274

Тесты

https://disk.yandex.ru/d/gn3KyWdA7gUokA.

Решение

Считаем расстояние фактического маршрута, после находим кратчайшее расстояние между двух точек (применяя, например, алгоритм Дейкстры), вычитаем первое из второго и приводим в качестве ответа.

Авторское решение задачи на языке Python 3.12 представлено в файле: https://disk.yandex.ru/d/q4-J-UicS38sgA.

3. Второй отборочный этап

3.1. Работа наставника НТО на этапе

На втором отборочном этапе НТО участникам предстоит решать как индивидуальные, так и командные задачи в рамках выбранного профиля. Подготовка к этому этапу требует от них не только глубокого понимания предметной области, но и умения работать в команде, эффективно распределять роли и применять полученные знания на практике. Наставник играет здесь важную роль — он помогает участникам выстроить осмысленную и целенаправленную траекторию подготовки.

Вот основные направления, в которых наставник может поддержать участника:

- **Подготовка по образовательным программам НТО.** Наставник может готовить участников, используя готовые образовательные программы по технологическим направлениям, рекомендованные организаторами, а также адаптировать их под уровень подготовки школьников.
- **Разбор заданий прошлых лет.** Изучение задач второго отборочного этапа прошлых лет помогает участникам понять формат заданий, определить типовые ошибки и выработать стратегии решения.
- **Онлайн-курсы.** Участники могут пройти курсы по разбору задач прошлых лет или курсы, рекомендованные разработчиками отдельных профилей. Наставник может включить эти курсы в план подготовки, а также сопровождать процесс изучения и помогать с возникшими вопросами.
- **Анализ материалов профиля.** Совместный разбор методических материалов, размещенных на страницах профилей, помогает уточнить требования к участникам и направить подготовку на ключевые темы.
- **Практикумы.** Это важный элемент подготовки, позволяющий применять знания на практике. Наставник может:
  - ◇ организовать практикумы по методическим материалам с сайта профиля;
  - ◇ декомпозировать задачи заключительного этапа прошлых лет на отдельные элементы и проработать их с участниками;
  - ◇ провести анализ требуемых профессиональных компетенций и спланировать занятия для развития наиболее значимых из них;
  - ◇ направить участников на практикумы и мероприятия от организаторов, которые анонсируются в официальных сообществах НТО, например, в телеграм-канале для наставников: https://t.me/kruzhok_association.
- **Командная работа.** Одной из ключевых задач наставника на втором этапе является помощь в формировании команды или в поиске подходящей. Наставник может помочь участникам определить их сильные стороны, выбрать роль в команде и сориентироваться в процессе командообразования, включая участие в бирже команд в рамках конкретного профиля.

Если участники не прошли отборочный этап

Случается, что несмотря на усилия и серьезную подготовку, участники не проходят во второй или заключительный этап Олимпиады. В такой ситуации особенно важна поддержка наставника.

- **Поддержка и признание усилий.** Наставнику важно подчеркнуть ценность пройденного пути: полученные знания, навыки, преодоленные трудности и личностный рост. Это помогает участникам сохранить мотивацию и не воспринимать результат как окончательное поражение.
- **Рефлексия.** Полезно организовать встречу для обсуждения впечатления от участия, трудности, с которыми столкнулись школьники и то, что они узнали о себе и команде. Наставник может направить разговор в конструктивное русло: какие выводы можно сделать? Что сработало хорошо? Что можно улучшить?
- **Анализ ошибок и пробелов.** Наставник вместе с участниками анализирует, какие темы вызвали наибольшие затруднения, чего не хватило в подготовке — теоретических знаний, практических навыков, командного взаимодействия. Это позволяет выстроить более эффективную стратегию на будущее.
- **Планирование дальнейшего пути.** Участникам можно предложить:
  - ◇ продолжить углубленное изучение профиля или смежных направлений;
  - ◇ заняться проектной деятельностью, которая укрепит знания и навыки;
  - ◇ сформировать план по подготовке к следующему циклу НТО, начиная с работы над типовыми заданиями и курсами.
- **Создание устойчивой мотивации.** Важно показать школьникам, что участие в НТО — это не просто соревнование, а часть большого образовательного маршрута. Даже неудачный результат может стать толчком к профессиональному росту, если воспринимать его как точку развития, а не как конец пути.

Таким образом, наставник помогает участникам не только готовиться к этапам НТО, но и справляться с неудачами, выстраивать долгосрочную стратегию и сохранять интерес к инженерному и технологическому творчеству.

3.2. Инженерный тур

Командная задача заключительного этапа представляет собой комплексную инженерную задачу и заключается в экономическом и энергетическом моделировании энергосистемы и конкуренции предложенных участниками решений. Участники разрабатывают экономические стратегии, работают с прогнозами погоды и потребления, пишут скрипты по управлению энергообъектами.

Подготовка к заключительному этапу проходит во время предыдущих этапов с постепенным увеличением сложности — во время инженерного тура первого этапа и во время второго этапа.

Второй этап — отборочный этап, задачи которого подобраны таким образом, чтобы объективно проверить индивидуальные знания участников и косвенно оценить личный вклад участников в результат командной работы. Второй тур является не только отборочным соревнованием, но прежде всего — введением в профиль. То есть его содержание (задачи, динамика) несет образовательную составляющую не только в рамках подготовки к заключительному этапу, но и для развития общих навыков (командное взаимодействие, работа со сложными моделями, информационный поиск и нестандартное мышление).

Второй этап включает в себя восемь командных задач. Для решения задач второго этапа необходимы не только школьные знания и умения программы, навыки использования школьных знаний для решения новых задач, но и факультативные знания в области программирования, математики и геометрии. Методические рекомендации к данному этапу позволяют очертить область знаний и навыков для самостоятельного изучения.

Кроме того, задачи второго этапа направлены на развитие элементов научного исследовательского подхода и навыков командной работы. Здесь участники формируют команды и загружают общее командное решение. Функций у этого этапа несколько:

- отбор команд из трех-пяти участников, способных решать сложные задачи;
- ознакомление участников с математическими и физическими концепциями, на которых будет построена задача заключительного этапа;
- предъявление командам требований к уровню программирования, идентичных тем, что будут предъявлены им на заключительном этапе.

Во время второго отборочного этапа участники решают задачи, которые отражают часть большой командной задачи заключительного этапа, и знакомятся с базовыми концепциями — вероятностными прогнозами, графами, математическими моделями. На рис. 3.2.1 представлена схема связей тем задач второго тура и навыков, необходимых для решения задач заключительного этапа.

Рис. 3.2.1. ИЭС Схема связей тем задач

Теория игр

На заключительном этапе участникам предстоит на одном поле столкнуться с другими командами, и понимание основ теории игр позволит объективнее оценивать игровую ситуацию в условиях конкуренции за ресурсы.

Математические модели

Работа со сложными математическими моделями и системами — фундаментальный навык, в полной мере раскрывающийся при работе с комплексной инженерной задачей.

Теория вероятностей

Мир сложен и неустойчив, и задача заключительного этапа моделирует это в полной мере. Лучше заранее подготовиться и научиться работать с вероятностями, для чего второй этап содержит достаточно много задач по теории вероятностей. При этом для полноценной работы не потребуется погружаться в нее с головой — достаточно знания основ математической статистики и распределений случайных величин.

Физика

В работе с энергетическими системами необходимо знать электротехнику, в частности, и физику в плане построения и работы с физическими моделями.

Алгоритмы

Задача заключительного этапа предполагает написание управляющего скрипта, и здесь важную роль играет навык разработки алгоритмов, равно как и поиска подходящих типовых. На проработку этих навыков и рассчитаны задачи раздела «Алгоритмы». При работе с ними важно делать акцент на информационном поиске и умении выявить типовую подзадачу.

Графы

Энергосети — это графы, и с ними нужно уметь работать. В работе с задачами этого раздела в первую очередь важно овладеть основным арсеналом — программные представления графов и базовые алгоритмы. Это может понадобиться при написании управляющего скрипта, не говоря уже о фундаментальном понимании сетей.

Все задачи второго отборочного этапа решаются на платформе Яндекс Контест. В заданное время всем участникам открывается доступ к очередному набору задач (их условиям и проверочной системе). Участники должны написать программу на языке Python 3.12 и загрузить ее текст на сервер. На сервере выполняется тестирование загруженной программы: ее ответы сверяются с ответами эталонного решения, составленного авторами задачи. Их решение требует наличия базовых навыков программирования, поскольку это является необходимой частью комплексной инженерной задачи. Случайные числа, используемые для генерации тестов, являются псевдослучайными. Все проверки программ всех участников производятся на одинаковых данных.

Командная работа начинается именно во время второго этапа. Помимо общих механизмов, здесь созданы условия для командного взаимодействия — ограниченное число попыток на каждую задачу для команды. Это приводит к необходимости уже на начальном этапе договариваться о стратегии сдачи решений общекомандно, и обозначать свою позицию по отношению к каждой задаче внутри команды — берешь ли ты ее на себя, пишешь варианты решений или передаешь ответственность за нее другим участникам команды. Так, задачи разделяются между участниками с учетом их сильных и слабых сторон, что создает предпосылки к формированию ролей в работе на заключительном этапе.

Учет общекомандных попыток ведется посредством внешнего бота, предоставляющего текущую информацию о командных баллах и оставшихся командных попытках. Бот оживляет и делает осмысленным соревнование во время второго тура — оно становится проявленным и работает на повышение мотивации. На протяжении всего второго тура на вебинарах поднимаются темы совместных ресурсов (Яндекс Диск, Github) для структурирования командной работы. Также на вебинарах не всегда возможно присутствие всех членов команды — и это порождает отдельный процесс по синхронизации информации и понимания. В конце второго тура командам предлагается совместная рефлексия с выявлением слабых и сильных сторон у каждого члена команды, определением белых пятен в знаниях и навыках у каждого и распределением нерешенных задач для дорешивания.

3.2.1. Командные задачи

На заключительном этапе команды попадают в мир энергетических систем. Профиль посвящен энергетическим системам ближайшего будущего. Построение архитектуры Интернета энергии предполагает создание многочисленных надежных гибких энергосистем, способных в любой ситуации эффективно распределять электроэнергию, использовать альтернативные источники и взаимодействовать с рынком мощностей.

Прежде всего, второй этап — командный, поэтому оценивается общий результат. Главная задача второго этапа — формирование команды. Чем лучше выстроена командная работа на втором этапе, тем больше шансов на победу у команды в каждом из этапов. Общайтесь, учитесь эффективно распределять задачи, искать сильные стороны каждого участника вашей команды, наращивайте общекомандные навыки — и коммуникативные, и профессиональные. Решайте задачи максимально эффективно и результативно.

Во время второго этапа предлагаются задачи, которые отражают часть большой командной задачи заключительного этапа и знакомят участников с базовыми концепциями. В них затронуты следующие темы:

- теория игр;
- математические модели;
- теория вероятностей;
- графы;
- физика;
- алгоритмы.

Все задачи требуют написания программного кода на языке программирования Python 3.12. Для работы с этим языком достаточно редактора IDLE, входящего в стандартную поставку, но также можно работать с любой удобной IDE (Visual Studio Code, Spyder и т. д.) или онлайн-средой (Repl.it, Google Colab и т. д.). Следует учитывать, что платформа Яндекс Контест использует Python версии 3.12 без дополнительных библиотек.

Все дополнительные материалы к заданиям и исходные коды решений доступны по ссылке: https://disk.yandex.ru/d/ZJQRGmly59ubxA.

Командные задачи второго этапа инженерного тура открыты для решения. Соревнование доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/69917/enter/.

Задача 3.2.1.1. Аукцион второго шанса, подзадача 1 (6 баллов)

Темы: теория игр, знание основ теории аукционов.

Условие

Среди M участников проводятся торги за некоторый лот в формате аукциона второй цены. Лот имеет истинную ценность C, но каждый из них оценивает его по-своему, и эта оценка выражается ставками (они гарантированно различаются).

При передаче проданного лота с победителя удерживается комиссия аукционного дома (10% от цены). Если после покупки лота участником выясняется, что истинная ценность лота меньше, чем он за него заплатил, он выставляет лот обратно на аукцион.

Определите номер участника, у которого в итоге окажется лот.

Формат входных данных

Два числа через пробел, M (натуральное), и C (вещественное). Далее M строк, в каждой вещественное число — ставка соответствующего участника.

Формат выходных данных

Единственное целое число, номер участника-владельца лота (нумерация с 0).

Примеры

Стандартный ввод
203137 4133324.096959806 1868457.6938164262 5589981.056932577 1051762.38668823 4803039.491311131 ... 3513325.4824158354
Стандартный вывод
156066

Тесты

https://disk.yandex.ru/d/T3ujmFZfYUUK0A.

Решение

Выплата за лот складывается из второй ставки (следующей за выигрышной), к которой прибавляется 10%. Если эта сумма больше истинной ценности, лот выставляется повторно.

Оптимальное решение не требует итерационно моделировать аукцион второй цены. Достаточно отсортировать ставки по убыванию и определить первую ставку, которая будет меньше истинной ценности с наценкой 10%. Участник со ставкой выше этой и будет итоговым владельцем лота.

Авторское решение задачи приведено на языке Python 3.12.

Python

```python
from dataclasses import dataclass

@dataclass
class Item:
    bet: float
    participant: int

N, C = input().split()
N = int(N)
C = float(C)

items = []

for i in range(N):
    elem = float(input())
    items.append(Item(bet=elem, participant=i))

items.sort(key=lambda x: x.bet, reverse=True)

winner = -1
for i in range(1, len(items)):
    cost = items[i].bet + items[i].bet * 0.1

    if cost < C:
        winner = items[i-1].participant
        break

print(winner)
```

Задача 3.2.1.2. Аукцион второго шанса, подзадача 2 (6 баллов)

Темы: теория игр, знание основ теории аукционов.

Условие

Среди M участников проводятся торги за некоторый лот в формате аукциона второй цены. Лот имеет истинную ценность C, но каждый участник оценивает его по-своему, и эта оценка выражается ставками (они гарантированно различаются). При передаче проданного лота с победителя удерживается комиссия аукционного дома (10% от цены). Если после покупки лота участником выясняется, что истинная ценность лота меньше, чем он за него заплатил, участник выставляет лот обратно на аукцион.

Какова будет прибыль аукционного дома?

Формат входных данных

Два числа через пробел, M (натуральное), и C (вещественное). Далее M строк, в каждой вещественное число — ставка соответствующего участника.

Формат выходных данных

Единственное вещественное число, прибыль аукционного дома.

Примеры

Стандартный ввод
203137 4133324.096959806 1868457.6938164262 5589981.056932577 1051762.38668823 4803039.491311131 ... 3513325.4824158354

Стандартный вывод
66669815647.2999

Тесты

https://disk.yandex.ru/d/WHbjchpJ9ZSTZg.

Решение

Это продолжение первой подзадачи. Имея отсортированный список ставок, достаточно просуммировать 10% комиссии от всех ставок, начиная со второй максимальной и заканчивая следующей за номером итогового владельца (который определяется в первой подзадаче).

Авторское решение задачи приведено на языке Python 3.12.

Python

```python
from dataclasses import dataclass

@dataclass
class Item:
    bet: float
    participant: int

N, C = input().split()
N = int(N)
C = float(C)

items = []

for i in range(N):
    elem = float(input())
    items.append(Item(bet=elem, participant=i))

items.sort(key=lambda x: x.bet, reverse=True)

winner = -1
```

```
profit = 0.0
for i in range(1, len(items)):
    profit += items[i].bet * 0.1
    cost = items[i].bet + items[i].bet * 0.1

    if cost < C:
        winner = items[i-1].participant
        break

# print(winner)
print(f"{profit:.4f}")
```

Критерии оценивания

Решение принимается, если значение отличается от эталонного не более, чем на 10^{-4}.

Задача 3.2.1.3. Бесконечно недетерминированный конь (15 баллов)

Темы: теория вероятностей, навык эффективной работы с вероятностными моделями.

Условие

Двустороннее шахматное поле шириной пять клеток и длиной L клеток свернули в ленту Мебиуса, а в ее середину поставили шахматного коня. После этого конь делает M случайных ходов по ленте. Какова вероятность, что в итоге конь окажется на клетке, из которой одним ходом можно вернуться в начальное положение?

Формат входных данных

Два натуральных числа через пробел, соответственно L и M. Известно, что $10 \leqslant L \leqslant 200$, $50 \leqslant M \leqslant 300$.

Формат выходных данных

Единственное вещественное число, искомая вероятность.

Примеры

Стандартный ввод
10 51

Стандартный вывод
0.14306267857158905

Тесты

https://disk.yandex.ru/d/n2TzGxcx4PR6Yw.

Решение

Решение строится по принципу динамического программирования. Строим матрицу вероятностей для случая $N = 0$ (какова вероятность, что конь следующим ходом придет в начальное положение?), после чего последовательно пересчитываем матрицу для случая на ход больше, пока не придем к N. Для большей точности предлагается использовать библиотеку `fractions`. Чтобы решение уложилось в ограничение по времени выполнения, следует обратить внимание, что матрица вероятностей симметрична относительно начальной точки по линиям вдоль и поперек ленты. Таким образом, достаточно считать лишь четверть поля.

Авторское решение задачи приведено на языке Python 3.12.

Python

```python
# используем Fraction для большей точности
from fractions import Fraction as frac

L, N = map(int, input().split())
# оптимизация: число ходов должно быть нечётным
if N % 2 == 0:
    print(0.0)
    from sys import exit
    exit()

W = 5
WM, L2 = W // 2, L * 2
WX, LX = WM + 1, L + 1 # размеры матрицы
s_col, s_row = WM, 0 # начальное положение

# матрицы для расчёта вероятностей
probs = [[frac(0) for _ in range(WX)]
         for _ in range(LX)]
new_probs = [[frac(0) for _ in range(WX)]
             for _ in range(LX)]
# смещения для хода коня
deltas = [
    (-2, -1), (-2, +1), (+2, -1), (+2, +1),
    (-1, -2), (-1, +2), (+1, -2), (+1, +2),
]

def wr(col, row, value): # запись в матрицу
    if not 0 <= col < W: return
    # оптимизация: симметрия вдоль ленты
    if col > WM: col = W-1-col
    # учитываем бесконечность ленты
    row = row % L2
    # оптимизация: симметрия от начальной точки
    if row > L: row = 2*L-row
    probs[row][col] = value

def get(col, row): # чтение из матрицы
    if not 0 <= col < W: return frac(0), 0
```

```
    if col > WM: col = W-1-col
    row = row % L2
    if row > L: row = 2*L-row
    return probs[row][col], 1

# заполняем конечное положение
for dc, dr in deltas:
    wr(s_col + dc, s_row + dr, frac(1))
# производим обратный расчёт
for mv in range(N):
    for row in range(LX):
        for col in range(WX):
            a, b = map(sum, zip(
                get(col+2, row+1), get(col+2, row-1),
                get(col-2, row+1), get(col-2, row-1),
                get(col+1, row+2), get(col+1, row-2),
                get(col-1, row+2), get(col-1, row-2)
            ))
            new_probs[row][col] = a / b
    new_probs, probs = probs, new_probs
# выводим ответ
print(float(probs[s_row][s_col]))
```

Критерии оценивания

Решение оценивается по совокупности тестов. Тест засчитывается на 50%, если ответ отличается от эталонного не более, чем на 10^{-5}. Полный балл за тест засчитывается при погрешности не более, чем 10^{-9}.

Задача 3.2.1.4. В поисках недопонятых гениев (18 баллов)

Темы: графы, теория вероятностей, математические модели.

Условие

Арсений Иголкин любит искусство, особенно непризнанное и малоизвестное. В каждом городе он ищет какой-нибудь авангардный или отдаленный музей и часами бродит по нему, разглядывая экспонаты. В очередной поездке у Арсения было совсем немного времени, поэтому из вариантов осталась только галерея современного искусства. Тогда Арсений решил найти в галерее самые непопулярные экспонаты и сосредоточиться на них.

Представим галерею в виде графа из экспонатов и двунаправленных переходов между ними. Вес перехода — это время, за которое от одного экспоната можно добраться до другого. Между двумя экспонатами может быть более одного перехода. Представим блуждающего посетителя, который, начиная от случайного экспоната, делает 100 случайных перемещений по галерее. При этом вероятность выбрать переход пропорциональна весу этого перехода (чтобы было время осмыслить увиденное).

Определите экспонат, который будет посещаться реже всего. Учитываются только посещения с переходов (то есть посещение в самом начале блуждания не рассматривается).

Формат входных данных

Два натуральных числа через пробел, число экспонатов N и число переходов M. Далее идет M строк, в каждой три числа, первые два — целые номера экспонатов (нумерация с нуля) и вес перехода (вещественное ненулевое число).

Формат выходных данных

Единственное целое число, наименее популярный экспонат.

Примеры

Стандартный ввод
5 21 0 1 69 1 2 65 2 3 20 3 4 56 ... 4 3 7

Стандартный вывод
2

Тесты

https://disk.yandex.ru/d/-iYv6zRV4QxMIQ.

Решение

В задаче рассматривается произвольный граф, для которого необходимо рассчитать наименее проходимую вершину. Аналитическое решение имеет слишком высокую вычислительную сложность (для каждой вершины необходимо провести перебор на большую глубину). По этой причине предлагается использование метода Монте-Карло: моделирование достаточно большого числа блужданий, чтобы получить статистику по посещаемости всех вершин графа, после определения наиболее редкой вершины.

Авторское решение задачи приведено на языке Python 3.12.

