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

по профилю «Программная инженерия в финансовых технологиях»

2024/25 учебный год

ntcontest.ru

УДК 373.5.016:336
ББК 74.266.5
Ф59

Авторы:
М. В. Бабушкин, Е. А. Болдырева, Н. В. Ведерников, И. А. Воронцов, А. А. Гаврилюк, Е. Н. Горечин, А. Р. Ершов, О. В. Зубков, Е. В. Милованович, Н. А. Циминтия

Ф59 Всероссийская междисциплинарная олимпиада школьников 8–11 класса «Национальная технологическая олимпиада». Учебно-методическое пособие
Том 27 **Программная инженерия в финансовых технологиях**
— М.: Ассоциация участников технологических кружков, 2025. — 197 с.

ISBN 978-5-908021-26-5

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
12.00.00 Фотоника, приборостроение, оптические и биотехнические системы и технологии
13.00.00 Электро- и теплоэнергетика
15.00.00 Машиностроение
24.00.00 Авиационная и ракетно-космическая техника
27.00.00 Управление в технических системах
45.00.00 Языкознание и литературоведение

ISBN 978-5-908021-26-5
9 785908 021265 >

УДК 373.5.016:336
ББК 74.266.5

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. Программная инженерия в финансовых технологиях

Профиль Программная инженерия в финансовых технологиях нацелен на разработку пользовательских решений в сфере финансовых технологий, применение которых дает возможность более качественно удовлетворить потребности частных и институциональных клиентов на всех видах современных рынков (в том числе цифровых активов) и способствует привлечению инвестиций в экономику.

В финансовом инжиниринге используется подход комбинирования финансовых инструментов с учетом различных параметров риска и доходности для реализации инвестиционной стратегии квалифицированных инвесторов. Программная инженерия в финансовых технологиях играет ключевую роль в клиентском бизнесе производных активов, разрабатывая и внедряя на практике структурированные продукты — деривативы.

Спрос на финансовых инженеров растет как в государственном, так и корпоративном секторе. Финансовым инжинирингом занимаются государственные корпорации, крупные банки, молодые стартап-компании.

Профиль Программная инженерия в финансовых технологиях привлекателен для тех, кто:

- хочет разрабатывать инновационные продукты, например, для таких экосистем как «Сбербанк» и «Тинькофф»;
- стремится научиться применять современные финансовые технологии (деривативы, торговые стратегии и хеджирование рисков);
- интересуется последними тенденциями развития финансовых рынков, услуг и сервисов.

Для подготовки участникам предоставлены материалы по программированию на Python и использованию основных библиотек для анализа данных, основам машинного обучения, финансовом моделировании и анализе, практикумы и сборники задач с предыдущих соревнований.

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

Урок погружает участников в выполнение реальных задач, связанных с анализом финансовых показателей и визуализацией данных, знакомит школьников с такими понятиями, как большие данные, машинное обучение, а также предлагает решить сложную задачу художественного переноса стиля на языке программирования Python с использованием инструмента визуализации Jupyter Notebook.

В рамках **первого отборочного этапа** участники:

- решают задачи по математике и информатике на предметном туре;
- осваивают теорию веб-разработки и API, визуализации данных через образовательный блок;
- развивают компетенции в анализе данных на инженерном туре.

Задачи **второго этапа** по олимпиадному программированию машинному обучению готовят к заключительному этапу.

На заключительном этапе участники соревнуются в разработке интерактивного пользовательского дашборда для ПриветБанка, содержащего информацию о финансовых показателях, графики и предсказания. Им необходимо:

1. выгрузить и обработать финансовые данные;
2. интерпретировать их;
3. построить финансовые модели и зависимости;
4. обеспечить техническое функционирование дашборда;
5. разработать интерфейс.

Участникам предоставляется доступ к репозиториям GitHub; они могут использовать необходимые библиотеки и языки. Решения загружаются в репозиторий GitHub в формате Docker-контейнера, а также файлом READ.MD, содержащим инструкцию по запуску проекта. После загрузки команды защищают дашборды перед комиссией жюри, демонстрируя функционал, свободу владения инструментами разработки.

Для подготовки к заключительному этапу предоставляется подборка материалов по машинному обучению, построению алгоритмов предсказательных систем, веб-разработки, на которые нацелена финальная задача. Для организации эффективной подготовки проводятся вебинары по тематике профиля и разбор заданий второго этапа.

Компетенции, приобретаемые благодаря участию в данном профиле Олимпиады, способствуют формированию hard skills, а именно, навыков:

- программирования;
- машинного обучения;
- разработки интерфейсов;
- аналитики данных,

а также знание как минимум языков программирования Python или R, основных библиотек, алгоритмов, их ограничений, базовые компетенции в теории вероятностей, статистике, математическом анализе и линейной алгебре.

Soft skills:

- умение работать в команде;
- эмоциональный интеллект;
- самоорганизация;
- тайм-менеджмент;
- проявление лидерских качеств;
- принятие решений;
- самостоятельная работа с учебными материалами.

Все эти навыки и компетенции в равной степени помогают участникам на пути к становлению гармоничной и развитой личности.

Победители и призеры профиля Программная инженерия в финансовых технологиях демонстрируют уровень навыков и компетенций, востребованных в веду-

щих вузах России на специальностях, связанных с информационными технологиями, и принимают участие в научно-технологических проектных школах.

Участие школьников в данном профиле позволяет повысить популярность и осознанность выбора профессии в области IT-технологий. Собственный реальный опыт в этой области дает возможность уже в школьном возрасте понять свое отношение и выбрать целевой профильный вуз, а значит, определить 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});
    }
```

```
    int s;
    cin >> s;
    dfs(s);
    reverse(order.begin(), order.end());
    vector<int> dp(n + 1, -1);
    dp[s] = 0;
    for(auto el : order){
        for(auto to : G[el]){
            dp[to.first] = max(dp[to.first], dp[el] + to.second);
        }
    }
    for(int i = 1; i <= n; i++){
        cout << dp[i] <<' ';
    }
}
```

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 $n \times m$.

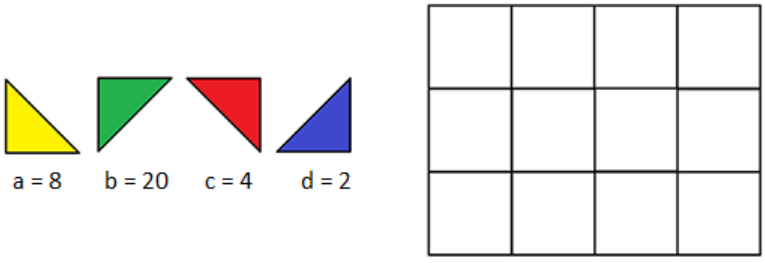


Рис. 2.2.1

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.

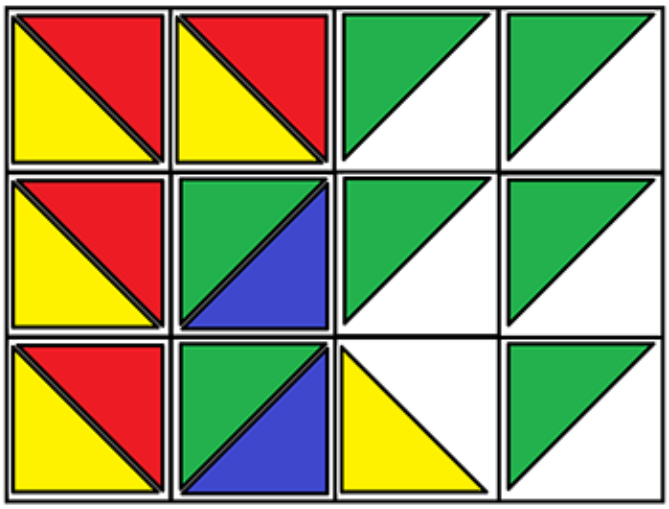


Рис. 2.2.2

Решение

Ниже представлено решение на языке C++.

C++

```
#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.

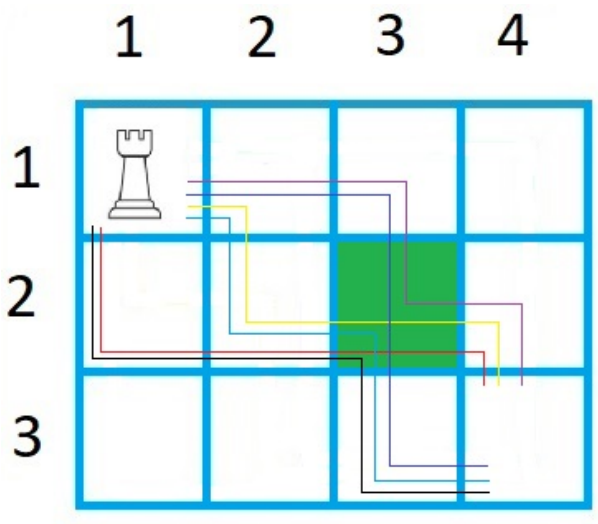


Рис. 2.2.3

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

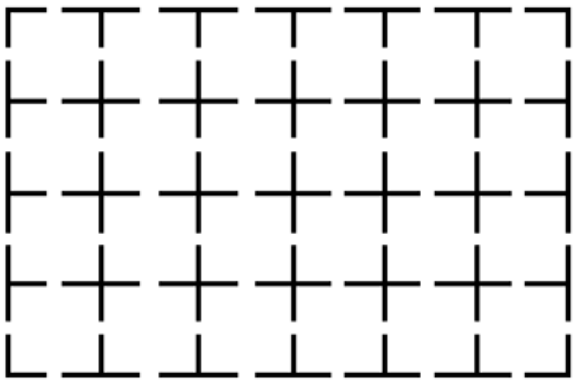


Рис. 2.2.4

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.

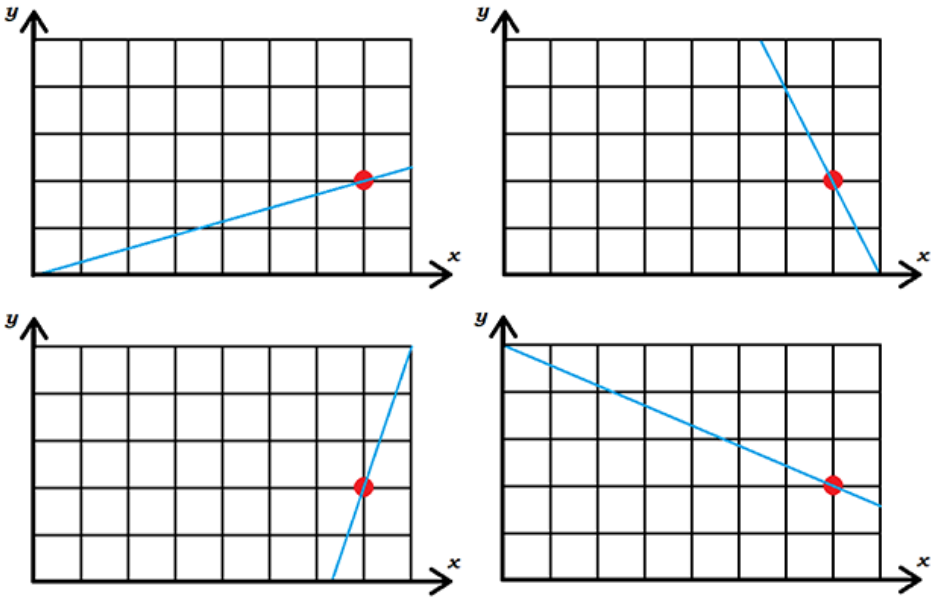


Рис. 2.2.5

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.

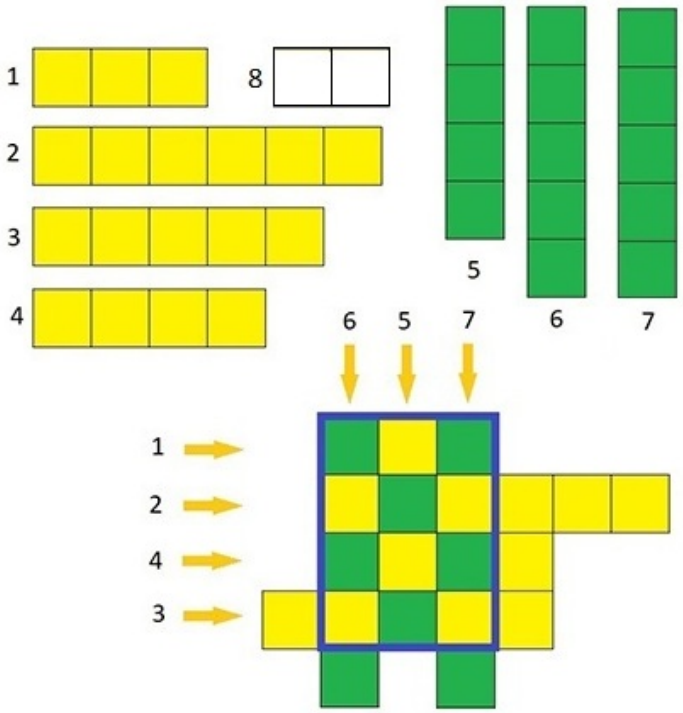


Рис. 2.2.6

Решение

Ниже представлено решение на языке C++.