Python

```python
import random
import sys
from typing import List, Tuple, DefaultDict, Dict
from dataclasses import dataclass
from collections import defaultdict

N = 25000

```

```python
@dataclass
class Transition:
    """
    Однонаправленный переход из одного узла в другой.
    Двунаправленный переход состоит из двух однонаправленных.
    """
    from_node: int  # Откуда
    to_node: int  # Куда
    weigth: int  # Вес
    probabilty: Tuple[float, float] = tuple([0., 0.])  # Вероятность пойти

class Node:
    transitions: List[Transition]

    def __init__(self):
        # Каждый узел характеризуется списком исходящих переходов
        self.transitions = []

    def set_probabilities(self) -> None:
        # Сумма всех весов исходящих переходов
        weigth_sum = sum(transition.weigth for transition in self.transitions)
        # Выставляем вероятности пойти по переходу в виде интервалов
        # (например, [0.0, 0.15], [0.16, 1.0])
        min_value = 0.0
        for transition in self.transitions:
            high_value = min_value + transition.weigth / weigth_sum
            transition.probabilty = [min_value, high_value]
            min_value = high_value

    def get_transition(self) -> Transition:
        # Получить случайный переход
        prob = random.random()
        for transition in self.transitions:
            min_value, max_value = transition.probabilty

            if prob > min_value and prob <= max_value:
                return transition

        raise Exception('No transitions!')

def walk(nodes: Dict[str, Node], N: int):
    # По умолчанию для всех узлов количество переходов равно нулю
    count: DefaultDict[str, int] = defaultdict(lambda: 0)

    def get_random_node() -> Node:
        nonlocal nodes
        node = random.choice(list(nodes.values()))

        return node

    for i in range(N):
        # начинаем в случайном узле
        current_node = get_random_node()
        for _ in range(100):
            transition = current_node.get_transition()
            count[transition.to_node] += 1
            # Добавляем в счетчик, что мы посетили узел
            current_node = nodes[transition.to_node]
```

```
67
68      return count
69
70  data = sys.stdin.readlines()[1:]
71
72  nodes: DefaultDict[str, Node] = defaultdict(lambda: Node())
73  # Считывание исходных данных
74  for unprocessed_transition in data:
75      from_node, to_node, weigth = map(int,
        ↪  unprocessed_transition.split())
76      # Так как переход двухнаправленный, добавляем
77      # переход в оба узла
78      nodes[from_node].transitions.append(
79          Transition(from_node, to_node, weigth))
80      nodes[to_node].transitions.append(
81          Transition(to_node, from_node, weigth))
82
83  # Приводим вероятности в виде интервалов для всех узлов
84  for node in nodes.values():
85      node.set_probabilities()
86
87  statistic = walk(nodes, N)
88  # Сортируем узлы по количеству переходов в них
89  sorted_nodes = sorted(statistic.keys(), key=lambda key:
    ↪  statistic[key])
90
91  print(sorted_nodes[0], flush=True)
```

Критерии оценивания

Решение принимается, если выбранный экспонат является одним из трех наименее популярных, выбранных авторским решением.

Задача 3.2.1.5. За двумя рыбами (32 балла)

Темы: теория игр, теория вероятностей, навык работы с динамической вероятностной моделью.

Условие

У Марины есть детская мечта — стать капитаном корабля. Несколько лет она упорно трудилась: участвовала в НТО, чтобы уметь работать в команде и быть лидером, училась в специализированном институте, работала — и вскоре сможет осуществить свою мечту. Чтобы покрыть расходы на аренду корабля с экипажем, она решает заняться ловлей рыбы — будут и выходы в море, и прибыль.

В течение сезона Марина с командой может сделать 50 выходов в море. В каждый из выходов они могут ловить либо минтая, либо тунца. Улов минтая дает в среднем 15 тыс. у. е. дохода, улов тунца — 30 тыс. Вероятность выловить минтая и тунца отличается, во время выхода заранее неизвестна, но постоянна в течение сезона. Аренда судна с экипажем на сезон стоит 385 тыс. у. е.

Помогите морякам и запрограммируйте такую стратегию ловли, чтобы не выйти в минус, а лучше — максимизировать ожидаемую прибыль.

Формат входных данных

- `POLLOCK` — выйти на минтая;
- `TUNA` — выйти на тунца.

В начале работы система ждет от вашего бота первое сообщение. Приняв последний ответ, ваше решение должно штатно завершиться.

Формат выходных данных

- `NO` — рыба не поймана;
- `YES` — рыба поймана.

Тесты

https://disk.yandex.ru/d/9p-KbBrXMSAvuw.

Решение

В авторском решении сначала производится по пять выходов на минтая и тунца, и рассчитывается статистика успешности каждого выхода. Цель следующих выходов выбирается, исходя из наибольшего математического ожидания прибыли (произведение текущей вероятности успеха и дохода за улов), и также дополняет статистику, влияя на дальнейший выбор.

Авторское решение задачи приведено на языке Python 3.12.

Python

```python
N = 50
REWARD_PO, REWARD_TU = 15, 30
CH_PO, CH_TU = 0, 1
WARMUP = 5

class Thinker:
    def __init__(self):
        self.time = 0
        self.po_found = self.po_fail = 0
        self.tu_found = self.tu_fail = 0

    def think(self):
        # пробные заходы
        if self.time < WARMUP: return CH_TU
        if self.time < WARMUP * 2: return CH_PO
        # считаем вероятности успеха
        po = self.po_found / (self.po_found + self.po_fail)
        tu = self.tu_found / (self.tu_found + self.tu_fail)
        # оцениваем выгоду и принимаем решение
        is_pollock = po * REWARD_PO > tu * REWARD_TU
        choice = CH_PO if is_pollock else CH_TU
        return choice

    def update(self, choice, result):
        # обновление статистики
```

```
26          self.time += 1
27          if choice == CH_PO:
28              if result:
29                  self.po_found += 1
30              else:
31                  self.po_fail += 1
32          elif choice == CH_TU:
33              if result:
34                  self.tu_found += 1
35              else:
36                  self.tu_fail += 1
37
38  t = Thinker()
39  for _ in range(N):
40      # принятие и отправка решения
41      choice = t.think()
42      msg = "POLLOCK" if choice == CH_PO else "TUNA"
43      print(msg, flush=True)
44      # получение и обработка обратной связи
45      answer = input()
46      answer = 1 if answer == "YES" else 0
47      t.update(choice, answer)
```

Критерии оценивания

Это интерактивная задача, значит, программа взаимодействует с проверочной системой посредством стандартных потоков ввода и вывода. Иными словами, после отправки сообщения программа должна очистить буфер (выполнить `flush`) и считать ответ от системы (прочесть строку). В случае некорректного ответа проверка прерывается с вердиктом `PE` (`Presentation Error`).

Решение будет проверено 200 раз, и итоговая оценка будет рассматриваться по среднему полученному результату. За неотрицательный итог начисляется 50%. Максимальный балл присуждается, если результат достигает или обгоняет по прибыли авторское решение.

Задача 3.2.1.6. Обучение переобучением (23 балла)

Темы: теория игр, математические модели, навык работы с оптимизационными моделями и «черным ящиками»

Условие

Кристина решила попытать счастья в мини-игре в своем любимом дачном симуляторе. Мини-игра представляет собой слепой турнир по игре в камень-ножницы-бумага. Суть в следующем: вы с соперником выкидываете по 300 жестов, но жесты друг друга вы не видите. Их видит арбитр, который молчаливо ведет счет и объявляет его в конце турнира. За победу в раунде присуждается 3 очка, за ничью — 1 очко, за проигрыш — 0.

Кристина очень хочет получить золотой наперсток, а для этого в турнире нужно набрать хотя бы 510 очков. Так как турнир играется раз в день, а хочется сейчас, Кристина сохранилась перед началом турнира, и после объявления результатов

загружается обратно. Так она точно выяснила, что соперник будет постоянно выкидывать одни и те же жесты.

Кристина очень усидчивая, но ее терпения хватит только на 100 таких турниров. Помогите Кристине выиграть, реализовав бота для подбора победной комбинации.

Формат входных данных

В одной строке приводится набор, состоящий из слов `ROCK`, `PAPER` и `SCISSORS` через пробел. Количество слов соответствует длительности турнира. Чтобы завершить турнир досрочно и зафиксировать последний результат, отправьте `CHEENAZES` и завершите программу.

В начале работы ваша программа отправляет сообщение первой. Приняв завершающее сообщение, программа должна штатно завершиться.

Формат выходных данных

- 4 999 — турнир сыгран со счетом 4 очка в вашу пользу, осталось 999 тренировочных турниров;
- 12 1 — турнир сыгран со счетом 12 очков, сейчас будет последний турнир;
- 43 `IMTIRED` — последний турнир сыгран со счетом 43 очка, завершите программу.

Тесты

https://disk.yandex.ru/d/ImvUQTFJPkq4-w.

Решение

Первая попытка состоит из случайного набора жестов. Результат этой попытки записывается как наилучший. Дальше производятся попытки улучшить результат путем замены случайного подотрезка другим набором жестов (по сути, максимизация результата по обратной связи). Когда остается последняя попытка, повторно отсылается наилучшее найденное решение.

Авторское решение задачи приведено на языке Python 3.12.

Python

```python
from random import randint

N, SEG_N = 300, 10

def encode(xs): # для формирования вывода
    return ''.join(['ROCK', 'PAPER', 'SCISSORS'][x] for x in xs)

# делаем случайное решение
base = [randint(0, 2) for _ in range(N)]
print(encode(base), flush=True)
# записываем его результат как лучший
```

```
r, _ = input().split()
best_res = int(r)

while True:
    # улучшаем решение заменой случайного подотрезка
    new = base.copy()
    salt_i = randint(0, N-SEG_N-1)
    new[salt_i:salt_i+SEG_N] = [randint(0, 2) for _ in range(SEG_N)]
    print(encode(new), flush=True)
    r, t = input().split()
    # проверяем, улучшилось ли решение
    res = int(r)
    if res > best_res:
        best_res, base = res, new
    # прерываемся, если осталась одна попытка
    if t == "1": break

# отправляем наилучшее решение
print(encode(base), flush=True)
_ = input()

```

Критерии оценивания

Это интерактивная задача, значит, программа взаимодействует с проверочной системой посредством стандартных потоков ввода и вывода. Иными словами, после отправки сообщения программа должна очистить буфер (выполнить `flush`) и считать ответ от системы (прочесть строку). В случае некорректного ответа проверка прерывается с вердиктом `PE` (`Presentation Error`).

Решение будет проверено 200 раз, и итоговая оценка будет рассматриваться по среднему полученному результату. Результат проверки считается по последнему сыгранному турниру.

Задача 3.2.1.7. Тонкая работа (10 баллов)

Темы: физика, математические модели, навык работы со сложными моделями.

Условие

Правильное платоново тело B с ребром длиной L мкм, состоящее из дигидрида кислорода при температуре 273,15 K, с помощью тончайшего молекулярного скальпеля разделили на M одинаковых правильных платоновых тел (обусловимся, что такое разделение всегда возможно). Определите, насколько в итоге изменилась энергия льда. Диаметр молекулы воды примите за 0,28 нм, ее массу — за $2{,}99 \cdot 10^{-23}$ г, а удельную теплоту плавления воды — за 330 Дж/г.

Формат входных данных

В одну строку через пробел — B (английское название тела в нижнем регистре), L (вещественное число, в микрометрах) и M (натуральное). Названия тел единообразны.

Формат выходных данных

Единственное вещественное число, в наноджоулях.

Примеры

Стандартный ввод
dodecahedron 1861.8616334625685 87842

Стандартный вывод
249168.78412828103

Тесты

https://disk.yandex.ru/d/Nt4SHJwOJsyspg.

Решение

1. По типу многогранника и длине грани вычисляем площадь поверхности S.
2. По количеству новых тел m определяем площадь поверхности малого тела S_m (через объем или с помощью пропорции).
3. Вычисляем площадь разрушенного слоя воды. Он составит половину разницы площадей, так как на две новые грани приходится один разрушенный слой:

$$\triangle S = \frac{S - (S_m \cdot m)}{2}.$$

4. Представим занимаемую молекулой воды площадь как круг диаметром 0,28 нм. Тогда количество расплавленных молекул составит отношение площади слоя к площади этого круга.
5. Умножаем число молекул на массу молекулы воды и умножаем на удельную теплоту его плавления.

Таким образом получаем теплоту, выделенную при разделении многогранника на m подобных многогранников.

Авторское решение задачи приведено на языке Python 3.12.

Python

```python
from math import sqrt, pi
S_C = {
    'tetrahedron': sqrt(3),
    'hexahedron': 6,
    'octahedron': 2 * sqrt(3),
    'dodecahedron': 3 * sqrt(25 + 10 * sqrt(5)),
    'icosahedron': 5 * sqrt(3),
}
lam = 330 # Дж/г
d_h2o = 0.28 # нм
S_h2o = ((d_h2o / 2) ** 2) * pi # нм2
```

```
m_h2o = 2.99e-23 # г (10^-23)

# подготавливаем входные данные
b, a, m = input.split()
a = float(a) * 1000 # um -> nm
m = int(m)
# считаем изменение площади поверхности
S = S_C[b] * a**2 # nm2
S_m = S / (m ** (2/3)) # nm2
dS = (S_m * m - S) / 2 # nm2
# пересчитываем в молекулы
n_h2o = dS / S_h2o # unit
# рассчитываем итоговую теплоту
p = n_h2o * m_h2o * lam / 1e-9 # nJ
print(p)
```

Критерии оценивания

Решение засчитывается, если ответ отличается от авторского не более, чем на 25%.

Задача 3.2.1.8. Ориентация на лету (25 баллов)

Темы: теория вероятностей, программирование, навык работы с динамической вероятностной моделью.

Условие

Разведовательный беспилотник «Фаина Клементьевна» совершал плановый облет местности на метеорите, имеющем форму тора, когда его застала сильная космическая буря с грозой. Удар молнии повредил ряд важных модулей, в том числе модуль геопозиционирования, а порыв ветра отнес беспилотник в неизвестную точку метеорита. К счастью, память не пострадала и сохранила точную посекторную карту метеорита с информацией о количестве ориентиров трех типов: холмы, кратеры и вкрапления руды.

Загруженная карта имеет один из двух форматов:

- формат 1 хранит информацию о количестве ориентиров всех типов в пределах данного сектора;
- формат 2 хранит информацию о группе ориентиров в северно-западном углу данного сектора. В данном случае подразумевается, что группы ориентиров расположены на угловых пересечениях секторов, при этом на каждом пересечении встречается произвольное количество ориентиров одного конкретного типа.

Запаса батареи «Фаины» хватит еще на 20 с. За 1 с беспилотник может переместиться в соседний сектор и сразу выполнить быстрое сканирование, в результате которого будет получена информация о примерном количестве ориентиров в текущем секторе. При быстром сканировании «Фаина» обнаруживает их по отдельности с вероятностью 80%, 70% и 60% соответственно.

Вместо очередного перемещения «Фаина» может передать сигнал бедствия в окрестности космоса. Но для успешного обнаружения беспилотник должен передать свои координаты с точностью до сектора.

Опишите стратегию, при которой «Фаина» сможет успешно сориентироваться и запросить помощь.

Формат входных данных

- `MAP 1 20 15` — готовьтесь принять карту в формате 1 размером 15 строк с 20 ячейками в каждой.
- `5;2;3 5;2;3 10;2;3 5;2;30 10;2;3` — передается строка из пяти ячеек формата 1 с информацией об количествах ориентиров, разделенных точкой с запятой.
- `0-20 1-25 2-13 1-23 0-2` — передается строка из пяти ячеек формата 2 с информацией о типе (от 0 до 2) и количестве ориентиров, разделенных дефисом.
- `10 23 10` — в текущем секторе обнаружено такое количество ориентиров.
- `YES` или `NO` — завершите программу.

Формат выходных данных

- `HELPME 0 5` — передать сигнал помощи с координатами (0; 5), столбец и строка соответственно (нумерация координат с 0!).
- `LEFT, RIGHT, UP, DOWN` — переместиться по карте на соседний сектор и отсканировать его.

В самом начале проверочная система присылает вашей программе карту и результат первого сканирования согласно формату. После этого программа должна передать запрос, получить ответ и, если указано, штатно завершиться.

Тесты

https://disk.yandex.ru/d/qCAkr36tVytwBA.

Решение

Авторское решение укладывается в ограничения задачи и работает путем обхода карты фиксированным циклическим набором команд. Результат каждого сканирования накладывается на считанную карту так, чтобы получить ячейки, на которых может находиться беспилотник. Число ориентиров на такой ячейке будет не меньше, чем обнаружено в ходе сканирования. Имея матрицу для предыдущего и текущего сканирования, а также смещение по матрице (на соседнюю ячейку), можно определить пары соседних ячеек, на которых мог находиться беспилотник, и тем самым отсеять часть вариантов. Таким образом, множество ячеек итерационно сводится к единственному варианту.

Авторское решение задачи на языке Python 3.12.

Python

```python
from copy import deepcopy
INF = float('inf')
MAP_N, MAP_M = 30, 30
N_ORIENT = 3
CMD_LEFT, CMD_RIGHT = "LEFT", "RIGHT"
CMD_UP, CMD_DOWN = "UP", "DOWN"
CMD_SOLVE = "HELPME"
DELTAS = {
    CMD_LEFT: (-1, 0), CMD_RIGHT: (1, 0),
    CMD_UP: (0, -1), CMD_DOWN: (0, 1),
}

def find_coord(f):
    # поиск первого ненулевого элемента
    for j in range(len(f)):
        for i in range(len(f[j])):
            if f[j][i]:
                return (i, j)
    return None

def filter_orient(m, sm):
    # получить бинарную матрицу ячеек,
    # где число ориентиров не меньше заданного
    return [
        [ all(x >= rx for (x, rx) in zip(c, sm))
          for c in l
        ] for l in m
    ]

def filter_connected(rm1, rm2, delta):
    # наложение бинарных матриц со сдвигом
    dx, dy = delta
    rm = deepcopy(rm2)
    for j in range(len(rm)):
        for i in range(len(rm[j])):
            x = (i - dx) % len(rm[j])
            y = (j - dy) % len(rm)
            if not rm1[y][x]:
                rm[j][i] = False
    return rm

my_map = None
step_log = []
scan_log = []
filters = []

steps = [
    CMD_DOWN,
    CMD_RIGHT,
    CMD_UP,
    CMD_RIGHT,
]
move_idx = 0

def next_move() -> str:
    global move_idx
    move_idx = (move_idx + 1) % len(steps)
    return steps[move_idx]

```

```
msg = input()
if msg[:3] == "MAP":
    # чтение карты
    toks = msg.split()
    map_type, MAP_N, MAP_M = int(toks[1]), int(toks[2]), int(toks[3])
    my_map = [[None for _ in range(MAP_N)] for _ in range(MAP_M)]
    if map_type == 1:
        # ориентиры помещаются в ячейку как есть
        for j in range(MAP_M):
            line = input().split()
            for i in range(MAP_N):
                c = line[i].split(";")
                my_map[j][i] = tuple(int(c[k]) for k in
                ↪ range(N_ORIENT))
    else: # map_type == 2
        # сначала считываем карту
        raw_map = [[None for _ in range(MAP_N)] for _ in range(MAP_M)]
        for j in range(MAP_M):
            line = input().split()
            for i in range(MAP_N):
                c = line[i].split("-")
                raw_map[j][i] = tuple(int(k) for k in c)
        # кладём ориентиры во все ячейки, граничащие с соответствующим
        ↪ углом
        my_map = [[[0 for _ in range(N_ORIENT)] for _ in range(MAP_N)]
        ↪ for _ in range(MAP_M)]
        for j in range(MAP_M):
            for i in range(MAP_N):
                for dx in (0, 1):
                    for dy in (0, 1):
                        cell = raw_map[(j+dy) % MAP_M][(i+dx) % MAP_N]
                        my_map[j][i][cell[0]] += cell[1]
else:
    raise ValueError("Expected MAP")

while True:
    # запрашиваем результат сканирования
    msg = input().split()
    if msg[0] == "YES" or msg[0] == "NO":
        break # штатное завершение
    msg = tuple(map(int, msg))
    scan_log.append(msg)
    # получаем вероятные ячейки
    if filters:
        filters.append(filter_connected(filters[-1],
        ↪ filter_orient(my_map, scan_log[-1]), step_log[-1]))
    else:
        filters.append(filter_orient(my_map, scan_log[-1]))
    # положение определено однозначно?
    if sum(1 for l in filters[-1] for c in l if c) == 1:
        print("HELPME", *find_coord(filters[-1]), flush=True)
        continue
    # делаем следующий ход
    move = next_move()
    step_log.append(DELTAS[move])
    print(move, flush=True)
```

Критерии оценивания

Это интерактивная задача, значит, программа взаимодействует с проверочной системой посредством стандартных потоков ввода и вывода. Иными словами, после отправки сообщения программа должна очистить буфер (выполнить `flush`) и считать ответ от системы (прочесть строку). В случае некорректного ответа проверка прерывается с вердиктом `PE` (`Presentation Error`).

Решение будет проверено 200 раз, и итоговая оценка будет рассматриваться по числу успешных обнаружений. Решение засчитывается на полный балл, если доля успешных испытаний составит не меньше 90%.

4. Заключительный этап

4.1. Работа наставника НТО при подготовке к этапу

На этапе подготовки к заключительному этапу НТО наставник решает две важные задачи: помощь участникам в подготовке к предстоящим соревнованиям и формирование устойчивой и слаженной команды. Заключительный этап требует высокой слаженности, уверенности и глубоких знаний, и наставник становится тем, кто объединяет усилия участников и направляет их в нужное русло.

Наставник помогает участникам:

- разобрать задания прошлых лет, используя официальные сборники, чтобы понять структуру финальных испытаний, типы задач и ожидаемый уровень сложности;
- изучить организационные особенности заключительного этапа, включая формат проведения, регламент, продолжительность и технические нюансы;
- спланировать подготовку — на основе даты начала финала составляется четкий график занятий, в котором распределены темы, практикумы и командные тренировки;
- обратиться (при необходимости) за консультацией к разработчикам заданий по профилю, уточнить, на какие аспекты подготовки следует обратить особое внимание, и получить дополнительные материалы.

Также рекомендуется участие в мероприятиях от организаторов, таких как:

- установочные вебинары и открытые разборы задач;
- хакатоны, практикумы и мастер-классы для финалистов;
- встречи в онлайн-формате, информация о которых публикуется в группе НТО во «ВКонтакте» и в телеграм-чатах профилей.