C++

```cpp
#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

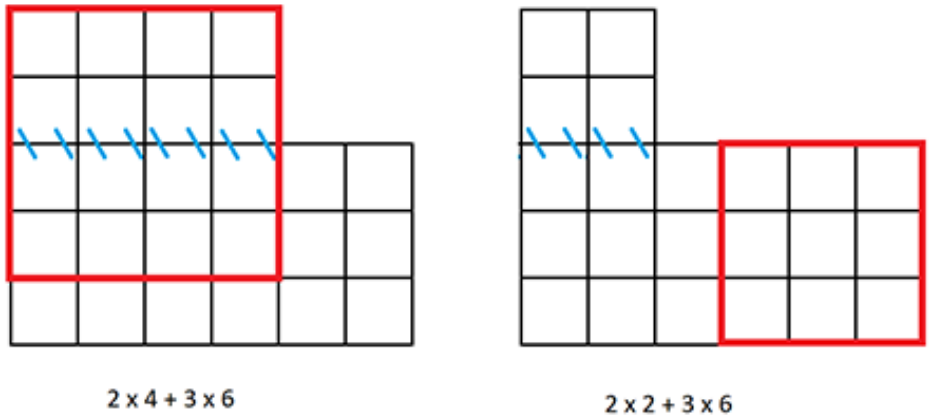


Рис. 2.2.7

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.

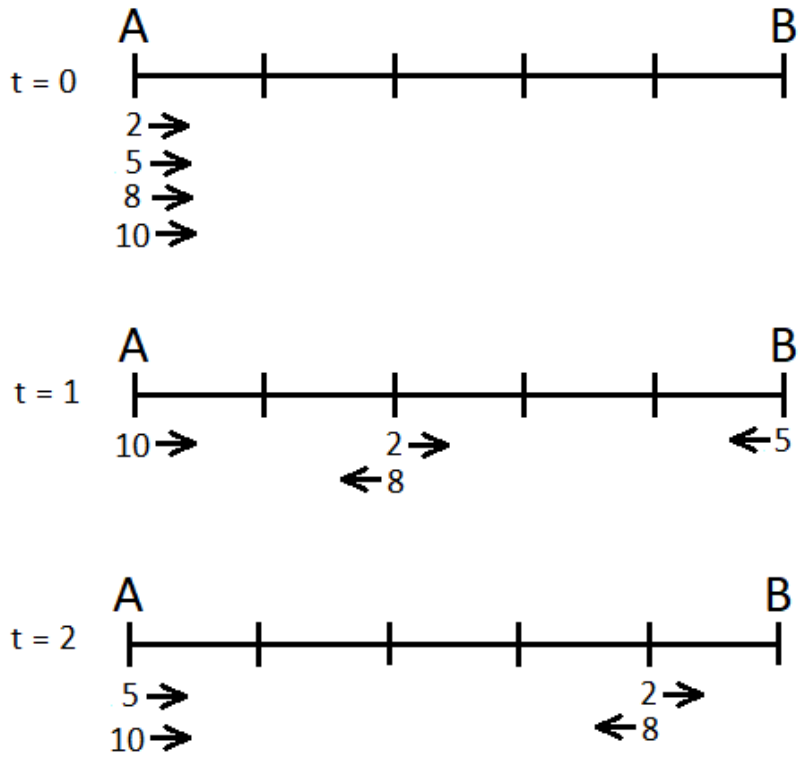


Рис. 2.2.8

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)

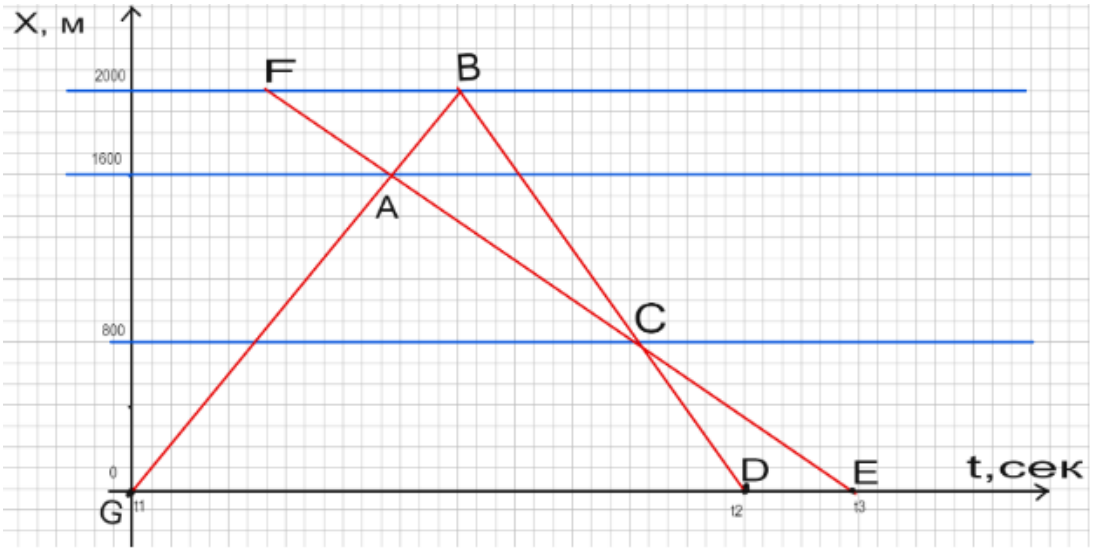
Темы: текстовые задачи и логика, графическое изображение движения, теорема Менелая.

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

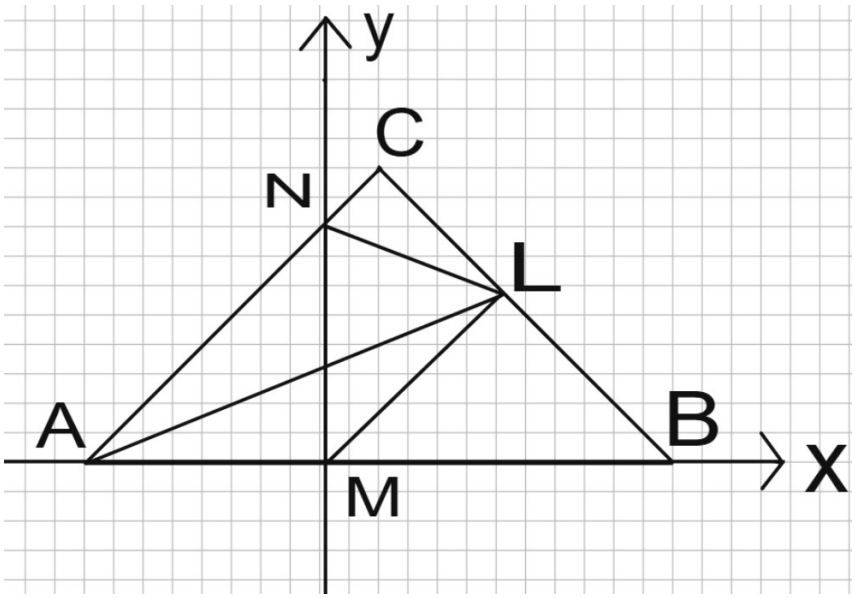
$MN = AM$, значит, угол ANM тоже равен 45° и NM перпендикулярно AB.

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.

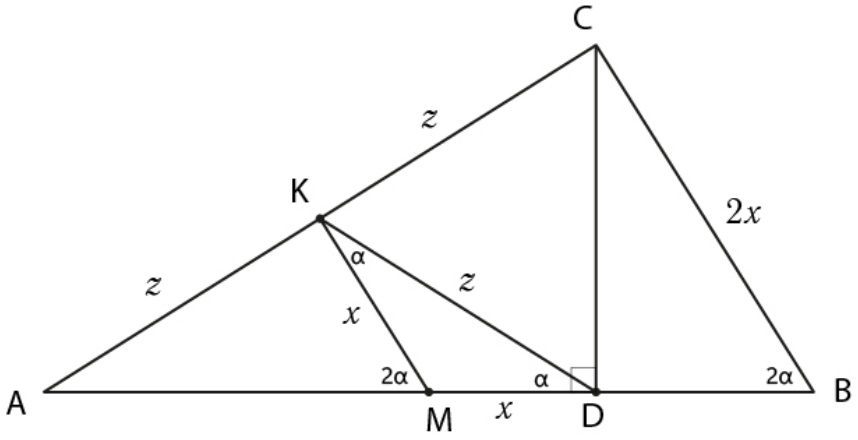
Задача 2.3.5.5. (25 баллов)

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)

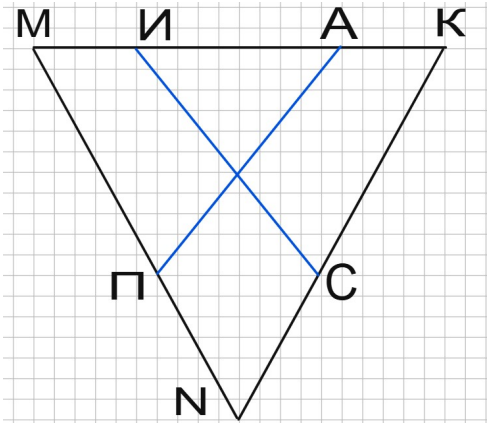
Темы: равносторонний треугольник, первый признак равенства треугольников.

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 · 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.

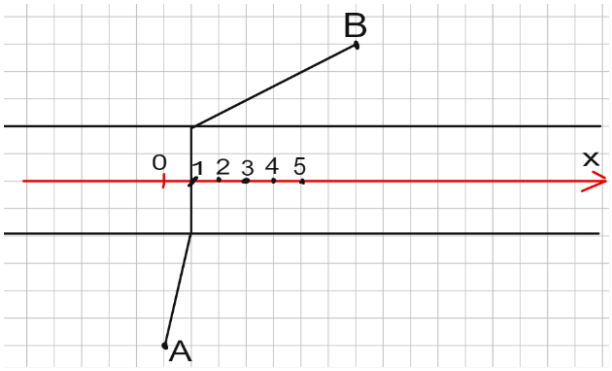


Рис. 2.3.1

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.

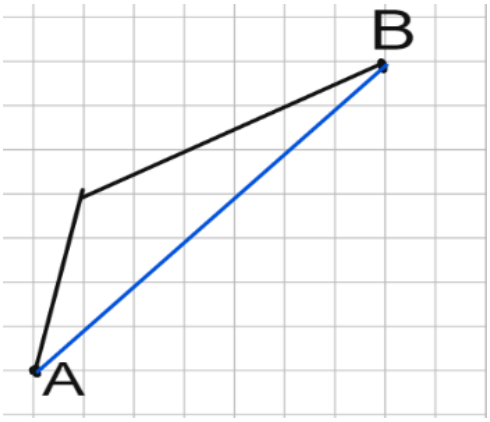


Рис. 2.3.2

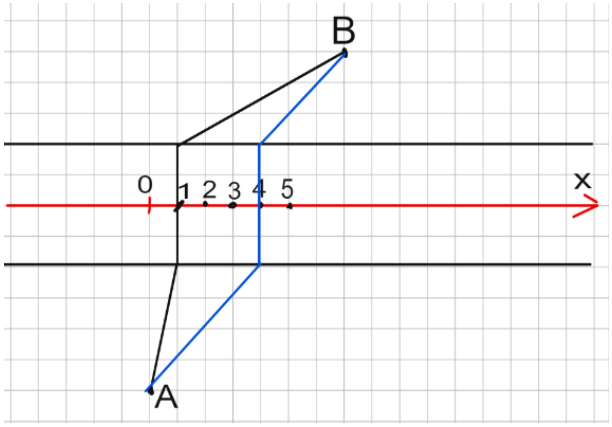


Рис. 2.3.3

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.

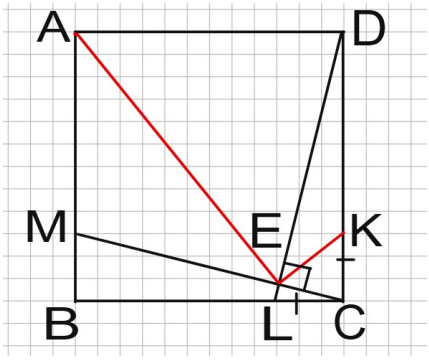


Рис. 2.3.4

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

Задача 2.4.1. Кредиты (100 баллов)

Тема: жадные алгоритмы.

Условие

Михаил задумал купить автомобиль и решил сначала посоветоваться со своим другом Сергеем. Сам Михаил хотел купить новый китайский автомобиль Халва, а его друг Сергей предложил купить старый немецкий БНВ. После долгих обсуждений Михаил принял решение купить оба автомобиля. Для этого ему потребовалось много денег, из-за чего Михаил набрал много «кредитов с подвохом».

Подвох таких кредитов в том, что в отличие от обычных, они устроены как накопительный счет, но только наоборот. То есть каждый месяц задолженность увеличивается на несколько процентов от текущей задолженности. Например, если Михаил взял в кредит с подвохом 10 рублей под 5 процентов, то на следующий месяц он будет должен уже 10,5 рублей, а через месяц — 11,025 рублей при условии, что Михаил не пытался выплачивать долг. Выплата долга происходит раз в месяц, до пересчета величины долга. Размер выплаты может быть любым, не обязательно целым числом.

Помогите Михаилу определить, сколько он в итоге потратит на погашение кредитов при оптимальной стратегии выплат, если каждый месяц у него есть фиксированная сумма денег, которую Михаил может потратить на выплаты долгов. Гарантируется, что кредиты можно выплатить за 2024 месяца.

Формат входных данных

Первая строка содержит натуральное число N ($1 \leqslant N \leqslant 100$) — число взятых кредитов.

Далее следуют N строк. Каждая i-я из этих строк содержит два целых числа d_i и p_i ($1 \leqslant d_i \leqslant 1000$, $0 \leqslant p_i \leqslant 100$) — размер задолженности по i-му кредиту и ежемесячный процент.

Последняя строка содержит одно целое число M ($1 \leqslant M \leqslant 1000$) — сколько Михаил может суммарно тратить в месяц на погашение задолженностей.

Формат выходных данных

Выведите одно вещественное число — ответ на задачу. Он будет засчитан, если абсолютная или относительная погрешности не превышают 10^{-6}.

Примеры

Пример №1

Стандартный ввод
3 1 1 2 2 3 3 4
Стандартный вывод
6.03

Решение

Оптимальной будет стратегия выплаты кредитов по убыванию процентов. Для вычисления общей суммы требуется проэмулировать помесячный процесс выплаты кредитов. На каждом шаге нужно завести переменную с текущей суммой и вычитать ее из оставшейся суммы кредита.

Для удобства сделаем замену $p'_i = (1 + p_i)$. Рассмотрим две пары индексов x и y. Предположим, что $d_x > M$ и $d_y > M$. Пусть в первый месяц будет произведен платеж за x-й, а во второй — за y-й кредит. Тогда после первого месяца общий долг за эти два кредита будет равен:

$$(d_x - M) \cdot p'_x + d_y \cdot p'_y,$$

а после второго:

$$(d_x - M) \cdot p'_x \cdot p'_x + (d_y \cdot p'_y - M) \cdot p'_y.$$

Раскроем скобки и получим:

$$d_x \cdot (p'_x)^2 - M \cdot (p'_x)^2 + d_y \cdot (p'_y)^2 - M \cdot p'_y.$$

Если же изменить порядок платежей, то общий долг за эти два кредита будет равен:

$$d_x \cdot (p'_x)^2 - M \cdot p'_x + d_y \cdot (p'_y)^2 - M \cdot (p'_y)^2.$$

Вычтем первую сумму из второй и получим:

$$M \cdot (p'_x)^2 + M \cdot p'_y - M \cdot p'_x - M \cdot (p'_y)^2 = M \cdot (((p'_x)^2 - p'_x) - ((p'_y)^2 - p'_y)).$$

Так как $M > 0$ и $p'_i \geqslant 1$, то разность будет положительной, если $p'_x > p'_y$.

Следовательно, выгодней выплачивать сначала кредит с наибольшим процентом вне зависимости от текущих задолженностей.

Ниже представлено решение на языке Python.

Python

```
n = int(input())
d = [0.0] * n
```

```
p = [0.0] * n

for i in range(n):
    d[i], p[i] = map(float, input().split())

m = float(input())

ans = 0.0

order = list(range(n))
order.sort(key=lambda i: -p[i])

for s in range(2024):
    c = m

    for i in order:
        if c > d[i]:
            c -= d[i]
            d[i] = 0
        else:
            d[i] -= c
            c = 0
    ans += m - c

    for i in range(n):
        d[i] *= (p[i] / 100.0 + 1)

    if c > 0:
        print(ans)
        break
```

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

В рамках второго этапа участникам предстоит решить комплекс задач, направленных на демонстрацию навыков программирования, работы с данными и базами данных в контексте финансового менеджмента. Для их выполнения необходимо использовать язык программирования Python, современные подходы к обработке данных и разработке программных решений, актуальных для финансовой отрасли.

3.2.1. Индивидуальные задачи

Легенда

Давайте знакомиться! Ваш спутник на этом этапе Олимпиады — ученик старшей школы Георгий. Любимый мультяшный персонаж из глубокого детства Георгия — Дядя Скрудж, который, благодаря своему усердию и умению обращаться с деньгами, стал самым богатым утенком на свете. И Георгий убежден, что может так же — заработать и накопить достаточно денег, чтобы осуществить свои самые смелые мечты.

Вдохновленный примерами успешных утят и некоторых не менее успешных людей, Георгий выбрал одну из самых перспективных профессий — программист. Он слышал, что хорошие программисты получают высокие зарплаты, а отличные программисты — еще больше. Каждый день он усердно учился, изучал языки программирования и решал сложные задачи, мечтая о том, как однажды сможет стать мастером своего дела и, возможно, превзойти даже Дядю Скруджа в умении управлять финансами.

Еще Георгий знает: чтобы стать богатым, нужно не только зарабатывать, но и уметь грамотно распоряжаться деньгами. Для достижения поставленной цели Георгий решил внимательно относиться и к личным расходам и доходам, и к расходам и доходам семьи. Он начал вести дневник, в котором записывал все свои траты на продукты, развлечения, транспорт и другие нужды. Помогите Георгию справиться с задачами второго этапа и подготовиться к заключительному. И, кто знает, возможно, однажды вы вместе с Георгием станете не только богатыми, но и мудрыми, как его кумиры.

Задача 3.2.1.1. Разминка. Сумма вклада (100 баллов)

Темы: финансовая грамотность, программирование.

Условие

Предположим, Георгий решил открыть вклад в банке на некую сумму в рублях под определенный процент годовых с капитализацией процентов. Проценты начисляются ежеквартально. Важно также, на какой период времени в годах открыт вклад.

Напишите функцию для расчета итоговой суммы вклада по окончании срока с учетом капитализации процентов. На вход функции должны подаваться следующие данные:

- начальная сумма вклада;
- процентная ставка (в процентах);
- срок вклада в годах.

Для написания функции необходимо использовать язык программирования Python. Для того чтобы ответ был засчитан корректно, функция должна выводить **только** целую часть числа без округления (!).

Для расчета итоговой суммы по вкладу с капитализацией используется формула сложного процента:

$$S = P \cdot \left(1 + \frac{r}{m}\right)^{m \cdot n},$$

где

- S — итоговая сумма;
- P — начальный вклад;
- r — годовая процентная ставка (в десятичной форме);
- m — количество капитализаций в году;
- n — количество лет.

Формат входных данных

В первой строке одно целое число — начальная сумма.

Во второй строке одно целое число — процентная ставка.

В третьей строке одно целое число — срок кредита в годах.

Формат выходных данных

Единственное целое число — сумма вклада по окончании срока с учетом капитализации процентов.

Примеры

Пример №1

Стандартный ввод
100000 6 2

Стандартный вывод
112649

Решение

Ниже представлено решение на языке Python.

Python