Наставнику необходимо уделить внимание работе на формированием устойчивой, продуктивной и мотивированной команды:

- **Сплочение команды.** Это особенно актуально, если участники живут в разных городах. Регулярные онлайн-встречи, совместная работа над задачами и неформальное общение помогают наладить доверие и улучшить командную динамику.
- **Анализ ролей.** Наставник вместе с командой определяет, кто за что отвечает, какие задачи входят в зону ответственности каждого участника. Также обсуждаются возможности взаимозаменяемости на случай непредвиденных ситуаций.
- **Оценка компетенций.** Важно определить, какими знаниями и навыками уже обладают участники, а какие необходимо развить. На основе этого формируется индивидуальный и командный план подготовки.
- **Участие в подготовительных мероприятиях от разработчиков профилей.**

Перед заключительным этапом проводятся установочные вебинары, разборы задач прошлых лет, практикумы, мастер-классы для финалистов. Информация о таких мероприятиях публикуется в группе НТО в VK и в чатах профилей в Telegram.

- **Практика в формате хакатонов.** Наставник может организовать дистанционные хакатоны или практикумы с использованием заданий прошлых лет и методических рекомендаций из официальных сборников.

Таким образом, наставник становится координатором и моральной опорой команды, помогая пройти заключительный этап НТО с максимальной уверенностью и результатом.

4.2. Предметный тур

Задачи третьего этапа предметного тура профиля по информатике открыты для решения. Участие в соревновании доступно на платформе Яндекс.Контест: https://contest.yandex.ru/contest/72664/enter/, https://contest.yandex.ru/contest/72647/enter/.

4.2.1. Информатика. 8–11 классы

Задача 4.2.1.1. Контейнеры (10 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Кот Матроскин дорос до высокой должности в Трансагенстве. Бочку он уже не катит, а занимается отправкой грузов контейнерами. В точке отправления скопились a больших контейнеров и b малых. Матроскин должен спланировать и заказать необходимое количество платформ для их перевозки (открытых вагонов, предназначенных для перевозки грузов, не боящихся атмосферных воздействий). На одну платформу технически можно поместить либо два больших контейнера, либо один большой и два малых. Иных способов загрузить платформу нет. Так как заказ платформы стоит денег, то экономный Матроскин никогда не отправит недогруженную платформу. Если некоторое количество контейнеров не получится загрузить сейчас, он отправит их на ожидание до следующего заказа.

Требуется помочь Матроскину и подсказать, какое максимальное количество платформ он может заказать, чтобы все платформы были полностью загружены контейнерами для отправки.

Формат входных данных

В единственной строке содержатся два целых числа a и b через пробел — количество больших и малых контейнеров в точке отправления соответственно, $0 \leqslant a, b \leqslant 100$.

Формат выходных данных

Вывести в ответ одно число — максимальное возможно количество платформ, которые получится полностью загрузить при указанных выше условиях.

Примеры

Стандартный ввод
5 7

Стандартный вывод
4

Пример программы-решения

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;
signed main(){
    int a, b;
    cin >> a >> b;
    int b1 = min(a, b / 2);
    a -= b1;
    b -= b1 * 2;
    cout << b1 + a / 2 << endl;
}
```

Задача 4.2.1.2. Три города (15 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Три города находятся на одной прямой: Первый и Третий с краев, Второй — где-то между ними. Между Первым и Третьим городом можно перемещаться либо на поезде, либо на самолете, при этом все перемещения проходят через Второй город. Более того, мэр Второго города, в целях поддержки обоих видов транспорта, обязал всех приехавших во Второй город поездом улетать из него на самолете и наоборот, всех прилетевших самолетом — уезжать из Второго города поездом.

Для определенности, пусть расстояние между Первым и Вторым городами равно A, расстояние между Вторым и Третьим городами равно B, причем $A \leqslant B$. Известно, что если из Первого во Второй лететь самолетом, а затем из Второго в Третий ехать поездом, время в пути составит X ч. Если же из Первого во Второй ехать поездом, а затем из Второго в Третий лететь самолетом, то время в пути составит Y ч.

Самолет летит со скоростью, в k раз превышающей скорость поезда. По заданным числам X, Y, k требуется определить, во сколько раз расстояние B больше

расстояния A.

Формат входных данных

В одной строке через пробел заданы три целых числа:

- X — время движения, если из Первого во Второй город лететь самолетом, а затем из Второго в Третий ехать поездом;
- Y — время движения, если из Первого во Второй город ехать поездом, а затем из Второго в Третий лететь самолетом;
- k — коэффициент, показывающий, во сколько раз самолет быстрее поезда.

$1 \leqslant X, Y \leqslant 7 \cdot 10^5$;

$2 \leqslant k \leqslant 1\,000$;

$X \geqslant Y$.

Формат выходных данных

Вывести одно число — отношение, показывающее, во сколько раз расстояние B больше расстояния A.

Тесты, на которых будет проверяться решение, сгенерированы таким образом, что ответ всегда является целым числом.

Примеры

Пример №1

Стандартный ввод
`5 3 3`
Стандартный вывод
`3`

Пример №2

Стандартный ввод
`12 12 3`
Стандартный вывод
`1`

Примечания

Приведем пример, показывающий, как получается ответ для первого теста из условия. Обращаем внимание, что это не единственный пример расстояний, для

которого возможен такой набор данных. В то же время, искомое отношение определяется входными данными всегда однозначно.

После некоторых рассуждений можно получить, что если, например, взять расстояние A равным 15 км, а расстояние B равным 45 км, скорость поезда равной 10 км/ч, а скорость самолета равной 30 км/ч, то получим требуемые результаты:

$$\frac{15}{30} + \frac{45}{10} = 5,$$

$$\frac{15}{10} + \frac{45}{30} = 3.$$

Отношение $\frac{B}{A}$ равно $\frac{45}{15}$ и равно 3.

Пример программы-решения

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
using namespace std;
int main() {
    int X, Y, k;
    cin >> X >> Y >> k;
    int rat= (X * k - Y) / (Y * k - X);
    cout << rat << endl;
}
```

Задача 4.2.1.3. Круглая география (20 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 128 Мбайт.

Условие

Чего только ни встретишь в глубинах космоса! Есть, например, удивительная планета, покрытая океаном. В этом океане встречаются острова. Они имеют форму абсолютно правильного круга. Внутри этих островов могут быть озера. Они тоже имеют форму круга. Внутри этих озер могут быть свои острова. И верно, они тоже имеют форму круга. Внутри этих островов... Ну, вы поняли.

Итак, задано множество окружностей, описывающих эту круглую географию. Для каждой окружности требуется выяснить, что она собой представляет: остров (`Island`) или озеро (`Lake`).

Формат входных данных

Будем считать, что вся планета исходно покрыта океаном, и только незначительную часть ее поверхности занимают острова. Представим эту область, содержащую острова, в виде прямоугольной поверхности (карты) и упрощенно будем считать, что эта поверхность является плоской. Вся остальная часть планеты, не входящая на карту, покрыта океаном, и не представляет для нас интереса.

В первой строке ввода задано одно натуральное число n — количество окружностей, $1 \leqslant n \leqslant 1\,000$. Далее в n строках заданы по три целых числа x, y, R через пробел — описание очередной окружности. Числа x и y задают координаты центра ($-100 \leqslant x, y \leqslant 100$), а R — радиус окружности ($1 \leqslant R \leqslant 100$).

Гарантируется, что никакие две окружности на входе не пересекаются и не касаются.

Формат выходных данных

Вывести в ответ n строк. Для каждой окружности на входе в соответствующую строку вывести либо слово `Island`, если она является островом, либо `Lake`, если она является озером. Порядок вывода должен совпадать с порядком окружностей на входе.

Примечания

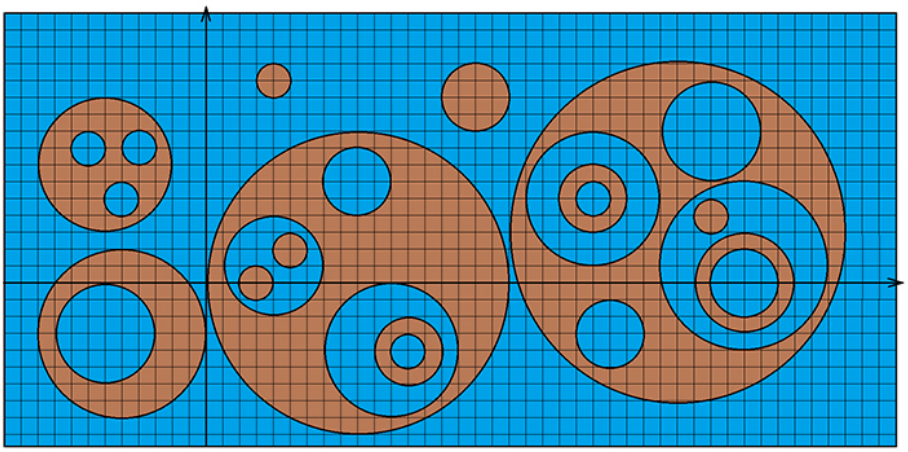


Рис. 4.2.1

На рис. 4.2.1 представлена карта для примера из условия. Синим обозначены водные пространства, коричневым — острова.

Примеры

Стандартный ввод
26 -6 7 4 -7 8 1 -5 5 1 -4 8 1 -5 -3 5 -6 -3 3 3 12 1 16 11 2 9 0 9 9 6 2 4 1 3 3 0 1 5 2 1 11 -4 4 12 -4 2 12 -4 1 28 3 10 23 5 4 23 5 2 23 5 1 24 -3 2 30 9 3 32 1 5 30 4 1 32 0 2 32 0 3

Стандартный вывод
Island Lake Lake Lake Island Lake Island Island Island Lake Lake Island Island Lake Island Lake Island Lake Island Lake Lake Lake Lake Island Lake Island

Пример программы-решения

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define sz(a) (int)a.size()
#define pb push_back
#define all(a) a.begin(), a.end()
#define for0(i, n) for(int i = 0; i < n; i++)
#define for1(i, n) for(int i = 1; i <= n; i++)
#define x first
#define y second
#define int long long
using namespace std;
typedef pair<int, int> pii;
typedef vector<int> vi;
const int INF = 1e18;
const int MOD = 1e9 + 7;
const int LG = 19;


struct circ{
    pair<double, double> c;
    double R;
};


double dist(pair<double, double> a, pair<double, double> b){
    return sqrt((a.x - b.x) * (a.x - b.x) + (a.y - b.y) * (a.y -
    ↪  b.y));
}




bool inside(circ inc, circ outc){
    return (dist(inc.c, outc.c) + inc.R < outc.R);
}


signed main(){
    int n;
    cin >> n;
    vector<circ> v(n);
    for(int i = 0; i < n; i++){
        cin >> v[i].c.x >> v[i].c.y >> v[i].R;
    }
    vi mask(n, 0);
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            if(i != j && inside(v[i], v[j])){
                mask[i]++;
            }
        }
    }
    for(int i = 0; i < n; i++){
       cout << ((mask[i] % 2)?  "Lake\n" : "Island\n");
    }
}
```

Задача 4.2.1.4. Мэллори подменяет сообщение (25 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 64 Мбайт.

Условие

Алиса посылает Бобу сообщение, которое состоит из n положительных целых чисел. Активная злоумышленница Мэллори пытается нарушить целостность сообщения. Она перехватывает эти посылаемые числа и по очереди заменяет их на другие, после чего посылает эти искаженные числа Бобу.

Мэллори не может менять полученные числа произвольным образом. Ее алгоритм построения искаженных чисел следующий: исходно у нее было число $t_0 = 0$. Перехватив очередное число Алисы a_i, Мэллори берет имеющееся у себя на данный момент число t_{i-1} и формирует из него новое число t_i либо как $t_i = a_i + t_{i-1}$ — этот вариант изменения доступен всегда, либо как $t_i = a_i - t_{i-1}$, такой вариант доступен, если $a_i > t_{i-1}$, то есть получающееся в итоге число является положительным и не противоречит характеру посылаемых чисел. После этого Мэллори пересылает полученное число t_i Бобу и использует новое t_i для следующей подмены.

Если вариант замены единственный ($t_{i-1} \geqslant a_i$), то Мэллори делает замену $t_i = a_i + t_{i-1}$ без вариантов. Если же возможны оба варианта замены, то Мэллори поступает случайным образом, и на выходе может получиться как $t_i = a_i + t_{i-1}$, так и $t_i = a_i - t_{i-1}$. При этом известно, что вариант подмены с вычитанием Мэллори сделает ровно k раз из n.

Получив вместо последовательности $a_1,a_2,\ldots,a_n$ последовательность $t_1,t_2,\ldots,t_n$, и, зная все проделки Мэллори, Боб хочет понять объем работы по восстановлению правильного сообщения. По заданному числу k и последовательности t_i он просит вывести сумму всех возможных последовательностей, которые потенциально могла бы исходно выслать ему Алиса. Если нет ни одного сообщения, из которого могла бы получиться последовательность t_i, вывести 0.

Так как ответ может оказаться очень большим, нужно вывести его остаток от деления на число $10^9 + 7$.

Формат входных данных

В первой строке содержатся два целых числа n и k через пробел, $1 \leqslant n \leqslant 1\,000$, $0 \leqslant k \leqslant n$.

Во второй строке содержатся n целых чисел t_i через пробел, $1 \leqslant t_i \leqslant 10^5$.

Формат выходных данных

Вывести одно число — сумму всех возможных последовательностей, которые могла бы потенциально послать Алиса таких, что применив к каждой из них неко-

торым образом свой алгоритм подмены, Мэллори получит последовательность t_i, причем вычитание в каждом из этих вариантов она произведет ровно k раз.

Примеры

Пример №1

Стандартный ввод
5 3 2 5 8 7 12
Стандартный вывод
252

Пример №2

Стандартный ввод
5 0 2 5 8 7 12
Стандартный вывод
0

Пример №3

Стандартный ввод
5 1 2 5 8 7 12
Стандартный вывод
28

Примечания

Рассмотрим первый пример: $n = 5$, $k = 3$. Существует ровно шесть последовательностей, которые исходно могла бы подготовить Алиса для пересылки таких, что, применив к каждой из них некоторым образом алгоритм подмены, Мэллори получит $2, 5, 8, 7, 12$. Приведем все эти последовательности:

$2, 7, 3, 15, 5$ $(2 - 0 = 2,\ 7 - 2 = 5,\ 3 + 5 = 8,\ 15 - 8 = 7,\ 5 + 7 = 12)$;
$2, 3, 13, 15, 5$ $(2 - 0 = 2,\ 3 + 2 = 5,\ 13 - 5 = 8,\ 15 - 8 = 7,\ 5 + 7 = 12)$;
$2, 3, 3, 15, 19$ $(2 - 0 = 2,\ 3 + 2 = 5,\ 3 + 5 = 8,\ 15 - 8 = 7,\ 19 - 7 = 12)$;
$2, 7, 3, 15, 19$ $(2 + 0 = 2,\ 7 - 2 = 5,\ 3 + 5 = 8,\ 15 - 8 = 7,\ 19 - 7 = 12)$;
$2, 7, 13, 15, 5$ $(2 + 0 = 2,\ 7 - 2 = 5,\ 13 - 5 = 8,\ 15 - 8 = 7,\ 5 + 7 = 12)$;
$2, 3, 13, 15, 19$ $(2 + 0 = 2,\ 3 + 2 = 5,\ 13 - 5 = 8,\ 15 - 8 = 7,\ 19 - 7 = 12)$.

В каждом случае для достижения последовательности t_i в исходной последовательности Мэллори производит ровно три вычитания. Можно видеть, что сумма всех чисел во всех этих последовательностях равна 252.

Во втором примере $k = 0$. Можно показать, что Алиса не может сгенерировать ни одной последовательности, из которой можно получить заданную последовательность t_i, ни разу не использовав подмены с вычитанием. Таким образом, ответ равен 0.

Пример программы-решения

Ниже представлено решение на языке C++.

C++

```cpp
#include<bits/stdc++.h>
#define sz(a) (int)a.size()
#define pb push_back
#define all(a) a.begin(), a.end()
#define for0(i, n) for(int i = 0; i < n; i++)
#define for1(i, n) for(int i = 1; i <= n; i++)
#define x first
#define y second
#define int long long
using namespace std;
typedef pair<int, int> pii;
typedef vector<int> vi;
typedef vector<vector<int> > vvi;
const int INF = 1e18;
const int MOD = 1e9 + 7;
const int LG = 19;

signed main(){
	ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
    int n, k;
    cin >> n >> k;
    vi v(n + 1, 0);
    for1(i, n){
        cin >> v[i];
    }
    vi add(n + 1, -1), subs(n + 1, 0);
    for1(i, n){
        if(v[i] > v[i - 1]){
            add[i] = v[i] - v[i - 1];
        }
        subs[i] = v[i] + v[i - 1];
    }
    vector<vector<pii> > dp(n + 1, vector<pii>(n + 1, {0, 0}));
    dp[0][0] = {0, 1};
    for1(j, n){
        for0(i, n + 1){
            if(i > 0){
       dp[i][j].x = (dp[i - 1][j - 1].x + (dp[i - 1][j - 1].y *
        ↪  subs[j]) % MOD) % MOD;
                dp[i][j].y = dp[i - 1][j - 1].y;
            }
            if(add[j] > 0){
           (dp[i][j].x += dp[i][j - 1].x + (dp[i][j - 1].y * add[j]) %
            ↪  MOD) %= MOD;
```

```
                (dp[i][j].y += dp[i][j - 1].y) %= MOD;
            }
        }
    }
    cout << dp[k][n].x << endl;
}
```

Задача 4.2.1.5. Обезьяна и Word (30 баллов)

Имя входного файла: стандартный ввод или `input.txt`.

Имя выходного файла: стандартный вывод или `output.txt`.

Ограничение по времени выполнения программы: 1 с.

Ограничение по памяти: 128 Мбайт.

Условие

Одна обезьяна решила проверить известную гипотезу об обезьянах и печатных машинках. Так как на дворе был XXI век, обезьяна открыла Word и начала печатать в нем случайные слова. i-е по порядку слово она вставляла на p_i-ю позицию, после чего все ранее напечатанные слова, находящиеся в этой или более правой позиции, естественным образом сдвигались на длину этого слова правее. В итоге у обезьяны получился текст из одной строки, разбитой на слова, между которыми стояло ровно по одному пробелу. Знаков препинания, начальных и концевых пробелов в ее тексте не было. Все слова состояли из заглавных и прописных латинских букв.

По заданному порядку слов, которые печатала обезьяна, и позициям, в которые она их вставляла, нужно получить окончательный текст.

Формат входных данных

В первой строке находится число n — количество слов, которые напечатала обезьяна, $1 \leqslant n \leqslant 10^5$.

В следующих n строках находятся по одному слову w_i и одному числу pos_i через пробел. Это означает, что обезьяна вставила слово w_i так, что оно стало на позицию pos_i, при этом все ранее напечатанные слова, имеющие такую или большую позицию, сместились на длину этого слова и одного пробела вправо. Слово w_i состоит из заглавных и прописных латинских букв и имеет длину не более 9. Позиция pos_i, куда его вставила обезьяна, находится в пределах от 1 до i.

Формат выходных данных

Вывести в одну строку итоговый текст, полученный обезьяной. Слова должны разделяться между собой ровно одним пробелом.

Примеры

Пример №1

Стандартный ввод
5 Tolstoy 1 Peace 2 and 2 Leo 1 War 3
Стандартный вывод
Leo Tolstoy War and Peace

Пример №2

Стандартный ввод
7 a 1 bb 1 ccc 1 dddd 1 eeeee 1 ffffff 1 ggggggg 1
Стандартный вывод
ggggggg ffffff eeeee dddd ccc bb a

Пример программы-решения

Ниже представлено решение на языке C++.

C++

```
#include<bits/stdc++.h>
#define sz(a) (int)a.size()
#define pb push_back
#define all(a) a.begin(), a.end()
#define for0(i, n) for(int i = 0; i < n; i++)
#define for1(i, n) for(int i = 1; i <= n; i++)
#define x first
#define y second
#define int long long
using namespace std;
typedef pair<int, int> pii;
typedef vector<int> vi;
const int INF = 1e18 + 1;
const int MOD = 1e9 + 7;
const int LG = 19;
const int N = (1 << LG);
```