```python
import math

def calculate_final_amount(principal, annual_rate, years,
    ↪ compounding_frequency):
rate_per_period = annual_rate / 100 / compounding_frequency
total_periods = years * compounding_frequency
final_amount = principal * (1 + rate_per_period) ** total_periods
return final_amount

principal = int(input())  # начальная сумма вклада
annual_rate = int(input()) # годовая процентная ставка
years = int(input())   # срок вклада в годах

compounding_frequency = 4  # количество капитализаций в год
    ↪ (ежеквартально)

# Вычисляем итоговую сумму, используем функцию math.floor,
# чтобы получить только целую часть
final_amount = math.floor(calculate_final_amount(principal,
    ↪ annual_rate, years, compounding_frequency))
print(final_amount)
```

Задача 3.2.1.2. Разминка. Кредит (100 баллов)

Темы: финансовая грамотность, программирование.

Условие

Георгий собирается взять кредит на некую сумму в рублях. Банк предлагает погашать кредит равными ежемесячными платежами (аннуитетные платежи). Напишите функцию для расчета ежемесячного платежа по кредиту. На вход функции должны подаваться следующие данные:

- сумма кредита;
- годовая процентная ставку (в процентах);
- срок кредита в годах.

Не забудьте рассчитать ежемесячный платеж по кредиту по формуле аннуитета:

$$A = P \cdot \frac{r \cdot (1+r)^n}{(1+r)^n - 1},$$

где

- A — аннуитетный платеж;
- P — сумма кредита;
- r — месячная процентная ставка (годовая ставка делится на 12 и переводится в десятичную дробь);
- n — общее количество платежей (количество месяцев).

Для написания функции используйте язык программирования Python. Для того чтобы ответ был засчитан корректно, функция должна выводить **только** целую часть числа без округления.

Формат входных данных

В единственной строке три целых числа, разделенных пробелом: сумма кредита, процентная ставка и срок кредита в годах.

Формат выходных данных

Единственное целое число — сумма ежемесячного платежа по кредиту.

Примеры

Пример №1

Стандартный ввод
521000 12 3

Стандартный вывод
17304

Решение

Ниже представлено решение на языке Python.

Python

```python
import math

def calculate_monthly_payment(principal, annual_rate, years):
# Преобразуем годовую процентную ставку в месячную
monthly_rate = annual_rate / 100 / 12
total_payments = years * 12
# Рассчитываем ежемесячный платеж по формуле аннуитета
monthly_payment = principal * (monthly_rate * (1 + monthly_rate) **
    ↪  total_payments) / ((1 + monthly_rate) ** total_payments - 1)
return monthly_payment

principal,  annual_rate, years = (int(x) for x in input().split())
# сумма кредита, годовая процентная ставка (в процентах!) и срок
    ↪  кредита в годах

# Вычисляем ежемесячный платеж, используем функцию math.floor,чтобы
    ↪  получить только целую часть
monthly_payment = math.floor(calculate_monthly_payment(principal,
    ↪  annual_rate, years))
print(monthly_payment)
```

Задача 3.2.1.3. Разминка. Расчет зарплаты (100 баллов)

Темы: финансовая грамотность, программирование.

Условие

Программист получает зарплату, которая зависит не только от уровня и количества технологий, но и от дополнительных бонусов.

Бонусы начисляются за участие в проектах:

- за каждый проект начисляется X руб.;
- программист уровня Junior получает базовую ставку 45 000 руб. и по 4 000 руб. за каждую технологию;
- программист уровня Middle получает базовую ставку 90 000 руб. и по 8 000 руб. за каждую технологию;
- программист уровня Senior получает базовую ставку 140 000 руб. и по 15 000 руб. за каждую технологию.

Напишите функцию, которая принимает на вход количество технологий, количество проектов и бонус за каждый проект в рублях (целое, положительное число рублей, строго больше нуля) через пробел и возвращает зарплату программистов всех уровней с учетом всех факторов в одной строке через пробел.

Для написания функции используйте язык программирования Python. Для того чтобы ответ был засчитан корректно, функция должна выводить **только** целую часть числа без округления.

Формат входных данных

Единственная строка содержит три целых числа, разделенных пробелом: количество технологий, количество проектов и бонус.

Формат выходных данных

В одной строке через пробел три целых числа: зарплата программистов всех уровней в порядке Junior, Middle, Senior.

Примеры

Пример №1

Стандартный ввод
`2 1 5000`

Стандартный вывод
`58000 111000 175000`

Решение

Ниже представлено решение на языке Python.

Python

```python
def calculate_salary_with_bonus(level, tech_count, project_count, bonus):
    if level == 'Junior':
        base_salary = 45000
        bonus_per_tech = 4000
    elif level == 'Middle':
        base_salary = 90000
        bonus_per_tech = 8000
    elif level == 'Senior':
        base_salary = 140000
        bonus_per_tech = 15000
    else:
        return 0
    project_bonus = project_count * bonus
    return base_salary + bonus_per_tech * tech_count + project_bonus

technologies_count,  projects_count, bonus = (int(x) for x in input().split())
junior = math.floor(calculate_salary_with_bonus('Junior', technologies_count, projects_count, bonus))
middle = math.floor(calculate_salary_with_bonus('Middle', technologies_count, projects_count, bonus))
senior = math.floor(calculate_salary_with_bonus('Senior', technologies_count, projects_count, bonus))
print(junior, middle, senior)
```

Задача 3.2.1.4. Разминка. Доходность акций (100 баллов)

Темы: финансовая грамотность, программирование.

Условие

Допустим, речь идет об инвестициях в акции. Имеются данные о месячной доходности акций за последние 12 месяцев. Чтобы оценить риск инвестиций, нужно рассчитать стандартное отклонение доходности, которое показывает, насколько сильно доходность может колебаться от среднего значения.

Напишите функцию, которая принимает список из 12 значений доходности (в процентах) в качестве строки (чисел, разделенных пробелами) и возвращает стандартное отклонение. Для написания функции используйте язык программирования Python. Для того чтобы ответ был засчитан корректно, функция должна выводить стандартное отклонение в процентах, округленное до двух знаков после запятой, используя функцию `round` (или ее аналог).

Примечание. Возможно, потребуется импортировать библиотеку `math`, и если нужно почитать подробнее про стандартное отклонени, рекомендация https://journal.tinkoff.ru/indicators/.

Формат входных данных

В единственной строке список из 12 целых чисел, разделенных пробелами — значений доходности (в процентах).

Формат выходных данных

Единственное число — стандартное отклонение в процентах, округленное до двух знаков после запятой (разделитель — точка).

Примеры

Пример №1

Стандартный ввод
`5 7 -3 8 6 2 10 -1 4 9 3 7`

Стандартный вывод
`3.79`

Решение

Ниже представлено решение на языке Python.

Python

```python
import math

def calculate_std(deviations):
    # Находим среднее значение доходности
    avg = sum(deviations) / len(deviations)
    # Рассчитываем и возвращаем отклонение каждого значения от среднего
    variance = sum((x - avg) ** 2 for x in deviations) / len(deviations)
    return round(math.sqrt(variance), 2)

input_list = [int(x) for x in input().split()]
print(calculate_std(input_list))
```

Задача 3.2.1.5. Работа с данными (300 баллов)

Темы: финансовая грамотность, программирование, работа с данными, работа с библиотекой pandas.

Условие

Теоретическая часть

Георгий уже начал анализировать финансовые потоки внутри семьи, но поделиться доходами на широкую аудиторию пока не готов. Поэтому он решил предложить вам вместе с ним покопаться в графе «Расходы».

Для этого Георгий аккуратно внес в таблицу все расходы с датами, суммами и категориями расходов и сохранил в формате csv: https://disk.yandex.ru/d/KrD3mMJ1_6ds3Q.

С этим форматом, как настоящим аналитикам, придется работать, используя соответствующую библиотеку `pandas`. Начало работы с этой библиотекой осуществляется ее импортом в файл проекта с помощью команды `import pandas`. Для более краткого обращения к библиотеке из кода проекта указывается короткое имя `pd`:

Python

```python
import pandas as pd
```

Библиотека Pandas и датафреймы

Pandas — это Python-библиотека, созданная специально для отображения, анализа и обработки структурированных данных. Название библиотеки расшифровывается как panel data или «панельные данные». Панельными называют данные, представленные в виде таблиц и имеющие четкую структуру (структурированные).

У библиотеки есть официальный сайт, на котором описано, как можно начать работу с этой библиотекой — почитать тут: https://pandas.pydata.org/getting_started.html.

Датафреймы (тип данных DataFrame из библиотеки Pandas) — это таблица со строками и столбцами, которая похожа на таблицу из приложения Excel. В этом типе данных можно разместить данные из нашего `csv`-файла и потом их обрабатывать: сортировать, группировать, производить вычисления.

Прочитаем содержимое датафрейма в переменную `df`, используя функцию:

Python

```python
pd.read_csv('expenses.csv', index_col=[0], parse_dates=[0])
```

В функции `pd.read_csv()` из библиотеки Pandas параметры `index_col` и `parse_dates` выполняют следующие функции.

- `index_col=[0]`: данный параметр указывает, что первый столбец (индекс 0) в загружаемом `csv`-файле должен использоваться в качестве индекса для датафрейма. Это значит, что значения из этого столбца будут служить уникальными метками для строк, а не просто обычным столбцом данных.
- `parse_dates=[0]`: данный параметр указывает, что первый столбец (также индекс 0) нужно интерпретировать как даты. То есть, если в этом столбце находятся даты, Pandas автоматически преобразует их в формат `datetime`, что позволяет удобно работать с временными рядами и выполнять операции с датами.

Таким образом, эта строка кода загружает данные из файла `expenses.csv`, устанавливает первый столбец в качестве индекса и преобразует его в формат даты.

Подготовка и обзор данных

Сначала Георгий предлагает прочитать таблицу с данными и ознакомиться с ними:

Python

```python
data = pd.read_csv('НАЗВАНИЕ_ФАЙЛА.csv', index_col=[0], parse_dates=[0])
```

Выведем первые 10 строк датафрейма и информацию о полях датафрейма. Столбцы называются полями.

Проверим, что все считано корректно.

Python

```python
data.head(10)
data.info()
```

Задание 1. Типы данных (2 балла)

Выведите тип данных для колонки `category` (полное название типа данных с маленькой буквы).

Задание 2. Типы данных (2 балла)

Выведите тип данных для колонки `amount` (полное название типа данных с маленькой буквы).

Для подготовки данных нужно проверить несколько моментов:

- проверить данные на количество пропусков;
- проверить данные на наличие дубликатов (явных или неявных).

Задание 3. Поиск пропусков (3 балла)

Выведите общее количество пропусков во всех столбцах данных.

Заполним пропуски в разделе `category` значением «Прочее»:

Python

```python
data['category'] = data['category'].fillna('Прочее')
```

Задание 4. Поиск неявных дубликатов (3 балла)

Заполните пропущенные значения категории `Amount` средним значением по каждой категории трат.

Было.

	name	value
0	A	2
1	A	NaN
2	B	NaN
3	B	4

4	B	3
5	B	1
6	C	3
7	C	NaN
8	C	1

Стало.

	name	**value**
0	A	2
1	A	2
2	B	4
3	B	4
4	B	3
5	B	1
6	C	3
7	C	3
8	C	1

Выведите среднее значение среди **всех** платежей независимо от категории после удаления пропусков по всему столбцу платежей. В ответе укажите только целую часть числа без округления.

Убираем неявные дубликаты

Неявный дубликат в датафрейме — это строки, которые выглядят почти одинаково, но не совсем идентичны. Например, представьте, что есть таблица с именами, и в ней есть «Георгий» и «георгий». На первый взгляд, они похожи, но из-за разного регистра букв (большие и маленькие) это уже не совсем одно и то же. Такие дубликаты могут мешать анализу данных, потому что можно не заметить, что это одно и то же имя. Важно следить за такими нюансами, чтобы не запутаться! Опечатки тоже могут быть неявными дубликатами, смысловые синонимы тоже (например, «Развлечение» и «На развлечения»).

Посмотрим все уникальные значения в графе категории, чтобы отловить неявные дубликаты. И уберем их для того, чтобы продолжить дальше работу. Подобные ситуации часто встречаются при работе с данными, именно так можно научиться с ними работать.

С помощью `data['category'].unique()` проверим, что названия категорий не содержат грамматических ошибок и не повторяются.

Python

```
data['category'] = data['category'].replace('Прочее', 'Прочие расходы')
data['category'] = data['category'].replace('Комунальные услуги',
    ↪  'Коммунальные услуги')
```

```
data['category'] = data['category'].replace('Продукт', 'Продукты')
data['category'] = data['category'].replace('Абразование', 'Образование')
data['category'] = data['category'].replace('Транспарт', 'Транспорт')
data['category'] = data['category'].replace('Прадукты', 'Продукты')
data['category'] = data['category'].replace('образование', 'Образование')
data['category'].unique()
```

Дополнение данных

Есть данные о датах трат, включая год, месяц, день и время. Георгий решил разобраться, куда уходят его карманные деньги и деньги его родителей, которые могли бы быть его карманными. Он понял, что если извлечь дополнительные сведения из этих данных, то будет проще отслеживать динамику расходов. Например, Георгий заметил, что в выходные он тратит больше на развлечения с друзьями, а в будние дни — на школьные принадлежности или обеды. Так, разделив дату на отдельные части, Георгий получает больше информации о расходах и может лучше понять, как меняются его траты с течением времени. Это помогает ему стать более осознанным в своих финансовых решениях и, возможно, даже сэкономить для чего-то важного!

Python

```
data['year'] = data.index.year
data['month'] = data.index.month
data['day'] = data.index.day
data['dayofweek'] = data.index.dayofweek
data.head()
```

Задание 5. День недели (10 баллов)

Выведите номер дня недели, в который совершали больше всего трат. Напишите программу, которая считает количество вхождений каждого номера дня недели, и выбирает тот, что встречается чаще всего. В ответ выведите номер (десятичную цифру). Если таких значений несколько — выведите наибольшую цифру.

Теперь можно перейти к анализу.

Задание 6. Основы статистики (10 баллов)

Есть данные о ежемесячных расходах семьи Георгия за два года. Задача — найти средний месячный расход в каждой категории: продукты питания, коммунальные услуги, транспорт, развлечения и другие расходы. В ответе выведите число: разность минимального и максимального среднемесячного расхода среди категорий (самой затратной и самой «легкой» категории). Если числа получаются дробными, то выведите только целую часть.

Задание 7. Основы статистики 2 (10 баллов)

Есть данные о ежемесячных расходах семьи Георгия. Задача — найти средний месячный расход в каждой категории в январе: продукты питания, коммунальные услуги, транспорт, развлечения и прочие расходы.

Предположим, что месяц был не из веселых, Георгий только дважды ходил в кино — один раз за 400 руб. и второй раз — за 600 руб., и больше никаких развлечений не было. Средний месячный расход в категории **Развлечения** — 500 руб. $((600 + 400)/2)$.

Зная такое значение в каждой категории, можно эти категории сравнить и вы-

брать ту, где расход был максимальным. А также определить, как для этой категории расходов рядовые траты отклоняются от среднего значения с помощью функции `std()`. В ответе выведите максимальное среднемесячное значение расходов среди всех категорий за январь. Если число получается дробным, то выведите только целую часть.

Задание 8. Основы статистики 3 (10 баллов)

Выведите значение среднеквадратичного отклонения для категории с максимальными среднемесячными расходами. Если число получается дробным, то выведите только целую часть.

Задание 9. Основы статистики 4 (10 баллов)

Георгий заметил, что расходы на продукты в разные месяцы имеют сезонный характер. Он решил запланировать свой бюджет на следующий год, основываясь на данных о расходах за прошлый год. Георгий хочет увеличить свои расходы на Образование (сумму трат) в каждом месяце на 10% в следующем году.

Задача: какой будет общий бюджет Георгия на образование в следующем году с учетом увеличения расходов на 10%? Напишите функцию для расчета общего бюджета на образование. В ответе укажите итоговую сумму в рублях. Если числа получаются дробными, то выведите только целую часть.

Задание 10. Сравнение средних квартальных расходов (10 баллов)

Георгий решил проанализировать свои расходы на продукты за год, чтобы понять, в каком квартале семья тратит больше всего, и сделать соответствующие выводы!

Найдите средние расходы Георгия на продукты за год и определите, на сколько рублей расходы в первом квартале превышают средние расходы по году.

Напишите функцию для расчета общего бюджета на продукты. В ответе укажите итоговую сумму в рублях. Если числа получаются дробными, то выведите только целую часть — округлите вниз.

- 1 квартал: январь, февраль и март.
- 2 квартал: апрель, май и июнь.
- 3 квартал: июль, август и сентябрь.
- 4 квартал: октябрь, ноябрь и декабрь.

Задание 11. Определение тенденции расходов и планирование экономии (10 баллов)

Георгий хочет проанализировать свои ежемесячные расходы за последний год в категории **Развлечения**, чтобы выявить тенденцию и определить, сколько он может сэкономить в следующем месяце. Он планирует веселиться чуть бюджетнее и установить цель по экономии — уменьшить свои расходы на 15% от среднего значения. Задача:

1. Вычислите средние ежемесячные расходы за год по категории **Развлечения** (расходы за каждый месяц).
2. Определите, сколько Георгий может сэкономить, установив цель по экономии в 15% от рассчитанного среднего значения (Георгий сокращает расходы в каждом месяце).
3. Выведите сумму, которую Георгий сможет сэкономить, если будет реже ходить

в кино и сам варить шарики для бабл-чая.

Задание 12. Крупная трата — сумма (10 баллов)

Однажды Георгий задумался, почему в конце января родители урезают его карманные расходы, которые он копит, чтобы купить Tesla. Куда вообще уходят деньги семьи в январе?! Ведь Георгий не ездит в школу, завтракает дома и целыми днями играет в видеоигры. Да и родители отдыхают дома и не тратятся на дорогу и обеды в столовой. Георгий что-то слышал про то, что за выходные не платят, но не может же все быть так просто. Наверное, родители тайно покупают себе что-то вкусное, а потом страдает семейный бюджет и финансовые цели Георгия.

Напишите программу, которая определяет, какая была наибольшая трата за январь. В ответе укажите итоговую сумму в рублях. Если числа получаются дробными, то в ответы впишите только целую часть.

Задание 13. Крупная трата — категория (10 баллов)

Выведите категорию самой крупной траты из предыдущего задания. Категорию необходимо написать ровно так, как она отображается.

Пример решения

Ниже приведен шаблон (пример) кода, который необходимо загрузить в качестве решения. Пример является корректным с точки зрения формата (так как содержит ровно 13 команд вывода ответов), но оценивается в 0 баллов. Ваша задача — написать код аналогичный по структуре, но выводящий правильные ответы на задания.

Python

```python
import pandas as pd

data = pd.read_csv('expenses.csv', index_col=[0], parse_dates=[0])
# TODO Напишите тут свой код, вычисляющий правильные ответы
#  и выведите эти ответы в том же порядке, в котором они указаны в условии задачи

print("unknown") #todo Заменить своим кодом, это неправильный ответ на 1 вопрос
print("unknown") #todo Заменить своим кодом, это неправильный ответ на 2 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 3 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 4 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 5 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 6 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 7 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 8 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 9 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 10 вопрос
```

```
print(0) #todo Заменить своим кодом, это неправильный ответ на 11 вопрос
print(0) #todo Заменить своим кодом, это неправильный ответ на 12 вопрос
print("unknown") #todo Заменить своим кодом, это неправильный ответ на 13 вопрос
```

Решение

Ниже представлено решение на языке Python.

Python

```python
import pandas as pd

data = pd.read_csv('expenses_var_1.csv', index_col=[0], parse_dates=[0])
task1 = str(data['category'].dtype)
task2 = str(data['amount'].dtype)
task3 = data.isna().sum().sum()

data['category'] = data['category'].fillna('Прочее')
data['amount'] = data['amount'].fillna(data.groupby('category')['amount'].transform('mean' ))
task4 = math.floor(data['amount'].mean())

data['category'] = data['category'].replace('Прочее', 'Прочие расходы')
data['category'] = data['category'].replace('Комунальные услуги', 'Коммунальные услуги')
data['category'] = data['category'].replace('Продукт', 'Продукты')
data['category'] = data['category'].replace('Абразование', 'Образование')
data['category'] = data['category'].replace('Транспарт', 'Транспорт')
data['category'] = data['category'].replace('Прадукты', 'Продукты')
data['category'] = data['category'].replace('образование', 'Образование')
#data['category'].unique()

data['year'] = data.index.year
data['month'] = data.index.month
data['day'] = data.index.day
data['dayofweek'] = data.index.dayofweek
#data.head()

task5 = data['dayofweek'].mode().max()
task6 = math.floor(data.groupby('category')['amount'].mean().max() - data.groupby('category')['amount'].mean().min())
task7 = math.floor(data.loc[data['month'] == 1].groupby('category')['amount'].mean().max())
task8 = math.floor(data.loc[(data['month'] == 1) & (data['category'] == 'Продукты')]['amount'].std())

budget = 0
data_cat_month = data.loc[data['category'] == 'Образование'].groupby('month').sum()
for i in data_cat_month['amount']:
                        budget += i*1.1
task9 = math.floor(budget)


```

```
39
40  def kvartal(df):
41      if df['month'] in [1, 2, 3]:
42          return 1
43      if df['month'] in [4, 5, 6]:
44          return 2
45      if df['month'] in [7, 8, 9]:
46          return 3
47      return 4
48
49  data['kvartal'] = data.apply(kvartal, axis = 1)
50  data_products = data.loc[data['category'] == 'Продукты']
51  mean_kvartal = data_products.loc[data_products['kvartal'] ==
    ↪  2]['amount'].mean()
52  full_mean = data_products['amount'].mean()
53  task10 = math.floor(mean_kvartal - full_mean)
54
55  task11 = math.floor((data.loc[data['category'] ==
    ↪  'Развлечения'].groupby('month').mean()['amount']*0.15).sum())
56
57  max_amount_yan = data.loc[data['month'] == 1]['amount'].max()
58  task12 = math.floor(data.loc[(data['month'] == 1) & (data['amount'] ==
    ↪  max_amount_yan)]['amount'])
59
60  cat = data.loc[(data['month'] == 1) & (data['amount'] ==
    ↪  max_amount_yan)]['category']
61  task13 = str(cat).split()[3]
62
63  # TODO Напишите тут свой код, вычисляющий правильные ответы
64  #  и выведите эти ответы в том же порядке, в котором они указаны в
    ↪  условии задачи
65
66  print(task1)
67  print(task2)
68  print(task3)
69  print(task4)
70  print(task5)
71  print(task6)
72  print(task7)
73  print(task8)
74  print(task9)
75  print(task10)
76  print(task11)
77  print(task12)
78  print(task13)
```

Ответ: для набора данных 1: https://disk.yandex.ru/d/KrD3mMJ1_6ds3Q/expenses_var_1.csv

```
1  object
2  float64
3  30
4  10198
5  4
6  1359
7  11060
8  2890
9  1856189
10 436
```

```
19665
19689
Образование
```

Для набора данных 2: https://disk.yandex.ru/d/KrD3mMJ1_6ds3Q/expenses_var_2.csv

```
object
float64
31
9799
4
530
13313
4317
1727901
665
17521
19890
Продукты
```

Для набора данных 3: https://disk.yandex.ru/d/KrD3mMJ1_6ds3Q/expenses_var_3.csv

```
object
float64
26
10380
5
1283
10422
5896
2074305
194
19555
18666
Развлечения
```

Критерии оценивания

Решение задачи должно считать данные из CSV-файла и вывести в консоль ответы на **13 заданий**, представленных выше. Каждый ответ должен быть выведен на отдельной строке (например, функцией `print()`).

Задача будет проверяться на трех **разных** файлах с данными, каждый из которых будет доступен вашему решению в файле `expenses.csv`.

Таким образом, решение при проверке будет запущено три раза, каждый раз на входе будет доступен новый (по содержимому) файл с одним и тем же именем `expenses.csv`. После чтения и обработки файла решение должно вывести 13 строк с ответами и завершиться.

За каждый такой тест на отдельном файле данных можно получить до **100 баллов**.

Итоговый балл является суммой баллов за три теста — то есть максимально за

задачу можно получить **300 баллов**.

Задача 3.2.1.6. Образовательный кредит (100 баллов)

Темы: работа с данными, финансовая грамотность, программирование.

Условие

Для того кто планирует быть топовым программистом, очень важно отличное образование. Георгий это понимает. Для оптимального распределения финансов в его жизни выгоднее поступить на бюджет. Но случиться может всякое. И Георгий решил посчитать, во что ему могут обойтись последствия залипания в интернете — образовательный кредит, чтобы оплатить обучение в университете на контракте. Он хочет понять, как будет изменяться его задолженность по кредиту в зависимости от ежемесячных платежей и процентной ставки. Кредит выплачивается равными **аннуитетными платежами** (изучите, что это, и найдите формулу расчета).

На вход функции должны подаваться следующие данные в одну строку через пробел:

- сумма кредита;
- годовая процентная ставку (в процентах);
- срок кредита в годах (не менее двух лет).

Напишите программу на Python, которая рассчитывает:

- общую сумму, которую Георгий должен будет вернуть по кредиту с учетом процентов;
- общую сумму процентов, которые он заплатит за весь срок кредита;
- остаток долга на конец каждого месяца.

В качестве ответа выведите через пробел общую сумму, общую сумму процентов и остаток долга на 14-й месяц выплаты. Для написания функции необходимо использовать язык программирования Python. Для того чтобы ответ был засчитан корректно, функция должна выводить **только** целую часть числа без округления (!).

Формат входных данных

В единственной строке три целых числа через пробел — сумма кредита, годовая процентная ставка (в процентах), срок кредита в годах (не менее двух лет).

Формат выходных данных

В единственной строке три целых числа через пробел — общая сумма, общая сумма процентов и остаток долга на 14-й месяц выплаты.

Примеры

Пример №1

Стандартный ввод
550000 12 3
Стандартный вывод
657643 107643 359153

Решение

Ниже представлено решение на языке Python.

Python

```python
import math

# Функция для расчета аннуитетного платежа
def calculate_annuity_payment(S, r, n):
    r_month = r/ 100 / 12  # Месячная процентная ставка
    A = S * (r_month * (1 + r_month) ** n) / ((1 + r_month) ** n - 1)
    return A

# Функция для расчета остатка долга на конец k-го месяца
def calculate_remaining_debt(A, r, n, k):
     r_month = r / 100/ 12  # Месячная процентная ставка
     B_k = A * ((1 + r_month) ** n - (1 + r_month) ** k) / r_month
     return B_k

# Входные данные
S, r, n = (int(x) for x in input().split())  # Сумма кредита
n =n*12

A = calculate_annuity_payment(S, r, n)
total_payment = math.floor(A * n)
total_interest = math.floor(total_payment - S)
# Остаток долга на конец 14-го месяца
debt_after_14_months = math.floor(calculate_remaining_debt(A, r, n,
    14))
print(total_payment, total_interest, debt_after_14_months)
```

3.2.2. Командные задачи

Введение

Георгий подумал, что для бóльшего контроля хорошо бы сохранить сведения о тратах в базу данных. И решил разобраться, как это сделать. Он собирается создать таблицу в базе данных с помощью Python, чтобы хранить информацию о расходах семьи. Он начал с изучения основ работы с базами данных и библиотек, которые могут помочь ему в этом процессе.

Создание таблиц в SQL

SQL (Structured Query Language) — это язык программирования, который используется для управления и манипуляции данными в реляционных базах данных. Одной из основных операций в SQL является создание таблиц, которые служат для хранения данных в структурированном виде.

Основные элементы создания таблицы

- Команда `CREATE TABLE`: используется для создания новой таблицы в базе данных.
- Имя таблицы: указывается сразу после команды `CREATE TABLE` и должно быть уникальным в пределах базы данных.
- Поля таблицы: каждое поле (или столбец) описывается с указанием имени и типа данных. Тип данных определяет, какой вид информации может храниться в этом поле (например, целые числа, дробные числа, текст и т. д.).
- Первичный ключ (`PRIMARY KEY`): это поле или набор полей, которые уникально идентифицируют каждую запись в таблице. Обычно используется для обеспечения уникальности записей.
- Автоинкремент (`AUTOINCREMENT`): специальный атрибут для целочисленных полей, который позволяет автоматически увеличивать значение при добавлении новых записей.

Полезные ссылки для изучения

- https://www.w3schools.com/sql/sql_create_table.asp. Этот ресурс предлагает простое и понятное объяснение команды `CREATE TABLE`, а также примеры использования.
- https://www.tutorialspoint.com/sql/sql-create-table.htm. Здесь можно найти более детальную информацию о создании таблиц, включая синтаксис и различные типы данных, используемые в SQL.

Эти ресурсы помогут Георгию и его друзьям лучше понять основы создания таблиц в SQL и начать работу с базами данных.

Выбор базы данных

Георгий выбрал SQLite как систему управления базами данных, поскольку она проста в использовании и не требует установки отдельного сервера. SQLite идеально подходит для небольших проектов и позволяет хранить данные в одном файле.

Установка необходимых библиотек

Для работы с SQLite в Python Георгий использовал встроенный модуль `sqlite3`, который позволяет взаимодействовать с базой данных. Ему не нужно

было устанавливать дополнительные библиотеки, так как `sqlite3` уже включен в стандартную библиотеку Python.

Импортирование библиотеки

Георгий начал с импорта модуля `sqlite3` в свой скрипт:

SQL

```
import sqlite3
from sqlite3 import connect
```

Создание подключения к базе данных

Затем он создал подключение к базе данных. Если базы данных не существовало, `sqlite3` автоматически создал бы новый файл базы данных:

SQL

```
conn = sqlite3.connect('transactions.db')
```

Создание курсора

Георгий создал объект курсора, который позволяет выполнять SQL-запросы:

SQL

```
cursor = conn.cursor()
```

Создание таблицы

Теперь Георгий хочет попробовать создать тестовую таблицу для хранения данных о финансах. Он определил структуру таблицы, включающую такие поля, как `id`, `loan_amount`, `loan_term`, `monthly_payment`, и `remaining_balance`. Он использовал SQL-запрос для создания таблицы:

SQL