```
17
18  vi tr(2 * N, 1);
19  vi res(N, 0);
20  void build(){
21      for(int i = N - 1; i >= 0; i--){
22          tr[i] = tr[2 * i] + tr[2 * i + 1];
23      }
24  }
25
26  void push(int t, int a, int cpos){
27      if(t >= N){
28          res[t - N] = a;
29          return;
30      }
31
32      if(cpos <= tr[2 * t]){
33          tr[2 * t]--;
34          push(2 * t, a, cpos);
35      }
36      else{
37          tr[2 * t + 1]--;
38          push(2 * t + 1, a, cpos - tr[2 * t]);
39      }
40  }
41  signed main(){
42      ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
43      int n;
44      cin >> n;
45      vector<string> v(n);
46      vi pos(n);
47      for0(i, n){
48          cin >> v[i] >> pos[i];
49      }
50      build();
51      for(int i = n - 1; i >= 0; i--){
52          push(1, i, pos[i]);
53      }
54      for0(i, n - 1){
55          cout << v[res[i]]<< ' ';
56      }
57      cout << v[res[n - 1]];
58      cout << endl;
59  }
```

4.2.2. Математика. 8–9 классы

Задача 4.2.2.1. (15 баллов)

Тема: алгебра.

Условие

На сетке расставлены фишки следующим образом: по одной фишке в центре каждой из ячеек и по одной фишке — в каждом из узлов сетки.

Для примера на рис. 4.2.2 приведена сетка размером 2×3, на которой 18 фишек (фишки отмечены синим цветом).

Найдите количество фишек на сетке размером 25×24 ячейки.

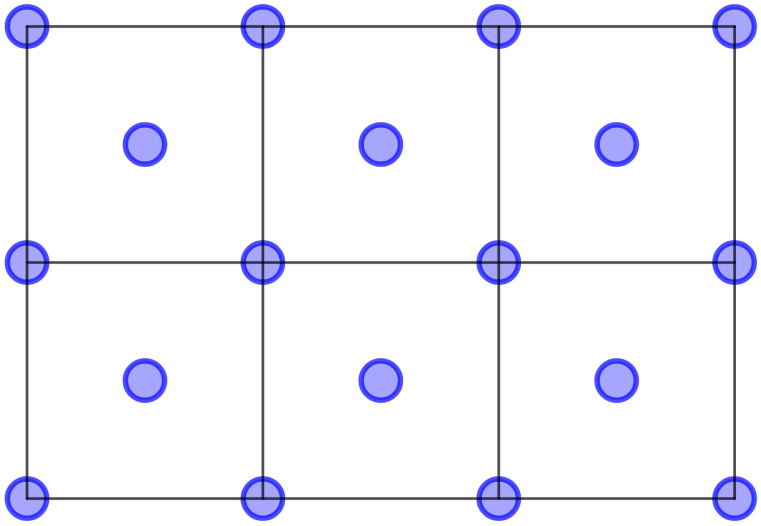


Рис. 4.2.2

Решение

Сетка содержит $25 \times 24 = 600$ ячеек, таким образом, 600 фишек находятся в центрах ячеек.

Количество фишек в узлах $(25 + 1) \cdot (24 + 1) = 650$, так как количество горизонтальных и вертикальных линий, ограничивающих ячейки, на единицу больше, чем число ячеек в строке или столбце.

Значит, всего фишек $600 + 650 = 1\,250$.

Ответ: 1 250.

Критерии оценивания

Только ответ без объяснения — 5 баллов.

Верно посчитано общее количество фишек в центрах ячеек — 5 баллов.

Верно посчитано общее количество фишек в узлах ячеек — 10 баллов.

Арифметическая ошибка при верной логике решения — 12 баллов.

Найдены 600 и 650, но не найдена сумма — баллы не снимаются.

Задача 4.2.2.2. (20 баллов)

Темы: комбинаторика, игры и стратегии.

Условие

В стране 20 городов, некоторые из которых соединены дорогами. Каждая дорога начинается в одном городе, заканчивается в другом и не проходит через остальные города. Дороги не пересекаются. Автолюбители Петя и Коля играют в следующую игру. Вначале Петя определяет два города, в первый из которых перемещается сам, а во второй перемещается Коля. Далее передвигаются по очереди, начинает Коля. За один ход игрок выбирает дорогу, ведущую из города, в котором он находится и едет по ней, попадая в другой город. Если после очередного хода оба игрока оказываются в одном городе, то сделавший ход игрок выигрывает. При каком наибольшем числе дорог может оказаться так, что у Пети есть выигрышная стратегия?

Решение

Оценка. Заметим, что наибольшее число дорог в стране будет в том случае, если каждый город соединен с каждым. В такой стране $\frac{20 \cdot 19}{2} = 190$ дорог. Но если в стране 190 дорог, то Коля первым же своим ходом может приехать в город, в котором находится Петя (каждый город соединен с каждым). Следовательно, в этом случае у Пети нет выигрышной стратегии. Поэтому в стране меньше 190 дорог, т. е. не больше 189 дорог.

Пример. Рассмотрим страну, в которой соединены дорогами все пары городов, кроме пары (A, B). В такой стране 189 дорог. Выигрышная стратегия Пети заключается в том, чтобы выбрать для себя город A, для Коли — город B. В таком случае Коля не сможет выиграть первым своим ходом, и в какой бы город D Коля ни поехал, Петя перейдет из A в D своим первым ходом и выиграет.

Ответ: 189.

Критерии оценивания

Получен только ответ без рассуждений или с неверными рассуждениями — 5 баллов (баллы за этот пункт не складываются с баллами за следующие пункты).

Доказана «оценка», что городов не более 189 — 10 баллов.

Построен «пример» в котором 189 дорог и приведена стратегия за Петю — 10 баллов.

Построен «пример» в котором 189 дорог, но нет стратегии за Петю — 0 баллов.

Задача 4.2.2.3. (20 баллов)

Тема: метод от противного.

Условие

На пальме растут 55 бананов, которые нужно распределить между девятью обезьянами так, чтобы любые две обезьяны получили в сумме хотя бы 9 бананов (некоторые обезьяны могут получить 0 бананов). Докажите, что по крайней мере три обезьяны получат одинаковое количество бананов.

Решение

Предположим противное, что одинаковое количество бананов могут получить не более двух обезьян. Теперь рассмотрим два случая.

1. Каждая обезьяна получила хотя бы 5 бананов. Тогда по 5 бананов получили не более двух обезьян, по шесть бананов — не более двух обезьян и т. д. Поэтому все обезьяны получили не менее $5\cdot 2+6\cdot 2+7\cdot 2+8\cdot 2+9=61$ банан. Противоречие.
2. Хотя бы одна обезьяна получила менее 5 бананов. Обозначим эту обезьяну за A, а за n, $n \leqslant 4$, обозначим количество полученных ею бананов. Рассматривая пары обезьян, одна из которых A, получим, что каждая из оставшихся получила хотя бы $9-n$ бананов. Действуя аналогично предыдущему случаю, получим, что все обезьяны получили в сумме хотя бы $n+(9-n)\cdot 2+(10-n)\cdot 2+(11-n)\cdot 2+(12-n)\cdot 2=84-7n$ бананов. Учитывая, что $n \leqslant 4$, получим $84-7n \geqslant 84-28=56$. Противоречие.

Критерии оценивания

Рассмотрен лишь первый случай или доказано, что есть обезьяна получившая не более 4 бананов — 5 баллов.

Рассмотрен лишь второй случай — 15 баллов.

Второй случай делается перебором возможных значений n, т. е. $n=4,3,2,1,0$. В этом переборе упущен или неверно рассмотрен один случай — 10 баллов за весь второй случай.

В этом переборе упущено более одного случая, при этом рассмотрен случай $n=4$ — 5 баллов за весь второй случай.

Рассмотрен лишь случай $n=4$ без попыток перебора — 5 баллов за весь второй случай.

В этом переборе упущено более одного случая, при этом не рассмотрен случай $n=4$ — 0 баллов за весь второй случай.

Баллы за первый и второй случаи складываются.

Задача 4.2.2.4. (20 баллов)

Тема: геометрия.

Условие

Пусть Ω — описанная окружность остроугольного треугольника ABC. Биссектрисы углов A, B и C пересекают Ω в точках A_1, B_1 и C_1 соответственно, а биссектрисы углов A_1, B_1 и C_1 треугольника $A_1B_1C_1$ пересекают Ω в точках A_2, B_2 и C_2 соответственно. Известно, что наименьший угол треугольника ABC равен $40°$. Найдите наименьший угол треугольника $A_2B_2C_2$.

Решение

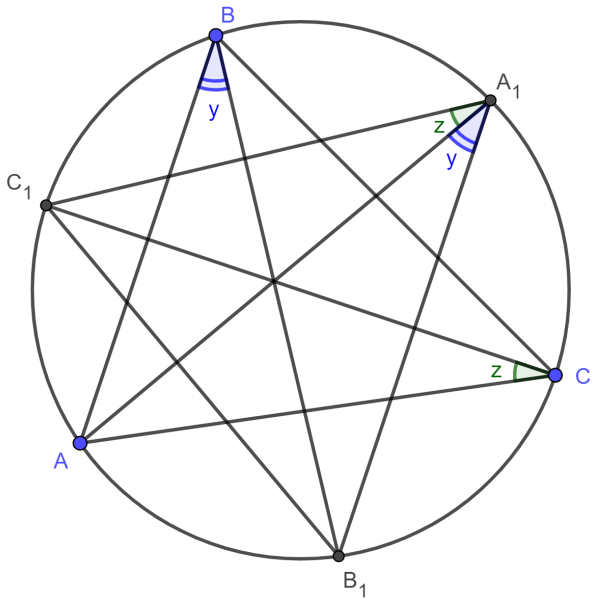


Рис. 4.2.3

Утверждение. Пусть дан треугольник ABC с углами $2x$, $2y$, $2z$, в котором его биссектрисы пересекают его описанную окружность в точках A_1, B_1, C_1 соответственно. Тогда углы треугольника $A_1B_1C_1$ равны $x+y, y+z, z+x$.

Доказательство. Заметим, что $\angle C_1A_1A = \angle C_1CA$ как опирающиеся на дугу C_1A (см. рис. 4.2.3). При этом $\angle C_1A_1A = \dfrac{1}{2}\angle ACB = z$. Аналогично $\angle B_1A_1A = y$. Тогда $\angle B_1A_1C_1 = y+z$. Аналогично для других углов треугольника $A_1B_1C_1$. Утверждение доказано.

Обозначим теперь углы исходного треугольника $4\alpha = 40°, 4\beta, 4\gamma$ $(\alpha \leqslant \beta \leqslant \gamma)$. Применяя полученное выше утверждение для треугольника ABC, получим, что углы треугольника $A_1B_1C_1$ равны $\dfrac{4\alpha}{2}+\dfrac{4\beta}{2} = 2\alpha+2\beta$, $\dfrac{4\beta}{2}+\dfrac{4\gamma}{2} = 2\beta+2\gamma$ и $\dfrac{4\gamma}{2}+\dfrac{4\alpha}{2} = 2\gamma+2\alpha$.

Применяя теперь то же утверждение для треугольника $A_1B_1C_1$ получим, что углы треугольника $A_2B_2C_2$ равны $\alpha+\beta+\beta+\gamma=\dfrac{4\alpha+4\beta+4\gamma}{4}+\beta=45^\circ+\beta$, $45^\circ+\gamma$, $45^\circ+\alpha$. Учитывая, что $\alpha\leqslant\beta\leqslant\gamma$, получим $45^\circ+\alpha\leqslant 45^\circ+\beta\leqslant 45^\circ+\gamma$. Поэтому наименьший угол треугольника $A_2B_2C_2$ равен $45^\circ+\alpha=55^\circ$.

Ответ: 55°.

Критерии оценивания

Дан только верный ответ — 1 балл.

При верном решении используется утверждение без доказательства — снимается 2 балла.

Сформулировано утверждение без дальнейших продвижений — 3 балла.

Сформулировано и доказано утверждение без дальнейших продвижений — 5 баллов.

Вычислены углы треугольника $A_2B_2C_2$ через углы треугольника ABC, ответ не получен — 10 баллов.

Вычислены углы треугольника $A_2B_2C_2$ через углы треугольника ABC и посчитано значение одного из углов, равного 55°, но не доказано, что он наименьший — 15 баллов.

Задача 4.2.2.5. (25 баллов)

Тема: алгебра.

Условие

Найдите всевозможные тройки целых чисел a, b, c, для которых выполняются равенства: $a+b-c=1$ и $a^2+b^2-c^2=-1$.

Решение

Перепишем равенства в виде:

$$a+b=c+1; \tag{4.2.1}$$

$$a^2+b^2=c^2-1. \tag{4.2.2}$$

Равенство (4.2.2) можно записать в виде $(a+b)^2-2ab=c^2-1$. Используя (4.2.1), получим $(c+1)^2-2ab=c^2-1$ или $c^2+2c+1-2ab=c^2-1$, если привести подобные и разделить обе части равенства на 2, получим: $c+1=ab$. Снова используя (4.2.1), получим $a+b=ab\Leftrightarrow ab-a-b+1=1\Leftrightarrow(a-1)(b-1)=1$. Последнее уравнение имеет два решения в целых числах:

1. $a-1=1$ и $b-1=1$, тогда получим тройку $a=2, b=2, c=3$;
2. $a-1=-1$ и $b-1=-1$, тогда получим тройку $a=0, b=0, c=-1$.

Ответ: $(2,2,3), (0,0,-1)$.

Критерии оценивания

Найден только один ответ подбором и показано, что он подходит — 3 балла.

Найдены оба ответа подбором и показано, что они подходят — 5 баллов.

Получено равенство $a+b=ab$ без дальнейших продвижений — 10 баллов.

Получено разложение $(a-1)(b-1)=1$ без дальнейших продвижений — 15 баллов.

Баллы за указанные выше продвижения и ответ (ответы) суммируются.

При решении уравнения в целых числах $(a-1)(b-1)=1$ потерян один из ответов, например, рассмотрен только случай $a-1=1, b-1=1$ — 20 баллов.

Баллы за указанное выше продвижение и ответ (ответы) не суммируются.

4.2.3. Математика. 10–11 классы

Задача 4.2.3.1. (15 баллов)

Темы: алгебра, тригонометрия.

Условие

Сколько должно быть слагаемых под знаком корня в выражении

$$\sqrt[3]{tg^3\frac{\pi}{3}+...+tg^3\frac{\pi}{3}}=6\sqrt{3},$$

чтобы равенство было верным?

Решение

Обозначим количество слагаемых под корнем через n, тогда $\sqrt[3]{n\cdot tg^3\frac{\pi}{3}}=6\sqrt{3}$. Вынося тангенс из-под корня и учитывая, что $tg\frac{\pi}{3}=\sqrt{3}$, получим $\sqrt[3]{n}=6$. Отсюда $n=6^3=216$.

Ответ: 216.

Критерии оценивания

Только ответ — 2 балла.

Записано уравнение, в котором неизвестным является количество слагаемых под корнем, например, $\sqrt[3]{n\cdot tg^3\frac{\pi}{3}}=6\sqrt{3}$ — 5 баллов.

Арифметическая ошибка при логически верном рассуждении — 10 баллов.

Задача 4.2.3.2. (15 баллов)

Тема: игры и стратегии.

Условие

На доске выписаны все натуральные числа от 1 до 999 включительно. Миша и Ян ходят по очереди, начинает Миша. За один ход игрок стирает с доски одно из чисел. Игра заканчивается, когда на доске останется два числа. Если их сумма равна кубу какого-либо целого числа, то выигрывает Миша, в противном случае выигрывает Ян. Кто из игроков имеет выигрышную стратегию, позволяющую ему победить вне зависимости от ходов противника?

Решение

Опишем выигрышную стратегию Миши. Первым ходом Миша стирает число 500, а оставшиеся числа мысленно делит на следующие пары: (1, 999), (2, 998), ..., (499, 501). Далее, какое бы число ни вычеркнул Ян, своим очередным ходом Миша вычеркивает число из той же пары. После 997-го хода, который сделает Миша, останется два числа, причем оба будут входить в одну и ту же пару. Осталось заметить, что сумма чисел в каждой паре равна 1000, что является кубом целого числа.

Ответ: Миша имеет выигрышную стратегию.

Критерии оценивания

Верно указан Миша как игрок, имеющий выигрышную стратегию — 1 балл.

Показано, что последний ход делает Миша — 1 балл.

Приведено разбиение на 499 пар таких, что сумма чисел в каждой паре является кубом целого числа — 5 баллов.

Баллы, указанные в критериях выше, складываются. И не складываются с баллами в нижеследующих критериях.

Приведена выигрышная стратегия за Мишу, возможно, отличающаяся от стратегии, содержащейся в авторском решении, но не доказано, что стратегия выигрышная — 12 баллов.

Задача 4.2.3.3. (20 баллов)

Тема: алгебра и теория чисел.

Условие

Пусть x_1, x_2, x_3, x_4, x_5 — последовательные натуральные числа (именно в таком порядке) такие, что $x_1 + x_2 + x_3 + x_4 + x_5$ — точный куб, а $x_2 + x_3 + x_4$ — точный квадрат. Найдите наименьшее возможное значение, которое может принимать x_3.

Решение

Обозначим x_3 через n. Тогда $x_1 + x_2 + x_3 + x_4 + x_5 = 5n$ и $x_2 + x_3 + x_4 = 3n$.

Так как $5n = m^3$ и 5 — простое число, то m делится на 5, m^3 делится на $5^3 = 125$, а значит, n делится на 25.

С другой стороны, $3n = k^2$, значит, k делится на 3, значит, n делится на 3.

Так как $(5, 3) = 1$ и $5n = m^3$, то n делится на 27. Таким образом, n делится на $25 \cdot 27 = 675$, поэтому $n \geqslant 675$.

Убедимся, что $n = 675$ подходит: $673 + 674 + 675 + 676 + 677 = (3 \cdot 5)^3 = 15^3$, $674 + 675 + 676 = 2025 = 45^2$.

Ответ: 675.

Критерии оценивания

Только ответ без объяснений — 0 баллов.

Ответ и проверка, что ответ подходит — 2 балла.

Доказано, что x_3 делится на 3 — 3 балла.

Доказано, что x_3 делится на 5 — 3 балла (суммируется с предыдущим).

Доказано, что x_3 делится на 27 — 6 баллов.

Доказано, что x_3 делится на 25 — 6 баллов (суммируется с предыдущим).

Доказано, что $x_3 \geqslant 675$ — 15 баллов.

Задача 4.2.3.4. (25 баллов)

Тема: геометрия.

Условие

Дана окружность ω с диаметром AB. На окружности выбрана точка N, отличная от точек A и B. Перпендикуляр из N к AB пересекает AB в точке M и повторно пересекает ω в точке K. Окружность с центром N и радиусом NM пересекает ω в точках P_1 и P_2. Точка L — точка пересечения P_1P_2 с отрезком KN. Докажите, что KP_1 вдвое больше P_1L.

Решение

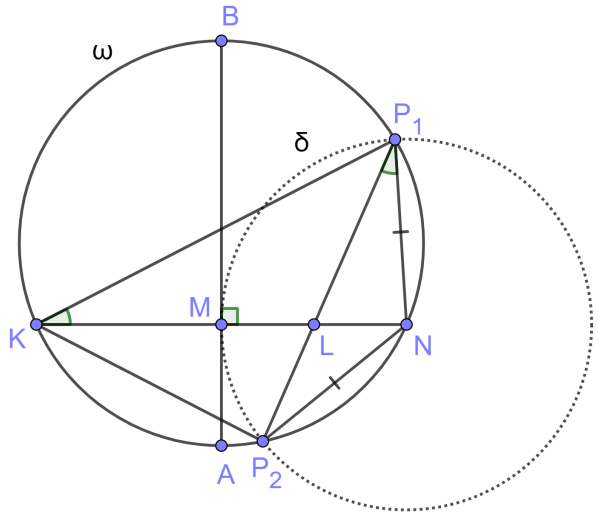


Рис. 4.2.4

Обозначим окружность с центром N и радиуса NM за δ.

Так как хорды P_1N и P_2N равны как радиусы окружности δ, то равны и дуги P_1N и P_2N. Следовательно, равны углы $\angle NKP_1 = \angle NP_1P_2$ как опирающиеся на равные дуги. Тогда треугольники NKP_1 и NP_1L подобны по двум углам ($\angle NKP_1 = \angle NP_1L$ и $\angle KNP_1$ — общий). Тогда $\dfrac{KP_1}{P_1L} = \dfrac{NK}{NP_1}$. Заметим теперь, что $NP_1 = NM$ как радиусы окружности δ и $NM = MK$, так как диаметр делит перпендикулярную ему хорду пополам. Поэтому $\dfrac{NK}{NP_1} = \dfrac{2NM}{NM} = 2$, откуда $\dfrac{KP_1}{P_1L} = 2$.

Критерии оценивания

Упоминается без доказательства, что диаметр делит перпендикулярную хорду пополам — баллы не снимаются.

Доказано, что треугольники NKP_1 и NP_1L подобны — 10 баллов.

Задача 4.2.3.5. (25 баллов)

Темы: теория чисел, комбинаторика.

Условие

Дано натуральное $n = p^k$, где p — простое, а k — нечетное. На доску выписали все натуральные делители n (в том числе 1 и само n). Юра разбил выписанные числа на пары, в каждой паре посчитал произведение чисел и все эти произведения выписал в тетрадку. Оказалось, что все числа в тетрадке имеют одинаковое количество натуральных делителей. Докажите, что у Юры есть ровно один способ такого разбиения делителей числа n на пары.

Решение

Обозначим $d(x)$ — количество натуральных делителей числа x.

Лемма. Количество делителей числа p^m, где p — простое, равно $m+1$.

Доказательство. Делители p^m — это числа $1, p, p^2, \ldots, p^m$. Их ровно $m+1$. Лемма доказана.

У числа $n = p^k$ следующие делители: 1, p, p^2, …, p^k.

Рассмотрим какой-нибудь способ разбиения Юрой делителей числа n на пары. Пусть p^x — делитель, попавший в пару с 1, а p^y — делитель, попавший в пару с p^k. Тогда $d(1 \cdot p^x) = d(p^y \cdot p^k)$. Откуда $d(p^x) = d(p^{y+k})$, значит, $x + 1 = y + k + 1$. Учитывая, что $x \leqslant k$ и $y \geqslant 0$, получим $x = k$, $y = 0$. Т. е. 1 и p^k попали в одну пару. Следовательно, число делителей в каждой паре равно $d(1 \cdot p^k) = k + 1$.

Рассмотрим делитель p^m числа n. К нему в пару попадет такой делитель p^t, что $d(p^m \cdot p^t) = k + 1$ или $m + t + 1 = k + 1$. Откуда $t = k - m$. Учитывая, что k нечетное, получим, что t и m разной четности, и $p^m \neq p^t$, а значит, p^t определяется однозначно.

Разбиение всех делителей на пары вида $\left(d, \dfrac{n}{d}\right)$ удовлетворяет условию задачи, так как в этом случае равны все произведения чисел в парах, а значит, равны и их количества делителей.

Ответ:

Критерии оценивания

Приводится разбиение на пары, удовлетворяющие условию задачи — 5 баллов.

Доказана лемма — 2 балла.

Доказано, что делитель 1 должно быть в паре с числом n — 6 баллов.

Доказано, что если пару составляют делители p^m и p^t, то $m+t = k$ — 12 баллов.

Баллы за указанные выше критерии складываются.

В верном решении не доказывается, что если пару составляют делители p^m и p^t, то $p^m \neq p^t$ — снять 5 баллов.

В решении не приводится разбиение на пары, удовлетворяющие условию задачи — снять 5 баллов.

4.3. Инженерный тур

4.3.1. Общая информация

Энергетика — отрасль для стороннего человека сложная и статичная — на деле стремительно развивается и уже сегодня предлагает множество интересных технологий: от солнечных и ветровых электростанций до микросетей и архитектуры интернета энергии.

Эти и многие другие технологии уже достаточно хорошо изучены, реализованы, но пока еще не встроены в реальные энергетические комплексы, потому что от надежной работы энергосистем в буквальном смысле зависят человеческие жизни. Тем не менее энергетика ближайшего будущего — это тема многих исследований и экспериментов.

Основная задача инженерного тура профиля — моделирование энергосистемы и разработка алгоритмов управления энергообеспечением.

Цель — разработать стратегию оптимального построения умной энергосети и управления ею в условиях локальной конкуренции. Победит та команда, чья энергосеть окажется самой эффективной с точки зрения производства, накопления и расхода энергии, и самой гибкой, способной выбрать оптимальный подход с учетом меняющихся условий.

4.3.2. Легенда задачи

Энергетика — одна из важнейших и сложнейших сфер современной цивилизации, и это возлагает на специалистов-энергетиков большую ответственность. Цена ошибки слишком высока, а любое изменение должно быть тщательно выверено.

Из-за этого кажется, что энергетика инертна и отстает от современности. Но ученые и инженеры постоянно работают над новыми технологиями. Многое из того, что могло казаться фантастикой, уже изобрели, исследовали, а где-то даже построили и используют:

- солнечные и ветряные электростанции,
- накопители энергии для домохозяйств,
- изолированные энергосети, которые обмениваются мощностями и многое другое — лишь малая доля из того, что уже позволяет сократить потери и поставлять стабильную электроэнергию в сложных условиях.

Но почему их не вводят повсеместно? Ответ простой: если собрать все новшества (даже самые проверенные) и внедрить сразу в больших масштабах — энергетика не выдержит и рухнет. Это сложная сфера со множеством взаимосвязей, и часто они совсем не очевидны. Именно для этого нужны ученые и инженеры, способные ставить эксперименты и просчитывать все варианты.

Взять на себя инициативу и заглянуть вперед можно уже сейчас. В инженерном

туре финала участники примерят на себя роль энергосбытовых компаний в энергетике ближайшего будущего, где представлены новые технологии. Стенд «Интеллектуальные энергетические системы» выступает полигоном, готовым к самым смелым идеям и стратегиям для создания наиболее эффективной и гибкой энергосистемы.

4.3.3. Требования к команде и компетенциям участников

Количество участников в команде: 4–5 человека.

Компетенции, которыми должны обладать члены команды:

- **Data-аналитик:** способен работать с данными, анализировать, выбирать те, что наиболее информативны в нужный момент времени. Этот человек может совмещать роль тестера в команде.
- **Системный аналитик:** анализирует стратегию игры и поведение других команд на площадке и на других распределенных площадках, занимается топологией сети. Здесь понадобятся навыки и знания из теории игр, умение комбинировать и искать оптимальные стратегии.

  Совместно оба аналитика должны быть способны определить, какие данные важны для создания алгоритмов управления и что будет являться управляющими параметрами.
- **Программист:** создает алгоритмы управления и вывода необходимых данных, строит обратные связи, работает с реализацией стратегии.

  В команде желательно два программиста, которые способны делать ревью кода друг друга.
- **Капитан:** принимает решения, находясь на слой выше в потоке информации. Должен обладать знаниями по всем основным темам, уметь разбираться в коде, анализировать задачи, строить стратегии. Видит картину в целом, отвечает за постановку цели, за распределение ролей в команде, за быстрое перераспределение задач в случае необходимости, отделяет главное от второстепенного, перенаправляет усилия команды. От работы капитана во многом зависит исход игры.

4.3.4. Оборудование и программное обеспечение

Практический тур ежегодно проводится на стендах-тренажерах «Интеллектуальные энергетические системы» (СТИЭС), разработанных компанией «Полюс-НТ». Это модель поселения с потребителями энергии (жилые дома, больницы, заводы), электростанциями на альтернативных источниках энергии.

Стенд воссоздает погодные условия (ветер и освещенность) и рынок электроэнергии (многоуровневая биржа микроконтрактов). Общий вид комплекса представлен на рис. 4.3.1, устройство комплекса представлено в Приложении 1. Заключительный этап этого года проводится сразу на двух площадках, и конфигурация комплексов на них максимально идентична по характеристикам (в частности,

пространственно-геометрическим).

Рис. 4.3.1. Общий вид комплекса ИЭС

На время инженерного тура каждая команда закрепляется за одним из ПК-терминалов комплекса «Интеллектуальные энергетические системы». Работа идет в программно-аппаратной среде терминала, для нее в этой среде могут быть использованы только предустановленные библиотеки и ПО.

Для анализа данных и формирования стратегий участники могут использовать внешние ноутбуки с любым ПО, которое считают необходимым. Команда может использовать для работы личные ноутбуки, а при их отсутствии — запросить у организаторов ноутбук/ПК.

Таблица 4.3.1

Наименование	**Описание**
Стенды-тренажеры «Интеллектуальные энергетические системы» (СТИЭС)	Основные стенды для проведения финальных соревнований

4.3.5. Описание задачи

Энергосистемы объединены в единую сеть и подсоединены к внешней энергосистеме по схеме «микрогрид»: каждая из энергосистем сначала балансируется собственными ресурсами, а затем — через внешнюю энергосистему, в том числе с помощью оппонентов.

Будущая энергосистема разделена между конкурирующими компаниями. Общий порядок работы заключается в следующем:

1. Каждая команда становится одной из конкурирующих энергокомпаний, строит свою собственную энергосистему и управляет ею. Разделение происходит через цепочку аукционов, в которых участники изначально находятся в принципиально одинаковых условиях.

2. Команды собирают из полученных объектов собственную энергосистему (при этом из одного и того же набора можно составить очень разные по эффективности энергосистемы) и готовят управляющие скрипты (программы на языке Python) для управления ими.
3. Выполняется моделирование нескольких «дней работы» энергосистемы, в ходе которого участники управляют своими энергосистемами посредством скриптов.

Таким образом, команды в течение нескольких сессий:

- анализируют прогнозы погоды и поведения потребителей, распределяют объекты потребления и генерации на аукционе,
- собирают из них энергосеть, настраивают автоматизированную систему управления (путем написания скриптов),
- испытывают ее в ходе моделирования энергосистемы в пробных играх.

Конечной метрикой эффективности разработанной стратегии является прибыль, полученная в результате штатного функционирования энергосистемы. В последние дни финала происходит экспериментальное измерение эффективности построенных энергосистем в соревновательных играх. Очки, набранные командами во время моделирования, пересчитываются в баллы, которые участники получат за командную часть олимпиады.

В этом году физические задачи определения параметров генераторов модифицированы: светильники на столах светятся несинхронно, имитируя движение солнца с востока на запад. Кроме этого, изменен характер прогнозов на силу ветра: они были разными на каждом из трех столов (но на соответствующих столах каждой из двух площадок были одинаковыми). Киберфизические солнечные электростанции, киберСЭС, введенные в прошлом году, сохранены, но незначительно изменена механика их поворота (поменялись скорость поворота и диапазон значений угла).

В самой игре изменен характер экономической модели:

- аукцион ведется по принципу all-pay (платят все) на пакеты объектов с заранее выставленными тарифами,
- в рамках моделирования начисляются экологические баллы, которые в конце игры конвертируются в баллы итогового счета,
- банк для итоговой выплаты формируется из налогов, взимаемых с команд на протяжении моделирования,
- введена механика дискретного износа сетей: в течение моделирования ветки подстанций накапливают износ, при превышении которого ветка отключается на один такт и автоматически включается,
- с помощью управляющего скрипта команды могут вручную отключать ветки для сброса накопленного износа.

Поскольку киберСЭС и ветрогенераторы в такой задаче конкурируют за общее пространство в центральной области стенда, главные подстанции выставлены на аукцион наравне с остальными объектами для того, чтобы команды могли выбирать себе подходящее место. Помимо этого, имелось незначительное влияние светильников соседних стендов, и оно в случае обнаружения его командами было бы монетизировано на аукционе: более выгодные подстанции были бы куплены дороже.

Рис. 4.3.2. Участник устанавливает киберСЭС

Задача командного практического тура заключительного этапа представляет собой комплексную интегральную задачу, полное оптимальное решение которой сложно. Однако полное приближенное решение способна найти любая команда. Качество и сложность используемых приближений отражает глубину понимания, знаний и уровень компетенций участников. Система оценки полностью автоматизирована и спроектирована таким образом, чтобы однозначно и численно оценить качество найденных и реализованных участниками решений.

Проведение заключительного этапа происходит в виде командного турнира. Цель команд — набрать наибольшее число баллов в турнире.

Регламент турнира

1. Подготовка

Этап проводился уже сформированными командами. Это продолжительный период времени, в течение которого участники знакомятся с правилами, изучают предоставленную систему и делают заготовки (управляющие скрипты, стратегии и вспомогательные программы).

Во время подготовки проводится ряд пробных игр как локальных (между команд на отдельной площадке), так и синхронных (среди всех команд на обеих площадках).

2. Финал турнира

Проводятся три синхронных соревновательных игры. В каждой игре вся совокупность принятых участниками решений интегрально оценивается стендом в автоматическом режиме. Каждая игра идет с разными прогнозами, и за каждую игру очки, заработанные участниками, переводятся в рейтинговые баллы. За все три игры результат складывается, после чего вычисляется итоговый балл умножением на 100 и делением на максимальную сумму рейтинговых баллов.

Игры во время командного тура заключительного этапа разделяются на два вида:

- пробные;
- соревновательные.

Все пробные игры предназначены для отработки решений и проведения экспериментов. Никаких баллов за пробные игры не начисляется. Все командные баллы участники получают только за результаты соревновательных игр.

Детальные правила игр 2024–25 года, которые предоставлялись участникам, можно найти в Приложениях 2 и 3.

Этапы одной игры

1. Анализ прогнозов погоды

Минимум за 10 мин до запуска аукциона участникам выдаются прогнозы погоды и потребления для каждой категории потребителей в предстоящей игре. Файл прогнозов выгружается из интерфейса игры в формате CSV и анализируется любым способом на усмотрение участников (с помощью табличного калькулятора или программно). За время анализа участники должны спроектировать энергосистему, наиболее удовлетворяющую предстоящим условиям.

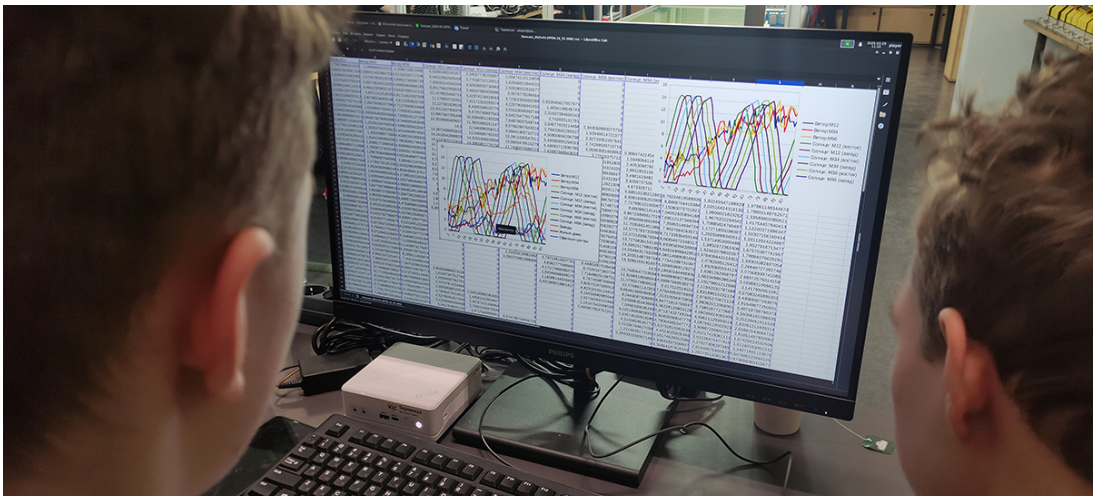


Рис. 4.3.3. Этап анализа прогнозов

2. Основной аукцион

На этом этапе определяется, какой объект к чьей энергосистеме будет подключен. Аукцион закрытого типа, с продолжением в случае догоняющих ставок (отличающихся от лидирующей на заданную правилами величину). Одновременно разыгрываются по три лота. Этот этап практически невозможно пройти успешно без глубокого предварительного анализа и разработанных участниками систем для поддержки принятия решений.

3. Дополнительный аукцион

Проводится через 60 с после основного. В этот интервал времени каждая команда имеет право повторно выставить на торги один из приобретенных ранее пакетов. Данный этап дает командам шанс на исправление ошибок в закупках, если они не слишком существенны.

Рис. 4.3.4. Этап аукциона

4. Проектирование сети и расположение электростанций

Участникам предоставляется время на проектирование топологии сети. За это время они находят оптимальную конфигурацию своей энергосистемы и собирают ее на стенде. В это же время команда адаптирует к получившейся энергосистеме составленные заранее управляющие скрипты.

Рис. 4.3.5. Этап сборки сети

5. Загрузка скрипта в систему стенда

Загрузка осуществляется через пользовательский интерфейс стенда. Написание скрипта производится любым способом на усмотрение команд. Желательно, чтобы скрипт был написан и проверен заранее (для этого участникам предоставляется тестовая библиотека с документацией).

6. Моделирование

Данный этап начинается после подтверждения готовности всех команд путем голосования через пользовательский интерфейс или по поднятой руке на локальных тренировочных играх. Все управление на этом этапе осуществляется только скриптами. Участники наблюдают за работой своих скриптов и могут их заменить, например, в случае обнаружения ошибки. Количество повторных загрузок скрипта не ограничено, во время очередного такта выполняется загруженный на данный момент скрипт.

Рис. 4.3.6. Этап моделирования

7. Оглашение результатов

На синхронных тренировочных и зачетных играх итоговые баллы за моделирование публикуются в чате инженерной задачи. На локальных играх баллы оглашаются устно или выводом табло на экран (в зависимости от технических способностей площадок проведения).

8. Анализ игры

Команды выгружают журнал управляющего скрипта, после чего анализируют его с использованием тестовой библиотеки (принцип и порядок анализа — на усмотрение участников). Кроме того, они анализируют и обсуждают результаты внутри команды, ищут гипотезы и составляют план их проверки.

Регламент аукциона

Аукцион проводится по закрытой схеме. Участники делают ставки в течение заданного отрезка времени, после чего система поводит итог.

В отличие от прошлых лет, на аукционе разыгрывается пакеты объектов с заранее установленными тарифами (для потребителей — цена за 1 МВт $\times$ такт потребленной энергии, для остальных объектов — арендная плата за каждый такт).

Ставки на пакеты делаются на повышение по правилам all-pay (платят все), а именно, в виде фиксированных сумм, которые участники готовы заплатить, чтобы пакет лотов достался именно им, но выплачивают свои ставки, даже если они не выиграли лот. Ставки участников на каждом раунде этого аукциона вычитаются из их результата.

Выигрывает предложивший наибольшую ставку, но если один участник или более предложили ставку, отличающуюся от лидирующей не больше, чем на заданную величину (на зачетных играх она составила 50 руб.), для владельца лидирующей и догоняющих ставок назначался повторный раунд аукциона. При этом ставка all-pay на повторном туре выплачивается отдельно от предыдущей. Стартовая цена каждого лота составляла 1 руб., каждый повторный раунд аукциона повышает стартовую цену в 2 раза (2, 4, 8 и т. д.). Если на лот никто не выставил ставку, он исключается из игры.

Сумма ставок на аукционе all-pay не может превышать 9 999, что контролируется системой управления ходом аукциона, и в случае исчерпания лимита команда автоматически исключается из всех последующих раундов all-pay.

В течение 60 с после окончания аукциона каждая команда имеет право выставить на торги один из своих пакетов. Команда ничего с этого не приобретает (выплаченные ставки не возвращаются), но получает возможность избавиться от лишнего объекта, нарушающего баланс энергосистемы. Она не имеет права делать ставки на объект, который выставила на торги.

На аукционе разыгрываются параллельно по три лота. Порядок выставления лотов фиксирован и известен участникам заранее.

На аукцион выставляются стартовые пакеты, в которые входят:

- главная подстанция,
- мини-подстанция типа А,
- кибер-СЭС,
- жилой дом.

Каждая команда обязана выкупить один из этих пакетов, в противном случае в конце аукциона команды получают один из этих пакетов случайным образом и бесплатно. Покупка двух таких пакетов не предусмотрена и ограничена технически (после выкупа пакета с главной подстанцией команда автоматически исключается из торгов за остальные такие пакеты).

Часть лотов (глобальные) доступна для всех участников аукциона, часть (локальные) — только в пределах одной из двух площадок (для обеих площадок состав объектов и условия были одинаковы).

Локальные лоты представляют собой из инструментов защиты от монополии (закупки избыточного числа объектов одной командой). Само по себе монопольное

поведение не является неспортивным, однако при широком его распространении игра становится вырожденной — выигрышной становится стратегия отказа от участия в аукционе. Она является абсолютно легальной и психологически нетривиальной, но требует решения только части подзадач стенда, поэтому целесообразно, чтобы она была выигрышной только в ситуациях, близких к экстремальным.

4.3.6. Система оценивания

Система оценивания полностью автоматизирована и спроектирована таким образом, чтобы однозначно и численно оценить качество найденных и реализованных участниками решений. Итог отдельной игры отражает интегральную эффективность решений команды в данных игровых условиях в виде условного экономического баланса. Проведение практического тура происходит в виде командного турнира. Цель команд-участников — набрать наибольшее число баллов в турнире.

Турнир представляет собой серию зачетных игр. Каждая игра проводится с разными прогнозами и оценивается по рейтинговому принципу: **две команды с наименьшим итоговым счетом получают 0 баллов, лучшая команда — 100 баллов**, остальные — количество процентов, пропорциональное между лучшим и вторым худшим результатами. Средний рейтинг по финальным играм составляет итоговый командный балл.

Иными словами, за каждую игру очки, заработанные участниками, переводятся в рейтинговые баллы по формуле:

$$p_i = 100\frac{max(m, x_i) - m}{M - m},$$

где p_i — рейтинговый балл i-й команды; x_i — очки i-й команды за игру; M — наилучший из всех команд результат за игру; m — второй с конца результат за игру.

Итоговый командный балл состоит среднего арифметического рейтинговых баллов за три зачетные игры:

$$\text{Итоговый балл} = \frac{1}{3} \times \text{ рейтинг 1-й игры} + \frac{1}{3} \times \text{ рейтинг 2-й игры} + \\ + \frac{1}{3} \times \text{ рейтинг третьей игры}.$$

Итоговый командный балл составляет 60% итогового индивидуального балла участника заключительного этапа. Максимально возможный индивидуальный балл участника заключительного этапа составляет 100 баллов.

Команда-победитель — это команда, набравшая наибольший итоговый командный балл.

4.3.7. Решение задачи

В ходе выполнения финальной задачи оценивается комплексное решение финальной задачи профиля. Выделение и формулирование подзадачи является частью

интегральной (главной) задачи профиля. Проверкой решения подзадачи является ее вклад в результат игры. Описанные ниже задачи выделены разработчиками и известны участникам из описания. Распределение усилий между подзадачами принимают сами команды.

Физика

Задачи из физики присутствовали, однако их физическая составляющая была сведена к минимуму, а физические модели были тривиальными.

Определение взаимосвязи между яркостью освещения и генерацией солнечных батарей

Эта задача возникает на пятом этапе игры, при моделировании энергосистемы. Вырабатываемая мощность солнечных батарей зависит от напряжения на солнечных панелях модели солнечной электростанции на стенде. Напряжение на солнечных панелях по отношению к яркости светильников, строго говоря, нелинейно. Однако характеристики солнечных панелей, измеряющих цепей и светильников подобраны так, что отклонение реальных значений от линейной их аппроксимации составляет не более 2%. В дальнейшем они калибруются до полной линейности внутренним ПО солнечных батарей.

Время релаксации измерительной системы солнечных батарей в два раза меньше минимального интервала между изменением яркости и измерением вырабатываемой мощности, поэтому генерация солнечных батарей зависит только от погоды на текущем такте игры.

В соревнованиях текущего года зависимость генерации от фактической яркости освещения составила определяется по формуле:

$$P(L) = max(0; 0{,}91L + 2{,}81).$$

Коридор отклонений модели от фактических данных (от $-0{,}2$ до $+0{,}3$) составляет менее, чем заложенное в модель случайное отклонение фактических значений яркости освещения от прогнозных (от $-0{,}5$ до $+0{,}5$).

В условиях задачи этого года, когда светильники светят асинхронно (со сдвигом на шесть тактов, но их прогнозы отличаются), у солнечной батареи нет однозначно наилучшего положения и угла наклона.

Расположение СЭС позволяет сдвигать график генерации СЭС, что может компенсировать уменьшение ее общей выработки. В частности, выгодно смещение СЭС к западу, чтобы пик выработки сместился ближе к вечернему пику потребителей третьей категории (и более высокому уровню цен на бирже).

Помимо этого, может быть выгодным разведение используемых СЭС на восток и запад для уменьшения пиковой генерации и, следовательно, уменьшения потерь. Использование киберСЭС позволяет регулировать получаемую генерацию в течение суточного цикла.

В целом, оптимальный выбор решения этой подзадачи зависит от решения остальных подзадач, а также составленной энергосистемы. При этом всегда остается доступным консервативный вариант по максимизации общей выработки энергии.

Определение взаимосвязи между скоростью ветра и генерацией ветряков

Эта задача возникает на пятом этапе игры, при моделировании энергосистемы. Команда должна иметь решение этой задачи для эффективной работы на этапах №№ 1–3 при проектировании энергосистемы и при участии в аукционе.

Рис. 4.3.7. ВЭС

Задача вычисления генерации ветровых электростанций по данным погоды похожа на такую же задачу для солнечных батарей, но является более сложной.

Для получения величины генерации используется скорость вращения анемометра, который установлен на модели ветровой электростанции. Устройство анемометра — вертикально-осевой; его лопасти устроены так, что скорость вращения линейно пропорциональна скорости ветра (если считать поток ветра гомогенным). В случае постоянной негомогенности ветрового потока (когда анемометр не перемещается) скорость вращения анемометра также линейна по отношению к максимальной скорости ветра в потоке.

Измерительная система — инерциальная система вращения с трением. Ее параметры нужно вычислять по данным прошедших игр. Время полной остановки анемометра с максимальной скорости вращения составляет около 5 тактов игры (оно зависит от максимальной скорости ветра в месте расположения анемометра). Время полного разгона аналогично составляет 2 такта игры.

Генерируемая мощность пропорциональна кубу скорости вращения анемометра. При достижении максимального уровня генерации (20 МВт) мощность далее не растет, но при увеличении скорости вращения еще на 30% ветряк уходит в штормовой режим и генерирует 0 до тех пор, пока скорость вращения не уменьшится.

Зависимость генерации от скорости ветра можно оценить большим числом способов на основании данных прошедших игр. Можно как построить собственную модель для каждого ветряка, так и создать общую для всех, со свободным параметром для калибровки модели под каждый конкретный ветряк.

Например, при построении единой модели как линейной комбинации от скорости

ветра за пять последних тактов зависимость будет такой:

$$P(w_0,w_{-1},w_{-2},w_{-3},w_{-4}) = k\frac{(0{,}987w_0 + 0{,}332w_{-1} + 0{,}154w_{-2} + 0{,}099w_{-3} + 0{,}051w_{-4})^3}{1008},$$

где w_{-n} — скорость ветра n тактов назад, а коэффициент $k(0;1]$ подбирается для каждого ветряка индивидуально по прошлым данным (знаменатель дроби подобран так, чтобы для наилучшего ветряка коэффициент k составлял единицу).

При вычислениях необходимо учитывать тот факт, что генерация ветровых электростанций ограничена правилами на уровне 20 МВт. Поэтому если прогнозируемая генерация превышает предельный уровень, то надо считать, что спрогнозировано именно на уровне предела.

Средняя величина ошибки такого набора коэффициентов между прогнозируемой и реальной генерацией на играх финала составила 5,9%, что, с одной стороны, связано с большой инерционностью физического анемометра. С другой стороны, из-за кубической зависимости генерации от силы ветра любые погрешности в средней части диапазона скоростей ветра очень сильно влияют на прогнозируемую мощность. Если скорость ветра мала или велика, погрешности влияют очень слабо.

Проектирование энергосистемы

Эта задача возникает на четвертом этапе игры, при сборке сети, или даже на этапах №№ 1–3, если участники системно подходят к задачам проектирования энергосистемы. В ней имеющиеся объекты нужно распределить по энергосистеме с использованием подстанций, дополнительных модулей и с учетом ограничений правил.

Цель проектирования сложна и зависит от участников. Ее возможные компоненты:

- Минимизация суммарных потерь (и связанных с ними штрафов).
- Минимизация стоимости обслуживания инфраструктуры.
- Максимизация доходов от потребителей.
- Минимизация убытков от регулярных ремонтов ЛЭП.
- Накопление энергии с целью продажи в период высоких цен на внешней бирже.
- Набор экологических баллов для получения большей доли банка экологических налогов.
- Злоупотребление механикой начисления экологических баллов за использование накопителей путем циклического перекачивания энергии между ними.
- Минимизация потерь через активное маневрирование мощностью при помощи ТЭС и накопителей.
- Возможность решить эту задачу быстро для перепроектирования энергосистемы в реальном времени, то есть сразу в течение аукциона по мере закрытия лотов.

Это — одна из интегральных задач стенда, в которой собираются все остальные (вторая такая — аукцион). Для нее существует множество решений, близких к оптимальному. Задача точного поиска глобального экстремума довольно сложна, однако нахождение локального экстремума, который незначительно отличается от глобаль-

ного, является не только разрешимой, но и используется для калибровки решений, предлагаемых участниками.

Каждая команда справляется с ней тем или иным способом: как через написание экспертной системы для автоматической оптимизации сети, системы прогнозирования и визуализации, так и вручную на листе бумаги.