```
cursor.execute('''
CREATE TABLE IF NOT EXISTS loans (
id INTEGER PRIMARY KEY AUTOINCREMENT,
loan_amount REAL,
loan_term INTEGER,
monthly_payment REAL,
remaining_balance REAL
)
''')
```

В таблице `loans` Георгий определил несколько полей, каждое из которых имеет свое назначение и тип данных. Вот описание каждого поля и обоснование выбора типов данных:

- `id INTEGER PRIMARY KEY AUTOINCREMENT`
  - ◇ Описание: это уникальный идентификатор для каждой записи в таблице. Он автоматически увеличивается при добавлении новой записи.
  - ◇ Тип данных: `INTEGER`.
  - ◇ Обоснование: использование целочисленного типа данных для идентификатора позволяет легко отслеживать и ссылаться на записи. `AUTOINCREMENT` гарантирует, что каждый новый идентификатор будет уникальным и не повторится.
- `loan_amount REAL`
  - ◇ Описание: это сумма кредита, которую Георгий взял.
  - ◇ Тип данных: `REAL`.
  - ◇ Обоснование: `REAL` используется для хранения чисел с плавающей запятой, что позволяет точно представлять суммы, включая дробные значения. Это важно для финансовых данных, так как суммы кредитов могут быть нецелыми.
- `loan_term INTEGER`
  - ◇ Описание: это срок кредита в месяцах.
  - ◇ Тип данных: `INTEGER`.
  - ◇ Обоснование: срок кредита выражается в целых числах (например, 12 месяцев, 24 месяца и т. д.), поэтому для этого поля используется целочисленный тип данных.
- `monthly_payment REAL`
  - ◇ Описание: это сумма ежемесячного платежа по кредиту.
  - ◇ Тип данных: `REAL`.
  - ◇ Обоснование: как и для суммы кредита, `REAL` позволяет хранить значения с плавающей запятой, что важно для точного представления ежемесячных платежей, которые могут включать дробные значения.
- `remaining_balance REAL`
  - ◇ Описание: это остаток по кредиту, который еще нужно выплатить.
  - ◇ Тип данных: `REAL`.
  - ◇ Обоснование: остаток по кредиту также может быть дробным числом, поэтому для этого поля выбран тип `REAL`, что позволяет точно отслеживать оставшуюся сумму долга.

Добавление данных

Георгий добавил несколько записей в таблицу:

SQL

```
cursor.execute('''
INSERT INTO loans (loan_amount, loan_term, monthly_payment,
↪  remaining_balance)
VALUES
(500000, 36, 15000, 450000),
(300000, 24, 12000, 200000),
(1000000, 60, 25000, 950000),
(200000, 12, 17000, 150000),
```

```
(400000, 24, 18000, 350000),
(750000, 48, 22000, 700000),
(550000, 36, 16000, 500000),
(900000, 48, 23000, 850000),
(1200000, 60, 27000, 1150000),
(600000, 36, 19000, 550000);
''')
query = '''
SELECT * FROM loans
'''
df_sql = pd.read_sql(query, conn)
df_sql
```

Сохранение изменений и закрытие подключения

После создания таблицы и внесения данных Георгий сохранил изменения и закрыл соединение с базой данных:

SQL

```
conn.commit()
conn.close()
```

Георгий успешно создал таблицу в базе данных, что позволило ему хранить и управлять данными о своем образовательном кредите. Он понял, что использование баз данных значительно упрощает процесс работы с данными и позволяет легко их организовывать и извлекать. Теперь он мог добавлять, изменять и запрашивать информацию о своих финансах, что помогло ему лучше планировать свои расходы. И потом использовать, если ему захочется разработать приложение, которое будет пользоваться этой базой.

Задача 3.2.2.1. Базы данных 1 (100 баллов)

Темы: программирование, работа с базами данных, SQL, финансовая грамотность.

Условие

Георгию необходимо создать таблицу для учета своих исходных расходов и доходов. Таблица должна содержать следующие поля.

- Дата (date): дата транзакции.
- Сумма (amount): сумма транзакции.
- Категория (category): категория траты.

Данные возьмем из файла `expenses_db.csv`.

SQL

```
conn = sqlite3.connect('expenses.db')
cursor = conn.cursor()
data_to_sql = pd.read_csv('expenses_db.csv')
data_to_sql.to_sql('expenses', conn, if_exists='replace', index=False)
```

Проверим, что все сохранено корректно.

SQL

```
query = '''
    SELECT * FROM expenses LIMIT 10;
'''
df_sql = pd.read_sql(query, conn)
df_sql
```

Задание: напишите запрос и выведите количество строк в запросе

SQL

```
SELECT * FROM название_таблицы
```

Решение

Запрос:

SQL

```
cursor.execute('SELECT count(*) FROM expenses')
result = cursor.fetchone()
print(result[0])
```

Ответ: результат запроса: 1 000.

Критерии оценивания

- Запросы написаны верно, дан верный ответ — 100.
- Запросы написаны верно, неверный формат вывода ответа (при этом ответ верный) — 75.
- Запросы написаны верно, нет вывода ответа — 50.
- Запросы написаны неверно, не написаны, код не запускается, представлен только ответ, ответ неверный — 0.

Задача 3.2.2.2. Базы данных 2 (100 баллов)

Темы: программирование, работа с базами данных, SQL, финансовая грамотность.

Условие

Напишите SQL-запрос, который посчитает, сколько было платежей (независимо от категории) выше 2 000 руб. по вторникам.

Решение

Запрос:

SQL

```sql
cursor.execute(
    '''SELECT count(*) FROM expenses
    WHERE amount > 2000 AND dayofweek = 1'''
)
result = cursor.fetchone()
print(result[0])
```

Ответ: 145.

Примечание. Обратите внимание на нумерацию дней недели в базе данных.

Критерии оценивания

- Запросы написаны верно, дан верный ответ — 100.
- Запросы написаны верно, неверный формат вывода ответа (при этом ответ верный) — 75.
- Запросы написаны верно, нет вывода ответа — 50.
- Запросы написаны неверно, не написаны, код не запускается, представлен только ответ, ответ неверный — 0.

Задача 3.2.2.3. Базы данных 3 (100 баллов)

Темы: программирование, работа с базами данных, SQL, финансовая грамотность.

Условие

Выведите количество записей о расходах из категорий «Коммунальные услуги», «Продукты», «Транспорт».

Решение

Запрос:

SQL

```sql
cursor.execute(
    '''SELECT count(*) FROM expenses
    WHERE category IN ('Коммунальные услуги',
     'Продукты',
    'Транспорт');'''
)
result = cursor.fetchone()
print(result[0])
```

Критерии оценивания

- Запросы написаны верно, дан верный ответ — 100.

- Запросы написаны верно, неверный формат вывода ответа (при этом ответ верный) — 75.
- Запросы написаны верно, нет вывода ответа — 50.
- Запросы написаны неверно, не написаны, код не запускается, представлен только ответ, ответ неверный — 0.

Задача 3.2.2.4. Интерфейс для базы данных (100 баллов)

Темы: программирование, работа с базами данных, SQL, финансовая грамотность, разработка интерфейсов.

Условие

В этой задаче Георгию потребуются библиотеки `ipywidgets` и `IPython.display`.

Python

```python
import ipywidgets as widgets
from IPython.display import display, clear_output
```

Используя библиотеку `ipywidgets`, нарисуйте интерфейс, который показан на рис. 3.2.1. Интерфейс должен выводить запись под установленные параметры пользователя.

Необходима опция выбора даты и категории расхода. Вывод данных осуществляется по нажатию кнопки **Вывести расходы**.

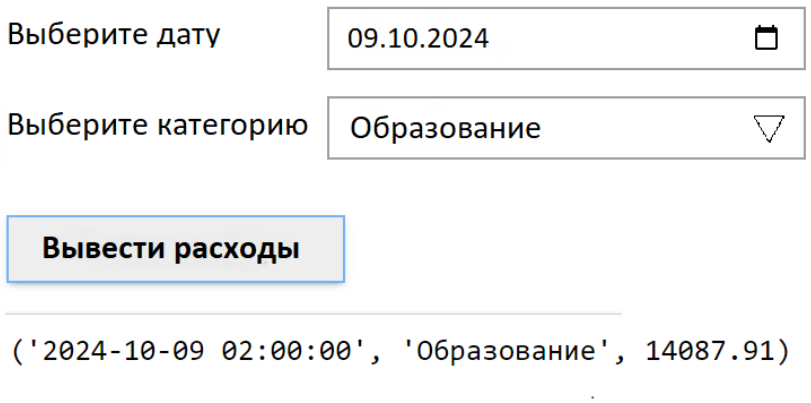


Рис. 3.2.1. Изображение виджета, который необходимо разработать (выбор даты и выпадающий список — категория расходов)

Вывод данных является лишь примером вывода.

Решение

Ниже представлено решение на языке Python.

Python

```python
import sqlite3
import pandas as pd
import ipywidgets as widgets
from IPython.display import display

conn = sqlite3.connect('expenses.db')
cursor = conn.cursor()
data_to_sql = pd.read_csv('expenses_db.csv')
data_to_sql.to_sql('expenses', conn, if_exists='replace', index=False)

query = "SELECT DISTINCT category FROM expenses;"
response = pd.read_sql(query, conn)
categories = response['category']

date_w = widgets.DatePicker(
    description='Выберите дату: ',
    disabled=False,
)
cat_w = widgets.Dropdown(
    options=categories ,
    value=categories [0],
    description='Выберите категорию:',
    disabled=False,
)
button = widgets.Button(
    description='Вывести расходы',
    disabled=False,
    tooltip='Вывести расходы',
)
output = widgets.Output(layout={'border': '1px solid black'})

display(date_w, cat_w, button, output)
def on_button_clicked(b):
    output.clear_output()
    if date_w.value:
        query = f'''
SELECT date, amount, category
FROM expenses
WHERE category = '{cat_w.value}'
AND year={date_w.value.year}
AND month={date_w.value.month}
AND day={date_w.value.day};
        '''
        result = pd.read_sql(query, conn)
        with output:
            for index, row in result.iterrows():
                print(row['date'], row['category'], row['amount'])
    else:
        with output:
            print("Не указана дата")

button.on_click(on_button_clicked)
```

Ответ: вариант отображения и вывода.

```
Дата: 14 октября 2024 года
```

Вывод:

```
2024-10-14 08:55:17 Развлечения 14802.74
2024-10-14 20:04:38 Развлечения 12882.39
2024-10-14 07:21:21 Развлечения 877.85
2024-10-14 01:52:05 Развлечения 14077.49
```

Отображение интерфейса на рис. 3.2.2.

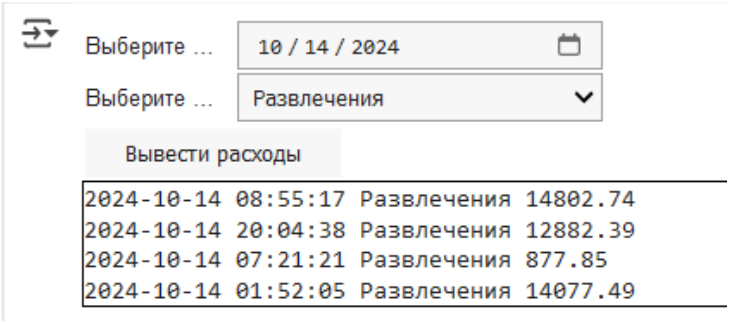


Рис. 3.2.2

Критерии оценивания

Оценка формируется путем суммирования двух баллов — за отрисовку интерфейса (виджета) и за взаимодействие интерфейса с данными.

- Балл за отрисовку интерфейса — от 0 до 25.
  - ◇ Виджет отрисован и отображается корректно по заданию — 25.
  - ◇ Виджет отрисован схоже, но есть неточности (например, выбор категории — не выпадающий список) — 10.
  - ◇ Виджет отрисован некорректно, требует перезагрузки кода для повторного нажатия и/или не реагирует на нажатия — 0.
- Балл за взаимодействие интерфейса с данными — от 0 до 75.
  - ◇ Данные отображаются корректно и в полном объеме — 75.
  - ◇ Данные отображаются под запрос, но не все записи (вывод частичный), сформирован неполный список категорий — 50.
  - ◇ Допущены неточности и ошибки в интерфейсе, которые могут повлиять на корректность вывода данных (например, неполный список категорий). В остальном код написан и отрабатывает корректно — 25.
  - ◇ Данные отображаются некорректно — 0.

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

4.2.1. Информатика. 8–11 классы

Задача 4.2.1.1. Олимпиада (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: 256 Мбайт.

Условие

Андрей пришел на олимпиаду, и ему выдали пароль из l символов, который содержал символы `N`, `T`, `O`, `F`, `I`, `N`, `2`, `0`, `2`, `5`. Каждый из них мог встречаться несколько раз или вообще не встречаться ни разу.

Андрей задумался: а какое минимальное число байт потребуется, чтобы сохранить пароли всех c участников, при условии, что на один пароль будет использоваться одинаковое целое число байт?

Формат входных данных

В первой строке дано одно целое число l $(10 \leqslant l \leqslant 50)$ — длина пароля.

Во второй строке дано одно целое число c $(20 \leqslant c \leqslant 200)$ — количество участников олимпиады.

Формат выходных данных

Вывести минимальное число байт, которое потребуется, чтобы сохранить пароли всех участников.

Примеры

Пример №1

Стандартный ввод
31 42

Стандартный вывод
504

Решение

Пароль содержит 8 уникальных символов: N, T, O, F, I, 2, 0, 5. Для кодирования одного символа требуется:

$$\log_2 8 = 3 \text{ бит}.$$

Для пароля длины l:

$$\text{Бит на пароль} = l \cdot 3.$$

1 байт = 8 бит. Чтобы найти целое число байт, округлим вверх:

$$\text{Байт на пароль} = \left\lceil \frac{l \cdot 3}{8} \right\rceil.$$

Эквивалентная формула для целочисленного деления:

$$\text{Байт на пароль} = \frac{l \cdot 3 + 7}{8} \quad \text{(целочисленное деление)}.$$

Умножим байты на пароль на количество участников c:

$$\text{Итого байт} = \left(\left\lceil \frac{3l}{8} \right\rceil \right) \cdot c.$$

Пример программы-решения

Ниже представлено решение на языке Python.

Python