Это задача, которую очень легко решить удовлетворительно (чтобы была возможность работать с остальными задачами), можно решить хорошо (чтобы получить минимальное преимущество перед другими командами), можно решить полностью, но только в рамках выбранного командой сужения задачи.

Математика

Вычисление полного энергетического баланса на основании данных прогнозов

Задача возникает на первом и пятом этапах игры, при анализе прогнозов и моделировании энергосистемы.

В течение всей игры командам доступны все данные о прогнозах погоды и действующих контрактах. Из них можно вычислить прогноз дефицита или профицита мощности на каждый такт. Для этого нужно на основании составленных заранее игроками моделей вычислить из прогнозов погоды прогнозы генерации. Из результирующего множества случайных величин (вероятная генерация для каждой электростанции и вероятное потребление для каждого потребителя) нужно вычислить их сумму. Она будет представлять собой распределение вероятностей профицита/дефицита мощности в системе. Важно учитывать, что максимальный дефицит или профицит мощности ограничен главной подстанцией и установленными на ней объектами.

В задаче текущего года, в отличие от прошлых лет, предельная возможная сложность энергосистем намного выше, поэтому приведенное ниже решение достаточно эффективно работает только для достаточно простых систем. В более сложных системах может понадобиться дополнительный учет потерь, который сильно зависит от конфигурации сети. Это приводит к тому, что задача становится значительно более интегрированной в систему поддержки принятия решений, разработка которой необходима для успешной работы, и которая формируется у каждой команды в том или ином виде — программном или таблично-вычислительном.

Вычисление баланса энергорайонов энергосистемы

Каждая энергосистема состоит из набора подстанций и подключенных к ним объектов. Эффективная энергосистема представляет собой граф-дерево, в узлах которого находятся подстанции, а ребра — энергорайоны с множеством подключенных объектов и, что очень важно, двумя узлами подстанций, к которым подключен энергорайон (причем, если энергорайон физически подключен только к одной подстанции, то второй узел виртуален).

По правилам мощность, протекающая через один узел, увеличивает износ линии, тем самым повышая уровень потерь в линии. Это приводит к тому, что необходимо прогнозировать энергобаланс в каждом энергорайоне по отдельности (пол-

ностью аналогично нахождению общего баланса) и учитывать складывание токов в энергосистеме при протекании их по дереву.

Вычисление экономического баланса энергосистемы на основании данных прогнозов

Задача возникает на первом и пятом этапах игры, при анализе прогнозов и моделировании энергосистемы. Команда должна иметь ее решение для эффективной работы на этапах №№ 2–3 — участии в аукционе.

Далее нужно решить задачу нахождения вероятностного распределения экономических потерь. Это делается почти тривиально при принятии консервативной оценки прибылей и убытков от взаимодействия с внешней энергосистемой: каждый мегаватт непредсказанной избыточной мощности эффективно несет убыток в 1 очко, а каждый мегаватт недостаточной — 2 очка. К этим значениям нужно добавить стоимость электроэнергии, закупленной/проданной во внешней энергосистеме по цене за 1 такт вперед.

Далее математическое ожидание получившейся случайной величины нужно минимизировать, используя параметры направляемой/извлекаемой мощности из аккумуляторов и закупаемой/продаваемой мощности во внешней энергосистеме. Выгоды от торговли с другими командами и ранних закупок во внешней энергосистеме можно учитывать независимо.

Эта задача немного проще, чем кажется, из-за того, что использование аккумуляторов экономически намного выгоднее взаимодействия с внешней энергосистемой, имеет смысл закупать/продавать электроэнергию в ней только в случае невозможности сделать это через аккумулятор. Оба аспекта образуют тандем, который можно обозначить как балансирующее воздействие на энергосистему. Иными словами, сначала решается задача вычисления дисбаланса (для удовлетворительного решения достаточно приблизительного прогноза или даже фактического значения), который балансируется сначала аккумуляторами, после — заявкой на биржу.

Задачу (нахождение балансирующего воздействия, минимизирующего математическое ожидание экономических потерь) недостаточно решить аналитически — алгоритм решения нужно реализовать также в управляющем скрипте (и других вспомогательных программах), что для большинства школьников является нетривиальной и неустойчивой к ошибкам задачей. Немного проще ее решить перебором, перебрав все значения балансирующего воздействия в размахе случайного распределения дефицита/профицита мощности в системе с фиксированным шагом, например, 0,1. Такой точности будет вполне достаточно, и такое решение можно реализовать быстрее, чем точное. Впоследствии, при наличии времени, его можно заменить точным.

В дальнейшем на основании этих данных можно вычислить распределение вероятностей прибыли за всю игру.

Использование накопителей

Накопители имеют три основные сферы применения:

- Накопление выработанной внутри энергосистемы энергии для более позднего

ее использования без взаимодействия с внешней энергосистемой. Это обычно выгодно, поскольку сначала продать энергию вовне, а затем позже ее купить практически всегда ведет к издержкам.

- Уменьшение потоковой мощности на линиях энергосистемы для уменьшения электрических потерь. Зависимость величины потерь от мощности нелинейна и сильно возрастает с увеличением мощности. Потерянную мощность не только нельзя продать, но она обкладывается дополнительными штрафами.
- Смещение взаимодействия с внешней энергосистемой во времени. Цены на ней непостоянны, поэтому способность временно задерживать электроэнергию в системе позволяет продавать ее по более выгодной цене.

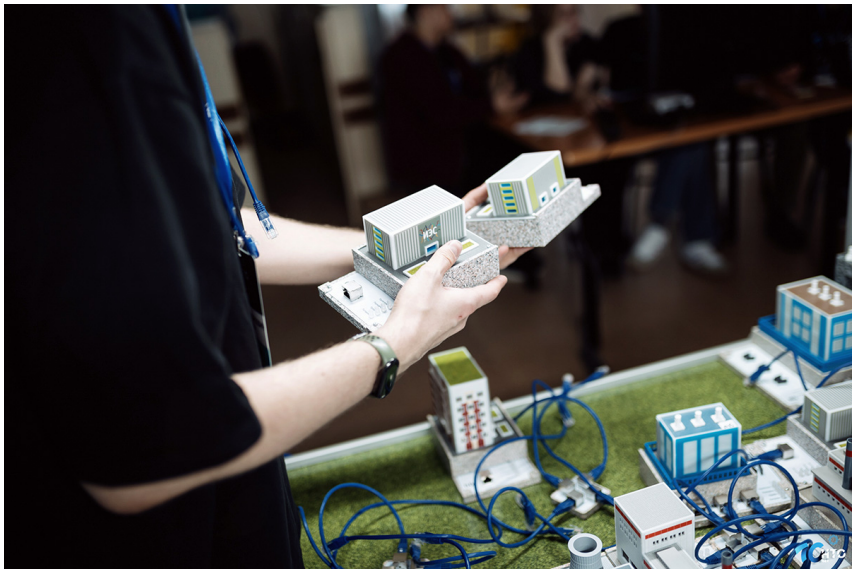


Рис. 4.3.8. Накопители

Кроме того, за использование накопителей щедро начисляются экологические баллы, что позволяет злоупотреблять ими, перекачивая энергию из одного в другой и, тем самым, получая баллы за один и тот же мегаватт несколько раз.

Расчет предельной цены объекта

Задача возникает на втором и третьем этапах игры при участии в аукционе.

Это задача нахождения предельной цены контракта объекта на аукционе — такой цены, приобретение контракта по которой ожидаемо не принесет ни прибылей, ни убытков. Для расширения над задачей нахождения полного экономического баланса энергосистемы достаточно вычислить суммарную прогнозируемую прибыль энергосистемы с этим объектом и без него.

Для электростанций разницу между этими величинами нужно разделить на число тактов в игре. Для потребителей — разделить на суммарное потребление его за всю игру (это случайная величина; для примерной оценки ее можно заменить на математическое ожидание, но предполагается, что команды к этому этапу накопили достаточную экспертизу и опыт вычислений со случайными величинами).

Получившееся число обозначает:

- для электростанций — максимальную осмысленную цену,

- для потребителей — минимальную.

С учетом того, что все объекты одного класса сделаны аналогичными, ее расчет можно сделать и без программирования, средствами электронных таблиц Excel или их аналогов.

В текущем году торги проводятся пакетно: каждый лот содержит по несколько объектов, обычно два–три (лоты с главными подстанциями содержали четыре: одну поворотную СЭС, одного потребителя третьей категории, одну миниподстанцию и собственно главную подстанцию. Такая механика смещает фокус с собственной цены объекта на ценность добавления к энергосистеме группы объектов. Это не меняет фундаментального свойства аукциона — комплиментарности объектов друг другу, вроде убыточности генерации при недостатке потребления, но делает его расчет из важного для победы в необходимый для участия.

Аукцион проводится по системе all-pay: участники платят сделанную ставку, даже если лот они не выиграли. Это также смещает фокус их внимания с произвольного выставления ставок к предварительному составлению стратегии ставок.

Составление и адаптация стратегии для аукционов

Задача возникает на втором и третьем этапах игры при участии в аукционе.

Хотя команды могут найти оптимальную энергосистему для любого набора прогнозов погоды, эту энергосистему им еще нужно собрать, обыгрывая конкурентов за каждый необходимый лот на аукционе. Игра в целом относится к рефлексивным. Игры этого класса не имеют устойчивого решения, а использование смешанных стратегий едва ли оправдано на одиночном сеансе.

Участникам нужно анализировать поведение оппонентов, предугадывать их цели и искать способы помешать целям оппонентов, при этом защитив свои. Для этого можно создать несколько вспомогательных инструментов:

1. Нахождение эффективных конфигураций энергосистем — не только оптимальной, но всех, которые достаточно хороши. Все команды будут стремиться к какой-то из них, и для любой промежуточной ситуации на аукционе можно предположить, к какой из них будут стремиться оппоненты.
2. Нахождение оптимальной ставки на аукционе, исходя из ценности лота для себя и оппонентов. Ценности лота для оппонентов можно оценить из предположения об энергосистеме, к которой они стремятся, с использованием решения задачи расчета прогноза рентабельности. В этих условиях оптимальной ставкой будет максимальная ставка всех оппонентов, при условии, что она не превышает собственной максимальной ставки. Далее участникам будет интересно от этой ставки отклониться, чтобы рискнуть и увеличить выгоду, либо нарушить стратегию оппонентов.

Также в течение всего времени аукциона командам нужно оценивать риск того, что целевую энергосистему составить не удастся и, при необходимости, переходить к альтернативным вариантам.

Работа с биржей электроэнергии

Эта задача, тривиальная сама по себе, значительно углубляет задачи вычисления экономического баланса энергосистемы и составления стратегии для аукциона.

Механика аукциона устроена следующим разом: те, кто устанавливают цены заявок хоть чуть-чуть хуже для себя, чем их оппоненты, получают преимущество и, потенциально, более выгодную цену. С другой стороны, в случае выставления на биржу дефицитного товара, сбивание цены приведет исключительно к потере очков, без получения иных преимуществ.

Более того, механика устроена таким образом, что не взаимодействовать с ней участники не могут, поэтому эту задачу можно во многом обойти, если спроектировать энергосистему так, что она всегда будет предлагать на биржу дефицитный товар (товар здесь — избыток или недостаток энергии, иначе говоря, генерация или потребление). В целом задача сводится к прогнозированию того, будет на бирже в дефиците спрос или предложение и, в случае размещения недефицитного товара, подстраивания цены под прошлое поведение оппонентов.

В модели текущего года на бирже присутствует предзаполнение автоматическими заявками, которые не принадлежат игрокам и по легенде представляют участников торгов из внешней энергосистемы. Они позволяют смягчить изменение цены при общей несбалансированности энергосистем игроков (энергоизбыточности или энергодефицитности), но требуют учета этих заявок при выборе цены собственной заявки. Эта механика подавляет жадное поведение, одновременно делая механику биржи менее катастрофической для ее полноценных участников.

Задача возникает на пятом этапе игры при моделировании энергосистемы.

Информатика

Система поддержки принятия решений на аукционе

Это первая точка сборки решений предметных задач, возникающих в командном туре. Основная задача таких систем — вычисление ценности лота при заданных параметрах энергосистемы.

Минимальный полезный вариант такой системы может быть устроен следующим образом:

1. Имеются прогнозы погоды и потребления на следующую игру (будем считать, что реальные значения будут точно соответствовать прогнозам).
2. Для каждого такта игры вычисляем генерацию. Например, солнечные батареи вычисляем из яркости солнца с найденным ранее коэффициентом конверсии солнечной энергии в мощность, например, 1 МВт / 284 лк.
3. Вычисляем энергобаланс в каждый такт игры.
4. Вычисляем изменение счета в каждый такт игры.
5. Добавляем в энергосистему интересующий объект.
6. Повторяем шаги 1–4.
7. Сравниваем результаты для энергосистемы при наличии и без наличия интересующего объекта:

- для электростанций оптимальная цена есть их цена плюс разница результатов энергосистем, деленная на число тактов игры;
- для потребителей оптимальная цена есть их цена плюс разница результатов игры, деленная на потребленную потребителем энергию.

Хорошую систему от примитивной могут отличать следующие характеристики:

- Учет погрешностей прогнозов. Расчет наихудшего варианта, наилучшего, наиболее вероятного и других статистических характеристик.
- Расчет стоимости лота для энергосистем оппонентов — чтобы выиграть лот, нужно ставить не собственную цену, а цену, большую, чем у оппонентов. Для этого необходимо оценивать ценность лота для оппонентов.
- Проверка корректности модели и ее параметров путем сравнения фактических ставок оппонентов с предсказанными.
- Прогноз эффективности задуманной энергосистемы.

Важно, что такая система лишь помогает принимать решения, но не гарантирует того, что команда с наилучшей реализацией этой задачи наиболее эффективно проведет аукцион.

Разработанные командами программы варьировались по сложности от простых таблиц Excel до веб-сервисов с элементами машинного обучения.

Управляющие скрипты

Это вторая точка сборки предметных задач, возникающих в командном туре. Задача управляющего скрипта — используя возможность управления продажей электроэнергии во внешнюю энергосистему, управления накопителями, прогнозирования генерации, максимизировать число очков, которое получит команда за командный тур.

Неработающий (или работающий плохо) управляющий скрипт значительно ухудшит результаты команды вне зависимости от того, насколько хорошо они сыграли аукцион и проработали предметные задачи командного тура.

Самый простой вариант скрипта может действовать, например, по следующему алгоритму:

1. Повернуть киберСЭС на угол, указанный в заранее вычисленной таблице значений.
2. Оценить генерацию на следующем такте. Вычислить энергобаланс следующего такта.
3. ЕСЛИ энергобаланс положителен, ПЕРЕЙТИ к п. 6.
4. Попытаться ликвидировать дефицит из накопителей.
5. Ликвидировать дефицит из внешней энергосистемы.
6. ЗАВЕРШИТЬ РАБОТУ.
7. Попытаться ликвидировать профицит, перенаправив мощность в накопители.
8. Ликвидировать профицит, продав мощность во внешнюю энергосистему.
9. ЗАВЕРШИТЬ РАБОТУ.

Хороший скрипт может обладать следующими характеристиками:

- Оценки энергобаланса на всю игру вперед и ранняя закупка электроэнергии.
- Коррекция прогнозов генерации на основании реальных данных от электростанций.
- Использование распределения вероятных значений энергобаланса, чтобы вычислять средневзвешенную величину его коррекции: дефицитный энергобаланс обходится дороже профицитного, соответственно, нужно минимизировать не модуль энергобаланса, а математическое ожидание экономических потерь в результате ошибок прогнозов.
- Коррекция заложенных в модель параметров генерации СЭС и ВЭС с учетом наблюдаемых значений генерации.
- Такое управление накопителями, которое полностью разряжает их к последнему такту игры.
- Такое управление накопителями, которое злоупотребляет начислением экологических баллов, при этом продолжая нормальное использование накопления.
- Моделирование состояния энергосистем оппонентов для предсказания будущих состояний биржи.
- Вычисление графика отключений, минимизирующего штрафы за отключение потребителей и исключающие отключения ТЭС и заводов.
- Предсказание графиков отключений оппонентов и влияние их на наполнение биржи заявками.
- Уточнение моделей энергосистем оппонентов на основании реального состояния биржи.
- Управление риском: в ряде случаев осмысленно использование неоптимальных характеристик управления, которые несмотря на то, что они снижают математическое ожидание счета в командном этапе, увеличивают вероятность обойти другую команду.

4.3.8. Материалы для подготовки

Курсы от разработчиков профиля:

1. Курс «Интеллектуальные энергетические системы»: https://onti.polyus-nt.ru/course/view.php?id=2.
2. Курс «Тематические разборы задач профиля ИЭС»: https://onti.polyus-nt.ru/course/view.php?id=4.
3. Курс «Разбор задач второго этапа профиля ИЭС НТО 24/25»: https://onti.polyus-nt.ru/course/view.php?id=24.

Материалы, подобранные по темам задач:

1. Курс «Теория игр» от Школа «Интеллектуал» и проекта «Дети и наука»: https://childrenscience.ru/courses/math_games/.
2. Курс МФТИ «Теория игр»: https://courses.mipt.ru/course/view/83.
3. Курс «Теория вероятностей — наука о случайности»: https://stepik.org/course/2911/promo.
4. Книга «Вероятность: примеры и задачи», автор А. Шень: https://old.mc

cme.ru//free-books//shen/shen-probability.pdf.
5. Курс Андрея Райгородского «Теория вероятностей», плейлист: https://www.youtube.com/playlist?list=PLthfp5exSWEr8tRK-Yf-i9aXgcFJ-O16d.
6. Курс «Основы теории графов»: https://stepik.org/course/126/promo.
7. Курс «Основы дискретной математики»: https://stepik.org/course/1127/promo.
8. Статья «Численные методы решения нелинейных уравнений»: https://prog-cpp.ru/digital-find/.
9. Книга «Код: тайный язык информатики», автор Чарльз Петцольд: https://www.livelib.ru/book/1000005181-kod-tajnyj-yazyk-informatiki-charlz-pettsold.
10. Курс «Программирование на Python»: https://stepik.org/course/67/syllabus.
11. Курс «Программирование на Python для решения олимпиадных задач»: https://stepik.org/course/66634/promo.
12. Курс «Python: основы и применение»: https://stepik.org/course/512/promo.
13. Курс «Основы машинного обучения»: https://openedu.ru/course/hse/INTRML/.
14. Курс «Python для извлечения и обработки данных»: https://openedu.ru/course/hse/PYTHON/.

4.3.9. Приложение 1. Устройство комплекса «ИЭС» 2024–25 года

Стенд-тренажер «ИЭС» 2024–25 года состоит из следующих основных частей:

1. Модельная поверхность, на которой располагаются модели объектов энергосистемы: электростанций, потребителей и объектов энергетической инфраструктуры.
2. Терминалы управления энергосистемой. Это персональные компьютеры, объединенные со стендом в единую информационную сеть. Каждая команда работает со своим терминалом.
3. Модели объектов энергосистемы — небольшие стереотипные архитектурные модели, содержащие в себе необходимую управляющую электронику и измерительные системы.
4. Светильники, моделирующие солнечное освещение. Они способны создавать освещенность в центре стола до 5 клк.
5. Мощные вентиляторы, моделирующие ветровые условия, которые могут создавать ветер со скоростью до 8 м/с на расстоянии не менее 1 м от плоскости вращения.
6. Управляющая электроника стенда. Эта часть скрыта от участников и, с их точки зрения, не несет функциональной нагрузки.

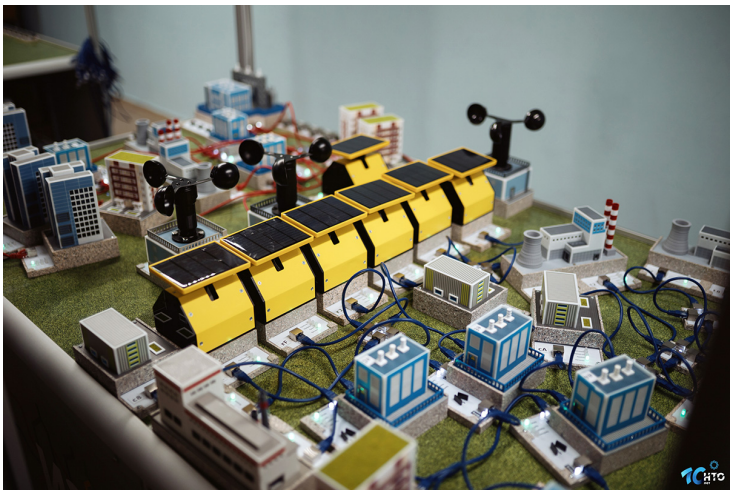


Рис. 4.3.9. Вид объектов стенда ИЭС вблизи

Рис. 4.3.10. Интерфейс аукциона во время игры

На рис. 4.3.11 показан интерфейс моделирования. Внизу находятся графики энергетического и экономического баланса, вверху — состав энергосистемы, слева — графики прогнозов погоды и потребления, справа — принятые приказы и итоговый счет. Остальные поля являются вспомогательными.

Рис. 4.3.11. Интерфейс моделирования

На рис. 4.3.12 показан интерфейс скрипта. Слева находится справка, в центре-слева редактор скрипта, в центре-справа загруженный скрипт, справа — стандартный вывод и вывод ошибок скрипта.

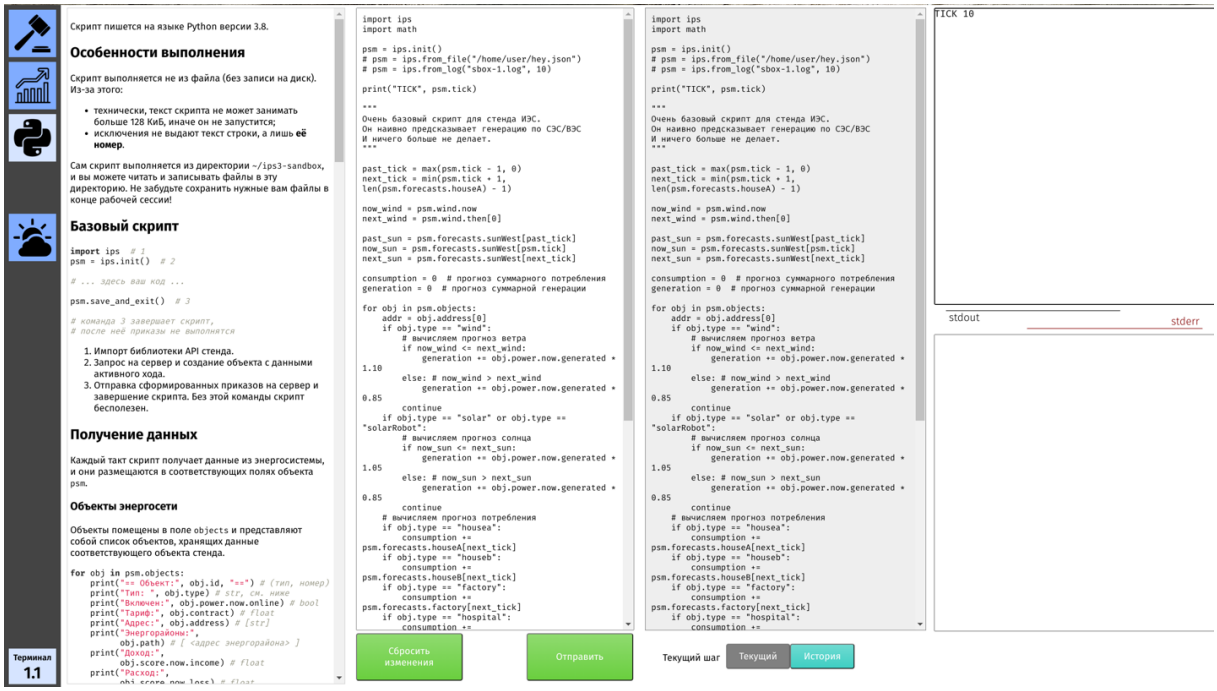


Рис. 4.3.12. Интерфейс скрипта

4.3.10. Приложение 2. Правила командной задачи, выдаваемые участникам

Структура игры

Члены команды проектируют энергосистему в конкуренции с другими командами. Кто каких потребителей себе подключит и какие электростанции установит, решается через аукцион. На аукционе разыгрываются наборы объектов с фиксированным тарифами, и торг идет на повышение по принципу all-pay (все ставки вычитаются из бюджета и соответственно из результата игры соответствующей команды).

После аукциона участники проектируют сеть, решают задачу оптимизации потерь в энергосистеме и увеличения ее надежности тем, как происходит подключение объектов. Здесь инструменты — подстанции и миниподстанции.

Затем проводится моделирование энергосистемы, в результате которого определяется то, сколько денег она принесет — это и есть счет команды. В этом этапе также участвует скрипт, который реагирует на происходящее в энергосистеме и предпринимает корректировки.

Обратите внимание, что один только скрипт игру точно не выиграет, но без хорошего скрипта за победу бороться трудно.

Энергосистема команды и энергосистемы конкурентов объединены через внешнюю энергосистему — гарантирующего поставщика. Ему и через него можно продавать (или покупать) энергию, в случае дисбаланса в энергосистеме.

Рекомендации по решению

Рекомендации не являются инструкцией. Задачу можно решать и по-другому, они приведены для того, чтобы участникам было, с чего начать. Скорее всего, все команды будут решать задачу иначе, но отличия от данных рекомендаций у всех будут разные.

Важное свойство поставленной задачи — она не решается без серьезной подготовки. Более того, ее крайне сложно решить полностью за имеющееся время (но команде достаточно решить ее лучше оппонентов), поэтому крайне важным становится распределение усилий. Допустим, что задача состоит их трех подзадач; команда, решившая каждую на 50%, 50% и 50% выиграет у команды, решившей на 100%, 100% и 0%, со значительным отрывом.

Далее участникам предлагаются рекомендации по этапам игры (еще раз повторим, что это рекомендации, а не предписания!):