```python
l = int(input())
c = int(input())
print((l * 3 + 7) // 8 * c)
```

Задача 4.2.1.2. Ограбление (18 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: 256 Мбайт.

Условие

Банк представляет собой квадрат 15×15, в каждой ячейке хранится определенное число денег. Василий заранее знает, сколько денег в каждой ячейке.

Изначально Василий оказался в нижнем правом углу банка. Выход из банка в верхнем левом углу. Вася хочет добраться до выхода как можно быстрее, поэтому он будет двигаться от ячейки к ячейке только вверх или влево.

Вася хочет собрать как можно больше денег, но при этом он считает, что он обязан побывать в ячейке, в которой больше всего денег.

Определите, сколько максимум денег он может забрать из банка.

Формат входных данных

В первых 15 строках даны по 15 целых чисел a_{ij} ($1 \leqslant a_{ij} \leqslant 100$) — число денег в ячейке с координатой (i, j). Гарантируется, что максимальное значение ровно в одной ячейке.

Формат выходных данных

В первых 15 строках даны по 15 целых чисел a_{ij} ($1 \leqslant a_{ij} \leqslant 100$) — число денег в ячейке с координатой (i, j). Гарантируется, что максимальное значение ровно в одной ячейке.

Примеры

Пример №1

Стандартный ввод
58 76 81 25 79 27 50 81 34 22 98 75 42 71 83 36 1 22 84 57 22 22 55 22 70 30 19 6 12 66 84 69 73 27 93 37 8 26 41 69 91 10 65 9 40 5 48 8 97 55 91 37 13 64 98 82 36 84 89 36 15 2 3 49 34 22 67 69 20 2 42 98 90 62 91 39 60 12 31 53 31 56 5 45 46 9 6 15 49 67 2 18 16 14 18 86 65 50 72 53 14 85 26 11 84 58 99 46 79 94 56 66 19 54 80 85 13 18 25 56 28 41 49 37 60 27 6 39 54 49 90 82 66 57 52 68 16 39 33 32 96 65 55 67 58 76 62 70 70 64 2 1 69 76 14 76 66 100 52 56 38 29 87 55 40 53 8 9 86 92 13 67 11 40 49 92 14 38 94 71 77 14 1 47 36 75 54 39 34 77 21 90 4 77 18 32 5 37 86 67 94 25 60 94 36 15 85 94 9 96 88 36 83 5 35 65 53 31 32 70 68 71 48 88 96

Стандартный вывод
1951

Примечания

Первый пример.

Таблица 4.2.1

58	76	81	25	79	27	50	81	34	22	98	75	42	71	83
36	1	22	84	57	22	22	55	22	70	30	19	6	12	66
84	69	73	27	93	37	8	26	41	69	91	10	65	9	40
5	48	8	97	55	91	37	13	64	98	82	36	84	89	36

15	2	3	49	34	22	67	69	20	2	42	98	90	62	91
39	60	12	31	53	31	56	5	45	46	9	6	15	49	67
2	18	16	14	18	86	65	50	72	53	14	85	26	11	84
58	99	46	79	94	56	66	19	54	80	85	13	18	25	56
28	41	49	37	60	27	6	39	54	49	90	82	66	57	52
68	16	39	33	32	96	65	55	67	58	76	62	70	70	64
2	1	69	76	14	76	66	100	52	56	38	29	87	55	40
53	8	9	86	92	13	67	11	40	49	92	14	38	94	71
77	14	1	47	36	75	54	39	34	77	21	90	4	77	18
32	5	37	86	67	94	25	60	94	36	15	85	94	9	96
88	36	83	5	35	65	53	31	32	70	68	71	48	88	96

Решение

Для решения задачи разобьем ее на две части:

1. Найдем координаты (x_{max}, y_{max}) ячейки с максимальным значением.
2. Разделим путь на два отрезка:
   - от начальной точки $(15, 15)$ до (x_{max}, y_{max});
   - от (x_{max}, y_{max}) до конечной точки $(1, 1)$.

Пример программы-решения

Ниже представлено решение на языке Python.

Python

```python
def solve(a, start, end):
    x1, y1 = start
    x2, y2 = end
    n = len(a)
    m = len(a[0]) if n else 0

    dp = [[0 for _ in range(m)] for _ in range(n)]
    dp[x1][y1] = a[x1][y1]

    for i in range(x1, x2 + 1):
        for j in range(y1, y2 + 1):
            if i == x1 and j == y1:
                continue
            dp[i][j] = a[i][j] + max(dp[i-1][j] if i > x1 else 0,
             ↪  dp[i][j-1] if j > y1 else 0)

    return dp[x2][y2]

n, m = 15, 15
a = [list(map(int, input().split())) for _ in range(n)]
```

```
20
21  x, y = 0, 0
22  for i in range(n):
23      for j in range(m):
24          if a[x][y] < a[i][j]:
25              x, y = i, j
26
27  print(solve(a, (0, 0), (x, y)) + solve(a, (x, y), (n - 1, m - 1)) -
    ↪  a[x][y])
```

Для каждого из этих отрезков применим динамическое программирование, аналогичное задаче о максимальном пути в матрице с движениями только вверх и влево.

1. Находим позицию максимального элемента (x_{max}, y_{max}).
2. Вычисляем максимальную сумму на пути от $(15, 15)$ до (x_{max}, y_{max}):
   - Создаем матрицу $dp1$ размером 15×15.
   - Инициализируем $dp1[15][15] = a[15][15]$.
   - Заполняем матрицу по правилу:

     $$dp1[i][j] = a[i][j] + \max(dp1[i+1][j], dp1[i][j+1])$$

     (с проверкой границ матрицы).
3. Вычисляем максимальную сумму на пути от (x_{max}, y_{max}) до $(1, 1)$:
   - Создаем матрицу $dp2$ размером 15×15.
   - Инициализируем $dp2[x_{max}][y_{max}] = a[x_{max}][y_{max}]$.
   - Заполняем матрицу по правилу:

     $$dp2[i][j] = a[i][j] + \max(dp2[i-1][j], dp2[i][j-1])$$

     (с проверкой границ матрицы).
4. Итоговый результат: $dp1[x_{max}][y_{max}] + dp2[1][1] - a[x_{max}][y_{max}]$ (вычитаем значение максимальной ячейки, так как оно учтено дважды).

Так как ограничения маленькие, то вместо написания динамики можно написать перебор всех путей от максимальной монеты до старта и до финиша.

Пример программы-решения

Ниже представлено решение на языке Python.

Python

```
1   def solve():
2       # Чтение входных данных
3       grid = []
4       max_val = -1
5       max_pos = (-1, -1)
6
7       for i in range(15):
8           row = list(map(int, input().split()))
9           grid.append(row)
10          for j in range(15):
```

```
            if row[j] > max_val:
                max_val = row[j]
                max_pos = (i, j)

    start = (14, 14)  # Нижний правый угол (индексы с 0)
    end = (0, 0)       # Верхний левый угол

    # Функция для генерации всех возможных сумм до целевой точки
    def generate_path_sums(start_pos, target_pos):
        dx = start_pos[0] - target_pos[0]  # Количество шагов вверх
        dy = start_pos[1] - target_pos[1]  # Количество шагов влево
        total_steps = dx + dy

        sums = {}

        # Перебор всех комбинаций шагов
        from itertools import combinations
        for up_indices in combinations(range(total_steps), dx):
            mask = 0
            for pos in up_indices:
                mask |= 1 << pos

            x, y = start_pos
            current_sum = grid[x][y]
            valid = True

            for i in range(total_steps):
                if mask & (1 << i):
                    x -= 1  # Вверх
                else:
                    y -= 1  # Влево

                if x < 0 or y < 0 or x > 14 or y > 14:
                    valid = False
                    break

                current_sum += grid[x][y]

                if (x, y) == target_pos:
                    break

            if valid and (x, y) == target_pos:
                sums[mask] = current_sum

        return sums

    # Генерируем все суммы для первого отрезка
    first_part = generate_path_sums(start, max_pos)

    # Генерируем все суммы для второго отрезка
    second_part = generate_path_sums(max_pos, end)

    # Находим максимальную комбинацию
    max_sum = 0
    for sum1 in first_part.values():
        for sum2 in second_part.values():
            total = sum1 + sum2 - max_val
            if total > max_sum:
                max_sum = total

```

```
71      print(max_sum)
72
73  solve()
```

Задача 4.2.1.3. Сдача (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: 256 Мбайт.

Условие

Банкир должен выдать на сдачу x руб. Сколькими способами он может это сделать, если у него есть монеты достоинством 1, 2, 5 и 10 руб.

Например, если банкиру надо сдать 25 руб., то существует 64 способа (способы считаются различными, если наборы различаются хотя бы одной монетой; порядок монет неважен).

Формат входных данных

В первой строке дано одно число x ($1 \leqslant x \leqslant 2025$) — сколько сдачи должен выдать банкир.

Формат выходных данных

Выведите количество способов сдать сдачу.

Примеры

Пример №1

Стандартный ввод
25

Стандартный вывод
64

Решение

Метод 1: динамическое программирование

Используем динамическое программирование по сумме. Определим $dp[i]$ как количество способов набрать сумму i заданными монетами.

1. Инициализация: $dp[0] = 1$ (существует один способ выдать 0 руб. — не выдавать ничего).
2. Для каждой монеты достоинством $c \in \{1, 2, 5, 10\}$ для всех сумм i от c до x обновляем значение: $dp[i] += dp[i - c]$.
3. Результат: $dp[x]$.

Пример программы-решения

Ниже представлено решение на языке Python.

Python

```python
n = int(input())
coins = [1, 2, 5, 10]
dp = [0] * (n + 1)
dp[0] = 1
for coin in coins:
    for j in range(coin, n + 1):
        dp[j] += dp[j - coin]

print(dp[n])
```

Метод 2: полный перебор

Перебираем все возможные комбинации монет, сумма которых равна x.

- Для всех возможных количеств монет 10 рублей ($0 \leqslant a \leqslant \lfloor x/10 \rfloor$).
- Для всех возможных количеств монет 5 рублей ($0 \leqslant b \leqslant \lfloor (x - 10a)/5 \rfloor$).
- Для всех возможных количеств монет 2 рублей ($0 \leqslant c \leqslant \lfloor (x - 10a - 5b)/2 \rfloor$).
- Оставшуюся сумму $d = x - 10a - 5b - 2c$ покрываем монетами 1 руб.
- Увеличиваем счетчик способов на единицу для каждой валидной комбинации (a, b, c, d).

Пример программы-решения

Ниже представлено решение на языке Python.

Python

```python
n = int(input())
cnt = 0
for i in range(n // 10 + 1):
    ost = n - i * 10
    for j in range(ost // 5 + 1):
        ost1 = ost - j * 5
        for k in range(ost1 // 2 + 1):
            cnt += 1
print(cnt)
```

Задача 4.2.1.4. Монетки (22 балла)

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: 256 Мбайт.

Условие

Банкир придумал новую игру с монетами. Всего у него есть g золотых монет и s серебряных. Необходимо сложить монеты в стопки по три штуки. Стопки, в которых все три монеты одинаковые, считаются *скучными*. Помогите понять, сколько максимум *нескучных* стопок может составить банкир.

Формат входных данных

Первая строка содержит одно целое число g ($1 \leqslant g \leqslant 10^{12}$) — количество золотых монет.

Вторая строка содержит одно целое число s ($1 \leqslant s \leqslant 10^{12}$) — количество серебряных монет.

Формат выходных данных

Вывести одно целое число — максимальное число не *скучных* стопок по три монеты.

Примеры

Пример №1

Стандартный ввод
3 4
Стандартный вывод
2

Пример №2

Стандартный ввод
3 10
Стандартный вывод
3

Решение

Разберем несколько случаев.

Если $g > s$, то можно их поменять местами, количество стопок от этого не изменится. Если $g < s/2$, то можно сделать только g стопок.

Если разница не столь велика, можно создать $g - s$ стопок, выровняв количество золотых и серебряных монет, и затем можно делать стопки: 2 золотые монеты и 1 серебряная, и 2 серебряные монеты и 1 золотая. Когда в каждом типе останется менее 3, в случае остатка 2 можно сделать еще одну стопку.

Пример программы-решения

Ниже представлено решение на языке Python.

Python

```python
n = int(input())
m = int(input())
if n < m:
    n, m = m, n
if n >= 2 * m:
    print(m)
else:
    print((n - m) + (2 * m - n) // 3 * 2 + (1 if (2 * m - n) % 3 > 1
    ↪  else 0))
```

Задача 4.2.1.5. Инвестор (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: 256 Мбайт.

Условие

Начинающий инвестор Василий узнал о *двоичных* финансовых инструментах, которые считаются надежными. Их номиналы состоят только из цифры 2, например, 2, 22, 222 и так далее. Василий хочет вложить сумму x, используя только эти инструменты. Сумма вложения равна произведению номиналов выбранных инструментов (каждый номинал можно использовать несколько раз, например, $8 = 2 \cdot 2 \cdot 2$, $44 = 22 \cdot 2$). Однако не все x возможно представить таким образом, поэтому Василий хочет инвестировать сумму y (где $x \leqslant y$), которую можно разложить на произведение *двоичных* номиналов.

Помогите Василию найти наименьшее число y, удовлетворяющее условиям.

Формат входных данных

В первой строке содержится одно целое число x ($2 \leqslant x \leqslant 10^{18}$).

Формат выходных данных

Вывести одно целое число y, которое можно представить в виде произведения *двоичных* номиналов.

Примеры

Пример №1

Стандартный ввод
70
Стандартный вывод
88

Решение

С помощью рекурсии сгенерируем все числа, которые можно получить в качестве произведения чисел, состоящих только из двоек. Таких чисел немного. После этого находим наименьшее число y, которое можно представить в виде произведения *двоичных* чисел.

Пример программы-решения

Ниже представлено решение на языке Python.

Python

```python
correct = set()
all = []
for i in range(18):
    all.append(int("2" * (i + 1)))

def gen(cur, last):
    global correct
    global all
    if cur > int(1e19):
       return
    correct.add(cur)
    for i in range(last, 18):
        gen(cur * all[i], i)

x = int(input())
gen(1, 0)
correct = sorted(list(correct))

for y in correct:
```

```
20      if y >= x:
21          print(y)
22          break
```

4.2.2. Математика. 8–9 классы

Задача 4.2.2.1. (15 баллов)

Темы: алгебра, неравенства.

Условие

Пусть $2 < a < b$. Найдите наибольшее значение b, при котором верно неравенство

$$a^2b - ab^2 - 2(a^2 - b^2) - 3a^2 + 3ab + 10a - 10b \geqslant 0.$$

Решение

Разложим левую часть неравенства на множители. Имеем

$$\begin{aligned}
&a^2b - ab^2 - 2(a^2 - b^2) - 3a^2 + 3ab + 10a - 10b = \\
&= ab(a - b) - 2(a - b)(a + b) - 3a(a - b) + 10(a - b) = \\
&= (a - b)(ab - 2a - 2b - 3a + 10) = \\
&= (a - b)(ab - 5a - 2b + 10) \\
&= (a - b)(a(b - 5) - 2(b - 5)) = (a - b)(a - 2)(b - 5) = (b - a)(a - 2)(5 - b).
\end{aligned}$$

Так как $b > a > 2$, то первые два множителя положительны. Следовательно, требуется найти такое b, что $5 - b \geqslant 0$. Тогда наибольшее такое b равно числу 5.

Ответ: 5.

Задача 4.2.2.2. (20 баллов)

Темы: делимость, остатки от деления.

Условие

Числа 1 765, 1 950, 1 543 дают одинаковый остаток при делении на некоторое натуральное число k, где $k \geqslant 2$. Найдите этот остаток.

Решение

Так как данные числа при делении на k дают одинаковый остаток, то разность любых двух из них делится на k. Имеем

$$1\,765 - 1\,543 = 222,$$
$$1\,950 - 1\,765 = 185.$$

Числа 222 и 185 делятся на k, следовательно, их разность, равная $222 - 185 = 37$, тоже делится на k. Так как 37 — простое число, то $k = 37$.

Разделив, например, 1 543 на 37, получим искомый остаток 26.

Ответ: 26.

Задача 4.2.2.3. (20 баллов)

Темы: подобные треугольники, параллелограмм, площади фигур.

Условие

На сторонах параллелограмма $ABCD$ отмечены точки A_1, B_1, C_1, D_1 так, что $AA_1 : A_1B = BB_1 : B_1C = CC_1 : C_1D = DD_1 : D_1A = 2:1$. Найдите отношение площади параллелограмма $ABCD$ к площади закрашенной фигуры (см. рис. 4.2.1).

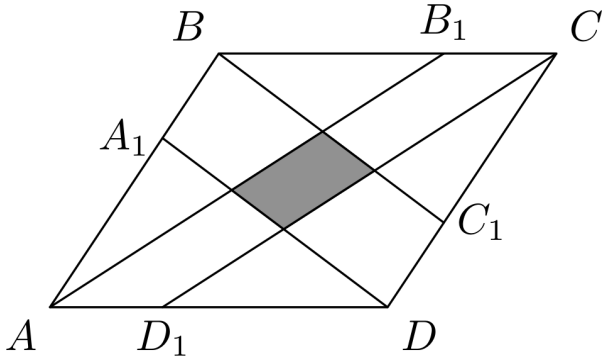


Рис. 4.2.1

Решение

Пусть $MNPQ$ — закрашенный четырехугольник (см. рис. 4.2.2).

Треугольники ABB_1 и CDD_1 равны, так как $AB = CD$ (противоположные стороны параллелограмма), $BB_1 = DD_1$ (составляют одинаковую часть от противоположных сторон параллелограмма), $\angle ABC = \angle CDA$ (свойство параллелограмма).

Тогда $\angle BB_1A = \angle CD_1D$.

Так как $BC \parallel AD$, то $AB_1 \parallel CD_1$.

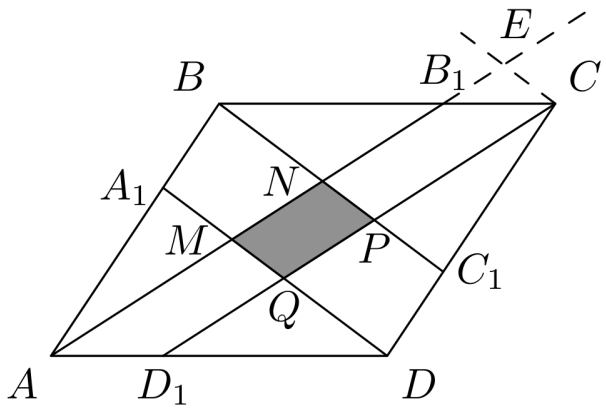


Рис. 4.2.2

Аналогично $BC_1 \parallel DA_1$. Следовательно, $MNPQ$ — параллелограмм.

Проведем через точку C прямую, параллельную BC_1. Пусть E — точка пересечения построенной прямой и прямой AB_1.

Треугольники BB_1N и CB_1E подобны по двум углам ($\angle BB_1N = \angle EB_1C$, $\angle NBB_1 = \angle ECB_1$), коэффициент подобия равен $BB_1 : B_1C = 2$. Тогда их площади отличаются в 4 раза. Пусть $S_{CB_1E} = x$. Тогда $S_{BB_1N} = 4x$.

Треугольники BCP и BB_1N подобны по двум углам ($\angle PBC$ общий, $\angle BCP = \angle BB_1N$), коэффициент подобия равен $BC : BB_1 = 3 : 2$. Тогда их площади отличаются в $\dfrac{9}{4}$ раза. Следовательно, $S_{PNB_1C} = 5x$.

Параллелограммы $MNPQ$ и $PNEC$ имеют одинаковую высоту, опущенную из точки N на прямую D_1C. Тогда их площади относятся как основания CP и PQ. По теореме Фалеса $CP : PQ = CC_1 : C_1D = 2$. Поскольку

$$S_{PNEC} = S_{PNB_1C} + S_{CB_1E} = 5x + x = 6x,$$

получаем, что $S_{MNPQ} = 3x$. То есть $S_{BCP} = 9x = 3S_{MNPQ}$.

Аналогично $S_{CDQ} = S_{DAM} = S_{ABN} = 3S_{MNPQ}$. Тогда

$$\frac{S_{ABCD}}{S_{MNPQ}} = \frac{S_{MNPQ} + S_{ABN} + S_{BCP} + S_{CDQ} + S_{DAM}}{S_{MNPQ}} = 1 + 4 \cdot 3 = 13.$$

Ответ: 13.

Задача 4.2.2.4. (20 баллов)

Темы: иррациональное уравнение, квадратное уравнение, замена переменной.

Условие

Найдите наибольший корень уравнения

$$4x^2 + 2x - 8 = 3\sqrt{x^4 - x^2 + 4}.$$

Решение

Попробуем найти положительные корни (если они есть, то неположительные нас уже не интересуют). Вынесем за скобку $4x$ в левой части, а в правой под корнем вынесем величину x^2, которую затем вынесем из-под корня. Имеем

$$4x\left(x - \frac{2}{x} + \frac{1}{2}\right) = 3x\sqrt{x^2 + \frac{4}{x^2} - 1}.$$

Разделив обе части на $x > 0$, получаем

$$4\left(x - \frac{2}{x} + \frac{1}{2}\right) = 3\sqrt{x^2 + \frac{4}{x^2} - 1}.$$

Положим $t = x - \frac{2}{x}$. Тогда $t^2 = x^2 + \frac{4}{x^2} - 4$. Следовательно, $x^2 + \frac{4}{x^2} = t^2 + 4$. Тогда уравнение равносильно

$$4\left(t + \frac{1}{2}\right) = 3\sqrt{t^2 + 3}. \tag{4.2.1}$$

Возведя обе части уравнения в квадрат, находим

$$16t^2 + 16t + 4 = 9t^2 + 27.$$

То есть

$$7t^2 + 16t - 23 = 0.$$

Корень $t = -\frac{23}{7}$ не обращает уравнение (4.2.1) в верное равенство, а корень $t = 1$ подходит. Значит,

$$x - \frac{2}{x} = 1 \quad \Longleftrightarrow \quad x^2 - x - 2 = 0.$$

Из корней $x = -1$, $x = 2$ нам подходит только положительный. Таким образом, $x = 2$.

Ответ: 2.

Задача 4.2.2.5. (25 баллов)

Темы: логика, графы.

Условие

В сообществе некоторой социальной сети, состоящем из двадцати человек, сложилась следующая ситуация. У девяти членов этого сообщества оказалось ровно по три друга, также входящих в это сообщество. А у остальных оказалось ровно по n друзей, также входящих в это сообщество. Определите наибольшее возможное значение n.

Решение

Представим сообщество в виде графа, вершины которого соответствуют членам сообщества, а ребра — дружбе между ними.

Разобьем вершины на два класса: «группа девяти» — вершины, имеющие степень 3, и «группа одиннадцати» — вершины, имеющие степень n.

Докажем, что $n < 13$. Пусть число n принимает значение хотя бы 13. Рассмотрим одну вершину из «группы одиннадцати». Эта вершина может быть соединена максимум с десятью вершинами «группы одиннадцати». Значит, эта вершина соединена хотя бы с тремя вершинами из «группы девяти». Следовательно, вершины «группы одиннадцати» добавляют к общей сумме степеней вершин «группы девяти» хотя бы $11 \cdot 3 = 33$. Но сумма степеней вершин «группы девяти» равна $9 \cdot 3 = 27$. Получили противоречие.

Так как каждое ребро графа добавляет к общей сумме степеней вершин двойку, то общая сумма степеней четна. Тогда число n не может быть равным 12, поскольку иначе сумма степеней будет равна нечетному числу $11 \cdot 12 + 9 \cdot 3$. Таким образом, $n \leqslant 11$.

Приведем пример, когда $n = 11$. «Группа одиннадцати» обозначена на рисунке овалом, а вершины «группы девяти» отмечены пронумерованными кругами.

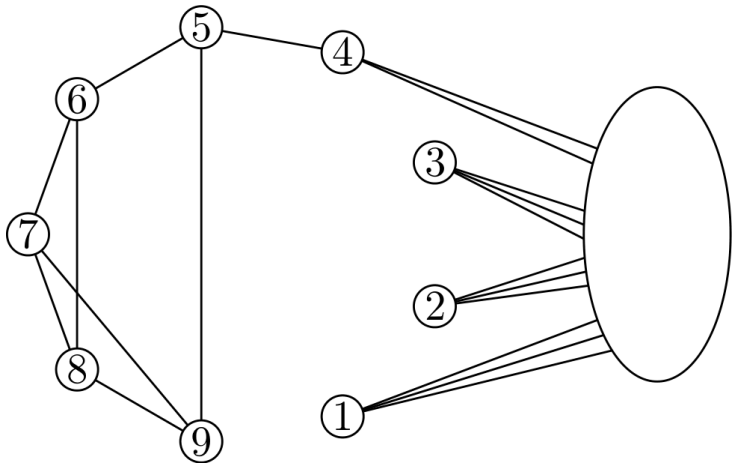


Рис. 4.2.3

Пусть «группа одиннадцати» образует полный подграф. При этом каждая вершина «группы одиннадцати» смежна ровно с одной вершиной «группы девяти». Вершины 1, 2 и 3 соединены только с «группой одиннадцати». Вершина 4 имеет две смежных в «группе одиннадцати». Тогда степени вершин «группы девяти» равны трем, а степени вершин «группы одиннадцати» равны 11.

Ответ: 11.

4.2.3. Математика. 10–11 классы

Задача 4.2.3.1. (15 баллов)

Тема: признаки делимости.

Условие

Семизначное число $*20*25*$ (звездочки обозначают некоторые цифры) делится на 2475. Какое это число?

Решение

Поскольку

$$2475 = 9 \cdot 25 \cdot 11,$$

то искомое число делится на 9, 25 и 11.

Из делимости на 25 следует, что последняя цифра равна нулю. То есть искомое число равно

$$\overline{a20b250}.$$

Из делимости на 9 следует, что сумма всех цифр делится на 9. Следовательно, $a + b \vdots 9$. Так как $a \neq 0$, то возможные варианты для $(a, b) : (1, 8)$, $(2, 7)$, $\ldots$, $(9, 0)$, а также $(9, 9)$.

Из делимости на 11 следует, что разность между суммами цифр на четных и на нечетных позициях делится на 11. То есть

$$(a+0+2+0)-(2+b+5)=a-b-5 \vdots 11.$$

Из перечисленных выше вариантов пар (a,b) подходит лишь один: $a=7$, $b=2$ (проверяется непосредственным перебором).

Ответ: 7202250.

Задача 4.2.3.2. (20 баллов)

Темы: текстовая задача, логика.

Условие

Ваня решил обновить мышь и клавиатуру на своем рабочем месте. То же самое решила сделать и Таня. Ваня готов заплатить за клавиатуру не более 2 000 руб., а за мышь не более r руб. Таня, наоборот, готова заплатить за клавиатуру не более r руб., а за мышь не более 2 000 руб. Ваня и Таня непременно покупают товар, если он стоит не более, чем они готовы за него заплатить, а при большей стоимости — не покупают. Сотрудники магазина «Вся электроника» устанавливают цены на товары, зная о ценовых предпочтениях своих клиентов. Поразмыслив, сотрудники сообразили: если продавать мышь и клавиатуру только в комплекте, то наибольшая возможная выручка от продажи товаров Ване и Тане будет на $p\%$ больше, чем наибольшая возможная выручка при продаже товаров по отдельности (Ваня или Таня купит комплект, если его стоимость не более $2\,000+r$ руб.). При каком наименьшем значении r величина p будет наибольшей?

Решение

При продаже клавиатуры и мыши в комплекте магазину необходимо установить цену $2\,000+r$ руб. за комплект (при меньшей стоимости выручка будет меньше, а при большей — и вовсе равна нулю). Тогда наибольшая выручка при продаже товаров Ване и Тане в комплекте равна

$$A=2\cdot(2\,000+r)=4\,000+2r \text{ руб.}$$

Теперь определим наибольшую возможную выручку при продаже товаров по отдельности. Заметим, что нет смысла устанавливать цены на клавиатуру и мышь, отличные от 2 000 или r. Поясним это утверждение. Пусть $m=\min\{2\,000,r\}$, $M=\max\{2\,000,r\}$, c — цена на товар. Если $c<m$, то товар купят все, а цену можно повысить до m, увеличив выручку. Если $m<c<M$, то товар купит кто-то один, а цену можно поднять до M, увеличив выручку. Если $c>M$, то товар никто не купит, следовательно, можно увеличить выручку, положив, например, $c=m$.

Если $r=2\,000$ руб., то нет разницы между продажей товаров в комплекте и по отдельности, то есть $p=0\%$.

Рассмотрим случай, когда $r<2\,000$.

Если клавиатура и мышь стоят одинаково по 2 000 руб., то Ваня купит только клавиатуру, а Таня — только мышь. Суммарная выручка магазина равна 4 000 руб.

Если клавиатура и мышь стоят одинаково по r руб., то все купят все товары. Суммарная выручка магазина составит $4r$ руб.

Если клавиатура и мышь стоят по-разному, то один из покупателей купит и клавиатуру, и мышь, а второй — только тот товар, который стоит r руб. Суммарная выручка магазина в этом случае равна $2\,000 + 2r$ руб.

Если $r \leqslant 1\,000$, то

$$4r \leqslant 2\,000 + 2r \leqslant 4\,000,$$

следовательно, наибольшая выручка равна $4\,000$. Отношение

$$\frac{A}{4\,000} = 1 + \frac{2r}{4\,000}$$

принимает наибольшее значение $3/2$ при $r = 1\,000$.

Если же $1\,000 < r < 2\,000$, то

$$4\,000 < 2\,000 + 2r < 4r,$$

следовательно, наибольшая выручка равна $4r$. Имеем

$$\frac{A}{4r} = \frac{1\,000}{r} + \frac{1}{2} < \frac{3}{2}.$$

Таким образом, в случае $r < 2\,000$ наибольшее значение $p = 50\%$ достигается при $r = 1\,000$.

Рассмотрим случай, когда $r > 2\,000$.

Если клавиатура и мышь стоят одинаково по 2 000 руб., то все купят все товары. Суммарная выручка магазина составит 8 000 руб.

Если клавиатура и мышь стоят одинаково по r руб., то Ваня купит только мышь, а Таня — только клавиатуру. Суммарная выручка равна $2r$ руб.

Если клавиатура и мышь стоят по-разному, то один из покупателей купит и клавиатуру, и мышь, а второй — только тот товар, который стоит 2 000 руб. Выручка магазина — $4\,000 + r$ руб.

Если $r \leqslant 4\,000$, то

$$2r \leqslant 4\,000 + r \leqslant 8\,000,$$

следовательно, наибольшая выручка равна 8 000. Отношение

$$\frac{A}{8\,000} = \frac{1}{2} + \frac{r}{4\,000}$$

достигает наибольшего значения $\frac{3}{2}$ при $r = 4\,000$.

Если $r > 4\,000$, то

$$8\,000 < 4\,000 + r < 2r.$$

Тогда наибольшая выручка равна $2r$,

$$\frac{A}{2r} = \frac{2\,000}{r} + 1 < \frac{3}{2}.$$

Таким образом, в случае $r > 2\,000$ наибольшее значение $p = 50\%$ достигается при $r = 4\,000$.

Отсюда делаем вывод, что наименьшее значение r, при котором величина p принимает наибольшее значение 50%, равно $1\,000$ руб.

Ответ: $1\,000$.

Задача 4.2.3.3. (20 баллов)

Темы: стереометрия, векторная алгебра.

Условие

Внутри тетраэдра $ABCD$ выбрана точка Z, обладающая свойством:

$$\overrightarrow{AZ} = 2\overrightarrow{ZB} + 3\overrightarrow{ZC} + 4\overrightarrow{ZD}.$$

Прямая DZ пересекает основание ABC в точке E. Чему равно отношение $\dfrac{DZ}{ZE}$?

Решение

Первый способ. Разместим в точках A, B, C и D массы 1, 2, 3 и 4. Поскольку выполняется равенство

$$\overrightarrow{ZA} + 2\overrightarrow{ZB} + 3\overrightarrow{ZC} + 4\overrightarrow{ZD} = \overrightarrow{0},$$

то точка Z — центр масс вершин тетраэдра. Центр масс не изменится, если заменить вершины основания ABC на точку M массы $1+2+3 = 6$ — центр масс точек A, B, C.

Точка Z лежит на отрезке DM, следовательно, точка M лежит на прямой DZ. В то же время, точка M лежит в плоскости ABC. Следовательно, $E = M$.

По свойству центра масс (архимедово правило рычага) имеем $\dfrac{DZ}{ZE} = \dfrac{6}{4} = 1{,}5$.

Второй способ. В данном в условии равенстве

$$\overrightarrow{AZ} = 2\overrightarrow{ZB} + 3\overrightarrow{ZC} + 4\overrightarrow{ZD}$$

каждый вектор $\overrightarrow{XY}$ представим как сумму $\overrightarrow{XE} + \overrightarrow{EY}$. Имеем

$$\overrightarrow{AE} + \overrightarrow{EZ} = 2(\overrightarrow{ZE} + \overrightarrow{EB}) + 3(\overrightarrow{ZE} + \overrightarrow{EC}) + 4(\overrightarrow{ZE} + \overrightarrow{ED}).$$

Раскрывая скобки и приводя подобные слагаемые, получаем

$$\overrightarrow{AE} + 2\overrightarrow{BE} + 3\overrightarrow{CE} = 10\overrightarrow{ZE} + 4\overrightarrow{ED}.$$

Заметим, что в левой части складываются векторы, принадлежащие плоскости ABC, следовательно, и вся сумма будет лежать в плоскости ABC. Сумма в правой

части — вектор, коллинеарный прямой DE. Поскольку прямая DE не лежит в плоскости ABC, то полученное равенство возможно лишь в случае, когда левая и правая части равны нулевому вектору. Тогда

$$\frac{|\overrightarrow{ZE}|}{|\overrightarrow{ED}|} = \frac{4}{10} = \frac{2}{5}.$$

Следовательно, $\dfrac{DZ}{ZE} = \dfrac{3}{2} = 1{,}5$.

Ответ: 1,5.

Задача 4.2.3.4. (20 баллов)

Темы: свойства логарифма, свойства тригонометрических функций, тригонометрические уравнения.

Условие

Найдите наименьший корень уравнения

$$\log_3(\operatorname{tg}^2 x + 2\operatorname{tg} x + 2) + \log_9\left(2\sin^2 x + \cos^2 x - \sqrt{2}\sin x + \frac{1}{2}\right) = 0$$