1. Анализ прогнозов и определение стратегий. Важно определить, на каком фронте состязания команда будет превращать преимущество перед оппонентами в лидировании в результатах. Как именно она станет это выполнять — дело ее членов, но без программирования и электронных таблиц вряд ли можно обойтись.
2. Аукцион. Он требует, исходя из выбранной стратегии, знания ценности для каждого из торгуемых объектов. Кроме того, он требует дисциплины в команде. Если участники будут отвлекаться на споры и решение технических вопросов, то в ставках, скорее всего, станут ошибаться.
3. Проектирование сети. Проект, скорее всего, будет составлен еще при составлении стратегии, но результаты аукционов могут заставить внести в него правки. Сеть члены команды собирают физически на столе вокруг своей главной подстанции.
4. Загрузка скрипта. Скрипт команды должен быть написан и проверен заранее (но понятно, что тестовые игры нужны именно для тестирования), внести в него исправления на ходу вряд ли получится безошибочно. Поэтому настоятельно рекомендуем перед зачетными играми не вносить в скрипт никаких изменений, насколько бы тривиальными и простыми они ни казались.
5. Анализ и обсуждение результатов внутри команды, поиск гипотез и составление планов их проверки. Это самый важный этап задачи, и победа будет зависеть от него.

Прогнозы

Прогнозы составляются для солнца, ветра и каждого типа потребителей.

Прогноз выдается на каждый такт (48 тактов — одни сутки). Для каждого такта он состоит из единственного числа — центра коридора, в который попадет реальное значение прогнозируемой величины. Размеры коридоров приведены в таблице 4.3.2.

Таблица 4.3.2

Прогноз	**Коридор**
Солнце	0,5[1]
Ветер	Прогноз точный
Заводы	0,5[2]
Дома	0,5[3]
Офисные центры	0,5[4]

Значение в 5,2 с коридором в 0,5 означает, что реальное значение будет лежать в интервале от 4,95 до 5,45.

У каждого типа объектов прогнозы свои, и все объекты одного типа полностью идентичны и в прогнозах, и в реальных значениях.

Прогнозы для ветра ведут себя немного иначе, чем остальные. Они отдельны для каждого вентилятора, при этом они похожи, хоть и не одинаковы.

Потребители

Потребители делятся на три типа:

- заводы,
- дома,
- офисные центры.

Они отличаются потреблением и паттернами потребления.

Если ТЭС или завод подключены обоими входами, то их переток мощности распределяется равномерно по обоим входам. Если произошло отключение по одной из линий подключения, то объект запитывается от второй и его нагрузка полностью ложится на нее.

Каждый потребитель имеет тариф — цену, которую он платит за каждый МВт × × такт поставляемой ему мощности. Чем тариф больше, тем потребитель выгоднее.

Штрафы за отключения

За полное отключение потребителей предусмотрены штрафы.

Первые пять[5] отключений объекта третьей категории (дома и офисные центры) не штрафуются. Каждое следующее штрафуется в размере 3[6] руб. за 1 МВт мощно-

[1] Константа конфига `corridorSun`. Далее для всех числовых значений будут приводиться названия их констант, значения которых скрипт может извлечь из конфига игры. Рекомендуем в скрипте опираться на них, чтобы уменьшить число «магических чисел».

[2] Константа `corridorClass2`.

[3] Константа `corridorClass3A`.

[4] Константа `corridorClass3B`.

[5] Константа `class3FinePardon`.

[6] Константа `class3Fine`.

сти, которую объект потребил бы на том такте, в котором он находился в отключенном состоянии.

Каждое отключение объекта второй категории (заводы) штрафуется в размере 5[7] руб. за 1 МВт непотребленной мощности.

Электростанции

Существуют электростанции трех типов:

- тепловые,
- ветровые,
- кибернетические солнечные (киберСЭС).

Ветровые и киберСЭС — это реальные физические измерительные системы, и их расположение на стенде очень важно. Тариф электростанции — это цена, которую нужно платить каждый такт за ее обслуживание. Чем он меньше, тем электростанция выгоднее.

ВЭС

Генерируемая ветряками мощность пропорциональна кубу скорости ветра (точнее, частоте их вращения, что не одно и то же). Есть предельная скорость их вращения[8] (около 8 об./с), при достижении которой ветряк отключается и уходит в «штормовую защиту», из которой выходит только при снижении его скорости до 87,5% от этой скорости[9]. Кроме того, максимум генерации достигается при 86%[10] от предельной скорости и далее не растет до самого отключения. Обратите внимание, что скорость вращения ветряка при одном и том же ветре очень сильно зависит от его расположения. Максимальная мощность ВЭС — 20 МВт[11].

Физическая инерция ветряков весьма велика и может составлять до 30 с (это не программируемое значение, а физическое). Более точные данные участникам нужно получить самим.

КиберСЭС

Это солнечные панели, которые могут изменять угол наклона по командам от управляющего скрипта.

Диапазон их наклона составляет около 100°, участники могут устанавливать наклон с градацией в 125 руб., от 0 до 124 включительно. Значению 0 соответствует предельный наклон назад (от разъемов), 124 — предельный наклон вперед (к разъемам). Скриптом участники отдают команду на установку определенного положения, а далее СЭС сама поворачивается к этому углу.

[7] Константа `class2Fine`.

[8] Константа `windBreakValue` — это доля от максимального значения, которое могут выдавать измерительные цепи ветряков.

[9] Константы `windRecoveryValue/windBreakValue`.

[10] Константы `windSummit/windBreakValue`.

[11] Константа `maxWindPower`.

Скорость вращения киберСЭС ограничена и не слишком велика, поэтому если дать команду для изменения угла на большое значение, то поворот может занять более одного такта игры. Впрочем, она гарантированно успевает повернуться на 20 единиц.

Максимальная мощность киберСЭС — 25 МВт[12].

ТЭС

Тепловая электростанция проще, чем СЭС и ВЭС в том, что она не имеет физической модели, однако это не обязательно означает, что с нею легче управляться.

На каждом такте ей нужно указывать, какой объем топлива следует сжигать. За топливо команда платит отдельно, сверх ее собственного тарифа. 1 единица топлива превратится в 1 МВт мощности с учетом КПД и с добавлением инерции: к выработке сжиганием топлива добавится часть мощности, выработанной на предыдущем такте. Точнее, 60%[13] от предыдущей мощности, уменьшенной на 0,5[14] МВт.

Единица топлива стоит 0,5 руб[15]. При КПД в 100% ТЭС перерабатывала бы 1 единицу топлива в 1,2 МВт × такт за первый такт[16], однако ее КПД не только не идеален, но и не линеен. Он представляет собой параболу, максимум которой в 45%[17] достигается при сжигании 12 единиц топлива[18], а при сжигании 0 единиц топлива КПД составляет всего 20%[19]. Предельный объем сжигаемого ТЭС топлива — 20 единиц[20].

Например, на прошлом такте ТЭС вырабатывала 10 МВт, на текущем ей указано сжечь 12 единиц топлива. Мощность на текущем такте составит: $12 \cdot 0{,}45 + 0{,}6 \cdot (10 - 0{,}5) = 5{,}4 + 5{,}7 = 11{,}1$ МВт.

ТЭС имеют две точки подключения. Как и у заводов, если обе они активны, то мощность между ними распределяется поровну.

Накопители

Накопители разыгрываются на аукционе наравне с потребителями и электростанциями. Тарифицируются они так же, как электростанции — фиксированная плата за каждый такт.

Емкость накопителя составляет 80 МВт × такт[21], предельная скорость заряда — 15[22] МВт × такт, разряда — 20[23] МВт × такт. Зарядом и разрядом накопителей управляет скрипт.

---

[12] Константа `maxRobotPower`.
[13] Константа `tpsInertia`.
[14] Константа `tpsInertiaFriction`.
[15] Константа `tpsFuelCost`.
[16] Константа `tpsFuelPower`. С учетом инерции вместе с последующими тактами будет выработано больше.
[17] Константа `tpsEcePeakValue`.
[18] Константа `tpsEcePeakPower`.
[19] Константа `tpsEceZeroValue`.
[20] Константа `tpsMaxPower`.
[21] Константа `cellCapacity`.
[22] Константа `cellChargeRate`.
[23] Константа `cellDischargeRate`.

Кроме того, накопители имеют потери. Каждый такт накопитель теряет 5%[24] содержащейся в нем энергии.

Монтаж сети

Подстанции и аукцион

Сеть строится вокруг главной подстанции — ее обязательно нужно приобрести на аукционе. Какую именно — нужно решать на основании прогнозов, собственной стратегии команды и стратегии оппонентов. Не приобрести ее нельзя — тогда команда не сможет играть. Поэтому если по завершении аукциона участники окажутся без подстанции, пакет с ней будет выдан случайным образом среди неразыгранных на площадке.

Во время аукциона можно приобрести только одну подстанцию с пакетом объектов, пакеты с другими подстанциями после этого будут недоступны во время аукциона.

Проектирование сети

Сеть энергосистемы строится при помощи подстанций — главных и миниподстанций А (с тремя выходами). Главные подстанции и миниподстанции разыгрываются на аукционе. Если команда приобрела миниподстанцию, то обязана ее установить.

Миниподстанции должны быть подключены входом в сторону главной подстанции.

От любого объекта должно быть можно пройти по проводу до главной подстанции, иначе до него по-настоящему не дойдет электричество.

Кольца и острова в энергосистеме создавать недопустимо, поэтому топология энергосистемы — дерево. Энергосистема ветками (выходами и входами) подстанций разбивается на энергорайоны, внутри которых находятся потребители или электростанции. Энергорайон — это область «от подстанции до подстанции».

Электростанции и потребителей подключать в один энергорайон нельзя, они должны быть разделены подстанциями.

Ветряки и солнечные электростанции устанавливаются в порядке их адресов независимо от порядка покупки. «КиберСЭС по умолчанию» (в пакете с главной подстанцией) устанавливаются после киберСЭС, купленных в остальных пакетах.

Если объект выигран, но не подключен, или подключен другой командой, то это такая же ошибка монтажа сети, как и смешение генераторов и потребителей в одном энергорайоне. В этом случае игра не начинается. Исключения из этого правила возможны только в случае тренировок, если все затронутые команды согласны продолжить без исправления ошибки с целью сэкономить время. На зачетных играх это запрещено.

[24] Константа `cellLeak`.

Биржа

В случае, если энергосистема не сбалансирована полностью (а так будет практически всегда), недостаток или избыток мощности ликвидируется через внешнюю по отношению к игроку энергосистему. Есть два способа это сделать:

1. Заранее подать заявку на продажу или покупку нужной мощности. Через один ход (не на следующий!) заявка попадет в «биржевой стакан» (про него чуть ниже) и превратится в мощность и деньги.
2. Ничего не делать. Тогда дисбаланс будет ликвидирован на рынке мгновенной мощности, цены на котором отличаются в менее выгодную сторону:
   - Покупка на рынке мгновенной мощности: 20 руб./МВт[25].
   - Продажа на рынке мгновенной мощности: 0 руб./МВт[26].

Стакан

Биржевой стакан устроен просто. Сначала все заявки делятся на два списка (на покупку и на продажу) и сортируются по невыгодности для своих отправителей. Затем заявки из этих двух списков сочетаются друг с другом, превращаясь в транзакции, начиная с самых невыгодных; цена сделки — средняя от цен заявок. Если мощности в заявках не совпадают, то неудовлетворенная часть большей заявки переходит дальше. Если заявки одного из типов закончились, то оставшиеся будут удовлетворены по фиксированным ценам:

- покупка: 10 руб./МВт[27];
- продажа: 1 руб./МВт[28].

В случае, если цена предложенной заявки выходит из этого диапазона, она будет скорректирована.

На бирже, в случае заключения сделки, между игроками действует комиссия: цена для покупателя поднимается на 10 коп.[29] от вычисленной, а цена для продавца — опускается. Пример приведен в таблице 4.3.3.

Таблица 4.3.3

Заявки на покупку	**Заявки на продажу**
Заявка 1: 5 по 3,5	**Заявка 3:** 3 по 2,5
Заявка 2: 2 по 2,5	**Заявка 4:** 3 по 3

Встречаются заявки 1 и 3, заявка 3 удовлетворена полностью, а заявка 1 — частично (таблица 4.3.4).

[25] Константа `exchangeExternalInstantSell`.
[26] Константа `exchangeExternalInstantBuy`.
[27] Константа `exchangeExternalSell`.
[28] Константа `exchangeExternalBuy`.
[29] Константа `exchangeCommission`.

Таблица 4.3.4

Заявки на покупку	**Заявки на продажу**
Сделка 1.1-3: 3 по 3 (для покупателя 3 по 3,1, для продавца 3 по 2,9)	
~~**Заявка 1.1:** 3 по 3,5~~	~~**Заявка 3:** 3 по 2,5~~
Заявка 1.2: 2 по 3,5	**Заявка 4:** 3 по 3
Заявка 2: 2 по 2,5	

Встречаются оставшаяся часть заявки 1 и заявка 4, которая удовлетворяется частично (таблица 4.3.5).

Таблица 4.3.5

Заявки на покупку	**Заявки на продажу**
Сделка 1.1-3: 3 по 3 (для покупателя 3 по 3,1, для продавца 3 по 2,9)	
~~**Заявка 1.1:** 3 по 3,5~~	~~**Заявка 3:** 3 по 2,5~~
Сделка 1.2-4.1: 2 по 3,25 (для покупателя 3 по 3,35, для продавца 3 по 3,15)	
~~**Заявка 1.2:** 2 по 3,5~~	~~**Заявка 4.1:** 2 по 3~~
Заявка 2: 2 по 2,5	**Заявка 4.2:** 1 по 3

Встречаются заявки 2 и остаток заявки 4. Заявка 2 удовлетворена частично (таблица 4.3.6).

Таблица 4.3.6

Заявки на покупку	**Заявки на продажу**
Сделка 1.1-3: 3 по 3 (для покупателя 3 по 3,1, для продавца 3 по 2,9)	
~~**Заявка 1.1:** 3 по 3,5~~	~~**Заявка 3:** 3 по 2,5~~
Сделка 1.2-4.1: 2 по 3,25 (для покупателя 3 по 3,35, для продавца 3 по 3,15)	
~~**Заявка 1.2:** 2 по 3,5~~	~~**Заявка 4.1:** 2 по 3~~
Сделка 2.1-4.2: 1 по 2,75 (для покупателя 1 по 2,85, для продавца 1 по 2,65)	
~~**Заявка 2.1:** 1 по 2,5~~	~~**Заявка 4.2:** 1 по 3~~
Заявка 2.2: 1 по 2,5	

Для остатка заявки 2 не осталось встречных заявок. Она удовлетворяется из сети (допустим, ее цена — 5) (таблица 4.3.7).

Таблица 4.3.7

Заявки на покупку	**Заявки на продажу**
Сделка 1.1-3: 3 по 3 (для покупателя 3 по 3,1, для продавца 3 по 2,9)	

Заявки на покупку	**Заявки на продажу**
~~**Заявка 1.1:** 3 по 3,5~~	~~**Заявка 3:** 3 по 2,5~~
Сделка 1.2-4.1: 2 по 3,25 (для покупателя 3 по 3,35, для продавца 3 по 3,15)	
~~**Заявка 1.2:** 2 по 3,5~~	~~**Заявка 4.1:** 2 по 3~~
Сделка 2.1-4.2: 1 по 2,75 (для покупателя 1 по 2,85, для продавца 1 по 2,65)	
~~**Заявка 2.1:** 1 по 2,5~~	~~**Заявка 4.2:** 1 по 3~~
Сделка 2.2-Сеть: 1 по 10	
~~**Заявка 2.2:** 1 по 2,5~~	

Предзаполнение стакана

В стакане, кроме заявок игроков, всегда содержатся также предзаданные заявки, представляющие прочих участников энергобиржи. Таких заявок 101[30] на продажу и столько же — на покупку. Их цены равномерно распределены от константы `exchangeExternalBuy` до `exchangeExternalSell` (обе цены включены: заявки с этими ценами есть в наличии). Объем каждой заявки — 5 МВт[31].

Экологические тарифы

В нашей энергетике действует экономическое давление, направленное на модернизацию оборудования и повышение общей эффективности энергосистем. За использование ТЭС, снабжение заводов и потери на линиях с игроков взимается налог:

- Заводы — 1 руб. за МВт × такт потребления[32].
- ТЭС — 1,5 руб. за единицу использованного топлива[33].
- Потери — 2 руб. за каждый потерянный МВт × такт[34].

Собранные штрафы накапливаются в специальном «бюджете», который в конце игры распределяется между игроками пропорционально набранным ими «экологическим баллам».

Баллы можно получать:

- за получение энергии от ВЭС: 2 балла за выработанный МВт × такт[35];
- за получение энергии от киберСЭС: 3 балла за выработанный МВт × такт[36];
- за изъятие энергии их накопителей: 1 балл за извлеченный МВт × такт[37].

Кроме того, на начало игры в банке уже находится 10 000 руб.[38] и 100[39] «судей-

[30] Константа `exchangeFillAmount`.
[31] Константа `exchangeFillPower`.
[32] Константа `green.taxFactory`.
[33] Константа `green.taxTPS`.
[34] Константа `green.taxLosses`.
[35] Константа `green.awardWind`.
[36] Константа `green.awardSolarRobot`.
[37] Константа `green.awardStorage`.
[38] Константа `green.initBank`.
[39] Константа `green.initPoints`.

ских» баллов.

Потери энергии

В энергосистеме есть потери, и они могут быть значительными. Для простоты вычисляются на выходах подстанций по формуле:

$$r_p(p) = \left(\frac{p}{A}\right)^B \cdot C,$$

где $A = 50$[40]; $B = 1{,}65$[41]; $C = 0{,}3$[42].

Эти потери растут с нуля при нагрузке на энергорайон в 0 МВт до уровня в 30%[43] при 50 МВт[44] (после чего не растут). Это значит, что если на линии есть генерация в 30 МВт, то потери от нагрузки составят

$$30 \cdot 12{,}9 = 3{,}87 \text{ МВт (значения округлены до сотых)},$$

и реальная нагрузка на линию (и количество энергии, которое из нее выйдет) будет 26,13 МВт.

Если же на линии есть потребление в 20 МВт, то потери составят

$$20 \cdot 6{,}6 = 1{,}32 \text{ МВт},$$

и реальная нагрузка (сколько в нее нужно закачать, чтобы удовлетворить потребителей) будет 21,32 МВт.

Износ ЛЭП

Во время работы ЛЭП изнашиваются, и их нужно выводить в ремонт для восстановления. За 1 такт каждая ЛЭП накапливает 1 единицу «усталости», а в случае, если ЛЭП перегружена (нагрузка на ней превышает 40 МВт[45]), то 2 единицы. При накоплении «усталости» более 14 единиц[46] ЛЭП автоматически отключается. Любая отключенная ветка на следующем такте автоматически включится, вмешательство скрипта для этого не нужно.

ЛЭП можно выключить в любой момент и самостоятельно, послав скриптом соответствующую команду.

После включения ЛЭП вся накопленная «усталость» сбрасывается до нуля.

[40] Константа `lossesThreshold`.
[41] Константа `lossesRate`.
[42] Константа `lossesLimit`.
[43] Константа `lossesLimit`.
[44] Константа `lossesThreshold`.
[45] Константа `wear.overloadThreshold`.
[46] Константа `wear.failLimit`.

Скрипты

Скрипты пишутся на языке Python 3. Они загружаются в систему через вкладку «Скрипты» пользовательского интерфейса. Загруженный скрипт может находиться в системе продолжительное время, и его можно в любой момент заменить. Внутри он выполняется один раз каждый такт игры. Он видит те же данные, что и участники в интерфейсе анализа, но в отличие от них, может на них реагировать, отправляя управляющие воздействия — приказы. Их немного:

1. Произвести включение/отключение энергорайона. Обратите внимание, что отключение энергорайона отключает и все зависимые от него, но включение — нет, и включать нужно будет все по отдельности.
2. Отправить заявку на покупку или продажу энергии на бирже. Мощность ограничена 60 МВт[47], а цена — вилкой цен гарантирующего поставщика. Подробнее см. главу «Биржа».
3. Зарядить или разрядить накопитель на некоторый объем мегаватт в доступных пределах, см. раздел «Накопители».
4. Установить угол поворота КиберСЭС. Целое число от 0 до 124.
5. Сжечь некоторый объем топлива в ТЭС.

Подробнее о том, как с помощью скрипта получать данные об энергосистеме, смотрите в справке скрипта (она есть в интерфейсе и будет доступна отдельно).

В случае, если было отправлено несколько приказов для одного и того же объекта, действовать будет последний. Исключение составляют приказы для ТЭС — они складываются; повторным приказом можно только увеличить выработку ТЭС, но не уменьшить.

Аукцион

Аукцион проходит по закрытой системе, как тендеры. Участники делают ставки в течение заданного отрезка времени, после чего система подводит итог.

В отличие от прошлых лет, на аукционе разыгрываются пакеты объектов с заранее установленными тарифами (для потребителей — цена за 1 МВт $\times$ такт потребленной энергии, для остальных объектов — арендная плата за каждый такт).

Ставки на пакеты делаются на повышение по правилам all-pay (платят все), а именно, в виде фиксированных сумм, которые участники готовы заплатить, чтобы пакет лотов достался именно им, но выплачивают свои ставки, даже если они не выиграли лот.

На аукционе действует механика догоняющих ставок: если есть хоть одна ставка, которая отличается от лидирующей хотя бы на 5[48], то объявляется повторный тур аукциона для владельца лидирующей и догоняющих ставок. При этом ставка all-pay на повторном туре выплачивается отдельно от предыдущей. Если на лот никто не выставил ставку, он исключается из игры.

В конце аукциона можно сбросить один[49] лот. Они выставляются на второй круг

[47] Константа `exchangeMaxAmount`.
[48] Константа `bidDiff`.
[49] Константа `dropQuantity`.

аукциона, где участвовать в борьбе за них будут те же команды, что и в первом круге, минус команды, их сбросившие.

На аукционе разыгрывается параллельно по три лота, поэтому нужно очень аккуратно налаживать протоколы коммуникации и выставления ставок на аукционе, иначе темп работы команды может оказаться ниже темпа аукциона.

Состав объектов

Это количественный состав объектов, разыгрываемых на аукционе. Здесь не приводится, в каком порядке они объединяются в пакеты. Обратите внимание: в локальные включены объекты, идущие в одном пакете с главными подстанциями.

Стартовые объекты — выдаются командам по умолчанию.

Локальные объекты — торгуются только в пределах площадки.

Глобальные объекты — торгуются между обеими площадками.

Таблица 4.3.8

	Локальные	**Глобальные**	**Всего в игре**	**Максимум на площадке**	**Минимум на площадке**
КиберСЭС	8	15	31	23	8
ВЭС	5	4	14	9	5
ТЭС	3	7	13	10	3
Накопители	5	10	20	15	5
Подстанции	6	0	12	6	6
Миниподстанции	8	12	28	20	8
Дома	6	8	20	14	6
Офисы	4	8	16	12	4
Заводы	2	7	11	9	2

Порядок установки объектов

Порядок установки объектов важен только для КиберСЭС и ВЭС.

Приоритеты:

1. Глобальные КиберСЭС.
2. Глобальные ВЭС.
3. Локальные ВЭС.
4. Локальные КиберСЭС.

Внутри одной группы более высокий приоритет имеет объект с меньшим номером (они торгуются на аукционе раньше). На экране аукциона глобальные объекты имеют номера синего цвета, локальные — красного.

4.3.11. Приложение 3. Справка скриптов

Скрипт пишется на языке Python версии 3.10.

Особенности выполнения

Скрипт выполняется не из файла (без записи на диск). Из-за этого:

- технически текст скрипта не может занимать больше 128 Кб, иначе он не запустится;
- исключения не выдают текст строки, а лишь ее номер.

Сам скрипт выполняется из директории `~/ips3-sandbox`, и команды могут читать и записывать файлы в эту директорию. Не забудьте сохранить нужные вам файлы в конце рабочей сессии!

Базовый скрипт

Python

```python
import ips  # 1
psm = ips.init()  # 2

# ... здесь ваш код ...

psm.save_and_exit()  # 3

# команда 3 завершает скрипт,
# после неё приказы не выполнятся
```

1. Импорт библиотеки API стенда.
2. Запрос на сервер и создание объекта с данными активного хода.
3. Отправка сформированных приказов на сервер и завершение скрипта. Без этой команды скрипт бесполезен.

Получение данных

Каждый такт скрипт получает данные из энергосистемы, и они размещаются в соответствующих полях объекта `psm`.

Объекты энергосети

Объекты помещены в поле `objects` и представляют собой список объектов, хранящих данные соответствующего объекта.

Python

```python
for obj in psm.objects:
    print("== Объект:", obj.id, "==") # (тип, номер)
```

```
    print("Тип: ", obj.type) # str, см. ниже
    print("Включен:", obj.power.now.online) # bool
    print("Тариф:", obj.contract) # float
    print("Адрес:", obj.address) # [str]
    print("Энергорайоны:",
          obj.path) # [ <адрес энергорайона> ]
    print("Доход:",
          obj.score.now.income) # float
    print("Расход:",
          obj.score.now.loss) # float
    print("Доход за первый ход:",
          obj.score.then[0].income)
    print("Расход за 5ый ход:",
          obj.score.then[4].loss) # float
    print("Генерация:",
          obj.power.now.generated) # float
    print("Потребление:",
          obj.power.now.consumed) # float
    print("Потребление за первый ход:",
          obj.power.then[0].consumed)
    print("Заряд (актуально для накопителя):",
          obj.charge.now) # float
    print("Штормовой режим (актуально для ВЭС):",
          obj.failed) # bool
    print("Установленный угол (актуально для КСЭС):",
          obj.angle.now.current) # аналогично power
    print("Количество тактов с отключениями:",
          obj.offlineCount) # int

# обозначения типов объектов
obj_types = [
    "main" # подстанции
    "miniA", # мини-подстанции А
    "miniB", # мини-подстанции Б
    "houseA", # дом А
    "houseB", # дом Б (офисный центр)
    "factory", # заводы
    "storage", # накопители
    "wind", # ветровые электростанции
    "solar", # солнечные электростанции
    "solarRobot", # кибер-СЭС
    "tps", # ТЭС
]
```

Энергорайоны (нумерация с 1)

Энергорайоны помещены в поле `networks` и представляют собой словарь, где ключи — индексы (нумеруются с 1!), а значения — структуры, хранящие в себе информацию о соответствующих районах (состояние, показатели).

Python

```
for index, net in psm.networks.items():
    print("== Энергорайон", index, "==")
    print("Адрес:", net.location)
          # [ (ID подстанции, № линии) ]
    print("Включен:", net.online) # bool
    print("Генерация:", net.upflow) # float
```

```
    print("Потребление:", net.downflow) # float
    print("Потери:", net.losses) # float
    print("Износ:", net.wear) # int
    print("Перегрузка:", net.overloaded) # bool
```

Прогнозы

На каждую величину (потребление и погода) дано по несколько прогнозов. Доступ к ним осуществляется через поле `forecasts`, после чего идет обращение к типу прогнозов и индекс прогноза.

Каждый прогноз является последовательностью медиан, при этом известно максимальное отклонение значения от медианы. Это отклонение общее для всех прогнозов этого типа.

Python

```python
# дом А, 1ый ход
x = psm.forecasts.houseA[0]
# дом Б, 11ый ход
x = psm.forecasts.houseB[10]
# завод, 6ой ход
x = psm.forecasts.factory[5]
# солнце на востоке, район M56, 2ой ход
x = psm.forecasts.sunEast["M56"][1]
# солнце на западе, район M56, 12ый ход
x = psm.forecasts.sunWest["M56"][11]
# ветер, район M12, 3ий ход
x = psm.forecasts.wind["M12"][2]

# максимальное отклонение прогноза для района M34
spr = psm.forecasts.wind["M34"].spread
```

Погода

Python

```python
print("Сила ветра в районе M56:", psm.wind["M56"].now) # float
print("Была на 1 ходу:", psm.wind["M56"].then[0]) # float

print("Текущая яркость солнца на западе в районе M12:",
      psm.sunWest["M12"].now) # float
print("Яркость солнца на востоке в районе M12 на 5 ходе:",
      psm.sunEast["M12"].then[4]) # float
```

Биржа

Python

```python
print("Фактические контракты:")
for receipt in psm.exchange:
    print("Контрагент:", receipt.source)
    # "exchange" = оператор,
```

```
    # "fill"     = аноним,
    #     иначе  = другой игрок
    print("Объём:", receipt.flux)
    # Плюс = покупка, минус = продажа
    print("Цена за МВт:", receipt.price)
    print("")
```

Игрок представлен словарем с ключами `place` и `player`.

История тарифов на бирже

Python

```
# минимальный тариф за энергию
x = psm.exchangeLog[10].min
# максимальный тариф за энергию
x = psm.exchangeLog[10].max
# все примеры для 11-го хода
```

Прочая информация

Эти поля из объекта стенда не связаны с важными данными, но тоже могут пригодиться.

Python

```
print("Ход:",  psm.tick)  # int
print("Всего ходов:",  psm.gameLength)  # int
print("Изменение счёта:", psm.scoreDelta)  # float

print("Всего сгенерировано:",
      psm.total_power.generated) # float
print("Всего потреблено:",
      psm.total_power.consumed) # float
print("Получено с биржи (минус = отправлено):",
      psm.total_power.external) # float
print("Всего потерь:",
      psm.total_power.losses) # float
```

Приказы

Для управления энергосистемой используются управляющие воздействия (приказы). Участники могут объявлять их с помощью функций из `psm.orders`. При корректном завершении скрипта (`psm.save_and_exit()`) эти приказы отправляются в систему.

Приоритет приказов:

1. Линии — выполняется последний отправленный.
2. Угол КСЭС — выполняется последний отправленный.
3. ТЭС — выполняется последний отправленный.
4. Накопитель — складываются.

5. Биржа — выполняются все по отдельности.

Отмена приказов не предусмотрена!

Все числовые параметры в приказах, кроме данных в графике — ненулевые положительные!

Python

```python
# Отправить 10 МВт в накопитель c3
psm.orders.charge("c3", 10)

# Забрать 5 МВт из накопителя c1
psm.orders.discharge("c1", 5)

# Включить линию 2 на подстанции M2
psm.orders.line_on("M2", 2)

# Выключить линию 1 на мини-подстанции m1
psm.orders.line_off("m1", 1)

# Заявка на продажу 10,2 МВт за 2,5 руб./МВт
psm.orders.sell(10.2, 2.5)

# Заявка на покупку 5,5 МВт за 5,1 руб./МВт
psm.orders.buy(5.5, 5.1)

# Выставить угол КСЭС r5 на 75 ед.
psm.orders.robot("r5", 75)

# Сжечь 5 ед. топлива в ТЭС r4
psm.orders.tps("r4", 5)
```

Отладка

Для локальной проверки и отладки скрипта можно использовать локальную среду IDLE со встроенным модулем ips.

В модуле реализованы команды:

- Для замены `init()`:
  - `init_test()` — данные из вшитого примера;
  - `from_json(json_str)` — данные из JSON-string;
  - `from_file(filename)` — данные из файла в JSON-формате.
- В дополнение к `Powerstand.save_and_exit` (без сервера он не выполняется):
  - `print(psm.get_user_data())` — вывод данных из пользовательских графиков;
  - `print(psm.orders.get())` — вывод приказов в чистом виде в `std out` без завершения скрипта;
  - `print(psm.orders.humanize())` — то же самое, но приказы представлены в читаемом виде.

Пример отладочной версии скрипта и правок для получения оной см. ниже.

Python

```python
import ips

# psm = ips.init()  # было
psm = ips.init_test() # стало

# ... здесь ваш код ...

print("\n".join(psm.orders.humanize()))
# графики в приказах не приводятся, но можно так:
print(psm.get_user_data())

# строку можно оставить, отправка в систему
# не производится, только выход из скрипта
psm.save_and_exit()
```

5. Критерии определения победителей и призеров

Первый отборочный этап

В первом отборочном этапе участники решали задачи предметного тура по двум предметам: математике и информатике и инженерного тура. В каждом предмете максимально можно было набрать 100 баллов, в инженерном туре 100 баллов. Для того чтобы пройти во второй этап, участники должны были набрать в сумме по обоим предметам и инженерному туру не менее 30,0 баллов, независимо от уровня.

Второй отборочный этап

Количество баллов, набранных при решении всех задач второго отборочного этапа, суммируется. Победители второго отборочного этапа должны были набрать не менее 104,0 баллов, независимо от уровня.

Заключительный этап

Индивидуальный предметный тур

- математика — максимально возможный балл за все задачи — 100 баллов;
- информатика — максимально возможный балл за все задачи — 100 баллов.

Командный инженерный тур

Команды заключительного этапа получали за командный инженерный тур от 0 до 100,00 баллов: команда, набравшая наибольшее число баллов среди других команд, становилась командой-победителем.

Все результаты команд нормировались по формуле:

$$\frac{100 \times x}{MAX},$$

где x — число баллов, набранных командой,

MAX — число баллов, максимально возможное за инженерный тур.

В заключительном этапе олимпиады индивидуальные баллы участника складываются из двух частей, каждая из которых имеет собственный вес: баллы за индивидуальное решение задач по предмету 1 (математика) с весом $K_1 = 0{,}2$, по

предмету 2 (информатика) с весом $K_2 = 0{,}2$, баллы за командное решение задач инженерного тура с весом $K_3 = 0{,}6$.

Итоговый балл определяется по формуле:

$$S = K_1 \cdot S_1 + K_2 \cdot S_2 + K_3 \cdot S_3,$$

где S_1 — балл первой части заключительного этапа по математике (предметный тур) ($S_{1\ \text{макс}} = 100$);

S_2 — балл первой части заключительного этапа по информатике (предметный тур) ($S_{2\ \text{макс}} = 100$);

S_3 — итоговый балл инженерного командного тура ($S_{3\ \text{макс}} = 100$).

Итого максимально возможный индивидуальный балл участника заключительного этапа — 100 баллов.

Критерий определения победителей и призеров

Чтобы определить победителей и призеров (независимо от класса) на основе индивидуальных результатов участников, был сформирован общий рейтинг всех участников заключительного этапа. С начала рейтинга были выбраны 3 победителя и 7 призеров (первые 25% участников рейтинга становятся победителями или призерами, из них первые 8% становятся победителями, оставшиеся — призерами).

Критерий определения победителей и призеров (независимо от уровня)

Категория	**Количество баллов**
Победители	71,27 и выше
Призеры	От 52,67 до 67,87

6. Работа наставника после НТО

Участие школьника в Олимпиаде может завершиться после любого из этапов: первого или второго отборочных, либо после заключительного этапа. В каждом случае после завершения участия наставнику необходимо провести с учениками рефлексию — обсудить полученный опыт и проанализировать, что позволило достичь успеха, а что привело к неудаче. Подробные материалы о проведении рефлексии представлены в курсе «Наставник НТО»: https://academy.sk.ru/events/310.

Наставнику важно проинформировать руководство образовательного учреждения, если его учащиеся стали финалистами, призерами и победителями. Публичное признание высоких результатов дополнительно повышает мотивацию.

В процессе рефлексии с учениками, не ставшими призерами или победителями, рекомендуется уделить особое внимание особенностям командной работы: распределению ролей, планированию работы, возникающим проблемам. Для этого могут использоваться опросники для самооценки собственной работы и взаимной оценки участниками других членов команды (P2P). Они могут выявить внутренние проблемы команды, для решения которых в план подготовки можно добавить мероприятия, направленные на ее сплочение.

Стоит рассказать, что в истории НТО было много примеров, когда не победив в первый раз, на следующий год участники показывали впечатляющие результаты, одержав победу сразу в нескольких профилях. Конечно, важно отметить, что так происходит только при учете прошлых ошибок и подготовке к Олимпиаде в течение года.

Важным фактором успешного участия в следующих сезонах НТО может стать поддержка родителей учеников. Знакомство с ними помогает наставнику продемонстрировать важность компетенций, развиваемых в процессе участия в НТО, для будущего образования и карьеры школьников. Поддержка родителей помогает мотивировать участников и позволяет выделить необходимое время на занятия в кружке.

С участниками-выпускниками наставнику рекомендуется обсудить их дальнейшее профессиональное развитие и его связь с выбранными профилями НТО. Отдельно можно обратить внимание на льготы для победителей и призеров, предлагаемые в вузах с интересующими ученика направлениями. Кроме того, ряд вузов предлагает льготы для всех финалистов НТО, а также учитывает результаты Конкурса цифровых портфолио «Талант НТО».