на отрезке $[-\pi, \pi]$.

Решение

Имеем

$$\operatorname{tg}^2 x + 2\operatorname{tg} x + 2 = (\operatorname{tg} x + 1)^2 + 1 \geqslant 1,$$

следовательно, первое слагаемое в левой части уравнения неотрицательно.

Аргумент второго логарифма равен

$$\sin^2 x + (\sin^2 x + \cos^2 x) - \sqrt{2}\sin x + \frac{1}{2} = 1 + \left(\sin x - \frac{1}{\sqrt{2}}\right)^2 \geqslant 1.$$

Значит, второе слагаемое левой части уравнения также неотрицательно.

Сумма двух неотрицательных слагаемых равна нулю лишь в случае, когда оба этих слагаемых равны нулю. Тогда уравнение равносильно системе

$$\begin{cases} \log_3((\operatorname{tg} x + 1)^2 + 1) = 0, \\ \log_9(1 + \left(\sin x - \frac{1}{\sqrt{2}}\right)^2) = 0 \end{cases} \quad \Longleftrightarrow \quad \begin{cases} \operatorname{tg} x = -1, \\ \sin x = \frac{1}{\sqrt{2}}. \end{cases}$$

На отрезке $[-\pi, \pi]$ имеется единственный корень $x = \dfrac{3\pi}{4}$, удовлетворяющий этой системе.

Ответ: $\dfrac{3\pi}{4}$.

Задача 4.2.3.5. (25 баллов)

Темы: комбинаторика, оценка + пример, сочетания.

Условие

Петя готовит блюда, и у него есть 9 ингредиентов на выбор с разными вкусами. Он руководствуется правилом: «1 блюдо = 3 вкуса». В процессе кулинарных экспериментов у Пети получилось сделать так, что любые два блюда имеют разные наборы вкусов. При этом для любых трех блюд есть вкус, который не входит ни в какое из них. Какое наибольшее число блюд мог приготовить Петя?

Решение

Оценка. Рассмотрим максимальный набор M блюд, который мог приготовить Петя. Дополним этот набор множеством A блюд так, чтобы новый набор $M \cup A$ был максимальным, обладающим свойствами: «1 блюдо = 3 вкуса», и любые два блюда имеют разные наборы вкусов. Всего в наборе $M \cup A$ будет $C_9^3 = 84$ блюда.

Из блюд множества $M \cup A$ можно составить тройки, содержащие в совокупности все 9 вкусов. Назовем такие тройки «плохими». Всего имеется $\dfrac{C_9^3 \cdot C_6^3 \cdot C_3^3}{3!} = 280$ «плохих» троек. Будем убирать из множества $M \cup A$ по одному блюда, входящие в набор A, пока не получим множество M. В конце этой процедуры все «плохие» тройки разрушатся. С одним блюдом можно составить $\dfrac{C_6^3 C_3^3}{2!} = 10$ «плохих» троек, поэтому, убрав одно блюдо, мы разрушим не более 10 «плохих» троек. Следовательно, для количества блюд $|A|$ в множестве A верно неравенство $|A| \cdot 10 \geqslant 280$. То есть $|A| \geqslant 28$. Поскольку $|M \cup A| = 84$, то $|M| \leqslant 84 - 28 = 56$.

Пример. Если Петя приготовил всевозможные различные блюда из трех ингредиентов, не используя девятый, то он получил как раз $C_8^3 = 56$ блюд.

Ответ: 56.

4.3. Инженерный тур

4.3.1. Общая информация

Задача третьего этапа — разработка системы уведомлений операторов колл-центра Департамента безопасности Сбера о подозрительных банковских транзакциях. Итогом работы является готовый программный комплекс, включающий алгоритм поиска мошеннических операций, архитектуру сервиса и рабочую версию веб-интерфейса с интеграцией в Telegram, позволяющие оперативно реагировать на потенциально мошеннические операции.

Участники анализируют предоставленную банком базу данных банковских транзакций, изучают ее структуру и создают алгоритм для поиска ID-мошенника, а также разрабатывают архитектуру системы, предназначенной для визуализации подозрительных транзакций и отправки уведомлений операторам. Команда пишет код системы, создает веб-интерфейс для операторов колл-центра, настраивает интеграцию с Telegram для автоматизированных уведомлений.

Функциональность системы может варьироваться от реактивной модели, когда система реагирует на уже свершившиеся мошеннические операции, до проактивной модели, которая прогнозирует потенциальные риски. Можно внедрить объяснимый ИИ, чтобы операторы могли интерпретировать результаты, а также реализовать дашборды и графовые модели связей транзакций для наглядного анализа данных.

4.3.2. Легенда задачи

Георгий — студент ИТМО, мечта которого — начать свой карьерный путь с ПАО Сбербанк. Как победителя Олимпиады его пригласили на стажировку в департаменте ИТ-блока «Сеть продаж» (ДИТ). ДИТ отвечает за ПО в банкоматах и офисах Сбера, а также добавляет в сервисы функционал ИИ.

Во время стажировки Георгий узнал, что Сбер сейчас проводит ежегодный хакатон «Лучший по профессии», и попросился в команду вместе с более опытными товарищами. Коллеги рассказали, что это отличный способ прокачать свои скиллы, создать полезный в реальности продукт и помочь команде получить очередную, шестую, медаль в зал славы ДИТ «Сеть продаж», обогнав остальные департаменты. А для Георгия лично это отличный шанс показать себя и быть в будущем зачисленным в штат!

Команде и Георгию выпало серьезное задание — борьба с телефонным мошенничеством. Как найти мошенника, его контакты и других его жертв и разработать алгоритм борьбы с социальным инжинирингом, который будет использоваться в реальной жизни?

4.3.3. Требования к команде и компетенциям участников

Количество участников в команде: 3–5 человек.

Компетенции, которыми должны обладать члены команды:

1. **Капитан:** координирует работу команды, собирает и анализирует решения подзадач от всех специалистов, принимает ключевые решения по архитектуре системы и стратегиям выявления мошенничества; обеспечивает согласованность между этапами разработки и отвечает за итоговую интеграцию всех компонентов.
2. **SQL-программист:** отвечает за работу с базой данных, включая написание сложных SQL-запросов, оптимизацию производительности и настройку структуры данных; разрабатывает алгоритмы для выявления мошеннических транзакций и интегрирует их с другими модулями системы. Данная роль может выполняться несколькими членами команды.
3. **Разработчик бэкенда:** создает серверную часть системы, включая API, логику обработки данных и интеграцию с Telegram; отвечает за реализацию триггеров и подключение машинного обучения, обеспечивая стабильную и безопасную работу сервиса.
4. **Разработчик фронтенда:** разрабатывает веб-интерфейс оператора колл-центра, создает удобную визуализацию подозрительных транзакций и обеспечивает интеграцию с бэкендом. Отвечает за удобство работы операторов и интуитивный пользовательский опыт.

4.3.4. Оборудование и программное обеспечение

Таблица 4.3.1

Наименование	**Описание**
Оборудование	Персональные компьютеры/ноутбуки с Windows, macOS или Linux. Доступ в интернет. Возможность локального запуска БД и серверов. (При необходимости) смартфон с Telegram для тестирования уведомлений.
Язык программирования	Python 3.8+, SQL, (при необходимости Java, Javascript, HTML и другие на усмотрение участников финала).
Базы данных (СУБД)	SQLite, PostgreSQL (или другие реляционные СУБД).
Среды разработки	Visual Studio Code, PyCharm, Jupyter Notebook.
Фреймворки (бэкенд)	FastAPI Flask (или другие фреймворки на усмотрение участников финала).
Фреймворки (фронтенд)	React/Vue (или другие фреймворки на усмотрение участников финала).

Наименование	Описание
API-инструменты	Telegram Bot API — для отправки уведомлений операторам, Swagger/Redoc — для документирования API.
Контейнеризация (опционально)	Docker (не обязателен, но может быть использован для контейнеризации проекта).
Система контроля версий	Git/GitHub — для хранения кода и совместной работы.
Инструменты визуализации	Miro, draw.io, Lucidchart (для диаграмм и архитектуры), Paint или аналоги для визуализации архитектуры и взаимодействия компонентов.
Документирование	Microsoft Office/Libre Office/Google Docs и др., Markdown, PDF — для оформления отчетных и презентационных материалов.
Демонстрация проекта	Zoom, запись видео экрана с демонстрацией запуска проекта

4.3.5. Описание задачи

Георгий — студент ИТМО, мечта которого — начать свой карьерный путь с ПАО Сбербанк. Как победителя Олимпиады его пригласили на стажировку в департаменте ИТ блока «Сеть продаж» (ДИТ). ДИТ отвечает за ПО в банкоматах и офисах Сбера, а также добавляет в сервисы функционал ИИ.

Во время стажировки Георгий узнал, что Сбер сейчас проводит ежегодный хакатон «Лучший по профессии» и попросился в команду вместе с более опытными товарищами. Коллеги рассказали, что это отличный способ прокачать свои скиллы, создать полезный в реальности продукт и помочь команде получить очередную, шестую, медаль в зал славы ДИТ «Сеть продаж», обогнав остальные департаменты. А для Георгия лично это отличный шанс показать себя и быть в будущем зачисленным в штат!

Команде и Георгию выпало серьезное задание — борьба с телефонным мошенничеством. Как найти мошенника, его контакты и других его жертв и разработать алгоритм борьбы с социальным инжинирингом, который будет использоваться в реальной жизни?

Требуемые результаты решения задачи: команда должна разработать систему обнаружения подозрительных банковских транзакций и уведомления об этом для операторов колл-центра банка. Итогом работы является готовый программный комплекс, включающий:

- алгоритм поиска мошеннических операций;
- архитектуру сервиса;
- рабочую версию веб-интерфейса с интеграцией в Telegram.

Техническое задание:

- База данных банковских транзакций, содержащая параметры платежей и метаданные клиентов.

- Описание типов мошеннических операций и требований к обнаружению аномалий.
- Требование к отправке уведомлений операторам через Telegram и веб-интерфейс.
- Ожидание возможной интеграции с машинным обучением и триггерами для повышения точности детекции.

Ограничения:

- Запросы к базе данных должны быть оптимизированными для быстрого выполнения.
- Уведомления должны отправляться в реальном времени или с минимальной задержкой.
- Веб-интерфейс должен обеспечивать удобство работы операторов и быть защищен от несанкционированного доступа.
- Внедрение машинного обучения возможно, но должно учитывать интерпретируемость решений для операторов.
- Система должна быть масштабируемой и устойчивой к нагрузкам.

Этап 1. Аналитика

- Проанализировать структуру базы данных банковских транзакций (предоставлена организаторами).
- Разработать алгоритм поиска ID мошенника на основе транзакционных данных.
- Оценить возможные критерии мошеннических действий (например, частота транзакций, географические аномалии, переводы на подозрительные счета).
- Подготовить программный код поиска мошенника и представить результат в виде ID мошенника.

Этап 2. Разработка архитектуры сервиса

- Спроектировать архитектуру системы (взаимодействие базы данных, бэкенда, фронтенда, Telegram-уведомлений).
- Разработать диаграмму вариантов использования для операторов колл-центра.
- Выбрать технологии для реализации (база данных, серверный фреймворк, UI-библиотеки, интеграция с Telegram API).
- Разработать структуру API и определить методы для обработки транзакций.
- Создать макет веб-интерфейса оператора колл-центра.
- Спроектировать механизм отправки уведомлений в Telegram на основе событий в системе.
- Описать потенциальное масштабирование системы.

Этап 3. Реализация проекта

- Разработать бэкенд: REST API, обработка запросов, бизнес-логика.
- Разработать веб-интерфейс оператора, отображающий подозрительные транзакции и их параметры.
- Настроить интеграцию с Telegram для отправки автоматизированных уведомлений.
- Реализовать механизмы триггеров и базовые алгоритмы машинного обучения для выявления аномалий (по желанию команды).

- Доработать интерфейс возможностью фильтрации данных (например, по периоду, сумме, клиенту) (по желанию команды).
- Публикация проекта в репозитории github.
- Оформление репозитория и создание инструкции по запуску проекта.

Каждый этап оценивается отдельно по 100-балльной шкале.

4.3.6. Система оценивания

Оценивание происходит вручную, как средняя оценка экспертов по проекту, на основе критериев. Работа ведется в течение трех дней. Каждый этап оценивается отдельно.

Этап 1. Аналитика (100 баллов) — оценивается после первого дня

- Дан неверный ответ, решения не предоставлено — 0 баллов.
- Описаны не менее трех ключевых параметров мошенничества как результат анализа базы данных; программный код не представлен, ответ верный или частично верный — 15 баллов.
- Разработан и описан алгоритм поиска личности мошенника, программный код представлен и задокументирован, ответ верный или частично верный — 25 баллов.
- Представлен корректный результат (личность мошенника), представлено алгоритм и корректно задокументированный программный код — 50 баллов.

Детализированная шкала оценивания для первого этапа (100 баллов)

1. Описание ключевых параметров мошенничества (максимально 20 баллов):
    - 0 баллов — параметры не описаны или не соответствуют теме.
    - 5 баллов — описан один параметр (например, сумма транзакции).
    - 10 баллов — описаны два параметра, поверхностный анализ.
    - 15 баллов — описаны три параметра, логично обоснованы.
    - 20 баллов — описано более трех параметров, хорошо структурированы, с примерами и объяснением, почему они важны.
2. Разработка и описание алгоритма поиска мошенника (максимально 20 баллов):
    - 0 баллов — алгоритм отсутствует.
    - 10 баллов — представлен общий подход, но без конкретики.
    - 15 баллов — алгоритм описан, но не оптимален или частично формализован.
    - 20 баллов — четкий пошаговый алгоритм, реализуемый в SQL-коде, с обоснованием / логикой исключений / возможностью масштабирования или повторного использования.

3. Представление корректного результата (ID мошенника) (максимально 50 баллов):
    - 0 баллов — ID не найден, ответ отсутствует или неверен.
    - 15 баллов — ID найден правильно, но решение неполное (нет адреса, номера телефона или чего-то другого), нет адекватного вывода на экран.
    - 25 баллов — ID найден правильно, но не подтвержден SQL-запросами / без пояснения логики решения.
    - 50 баллов — ID найден правильно, подтвержден SQL-запросами и сопровождается понятным объяснением логики поиска.
4. Качество кода и документации (максимально 10 баллов):
    - 0 баллов — код нечитаем, нет комментариев.
    - 3 балла — код частично читаем, с минимальными комментариями.
    - 6 баллов — код структурирован, соблюден базовый стиль, есть краткие пояснения.
    - 10 баллов — код оформлен согласно стандартам, документирован, с пояснением логики.

Оценивание происходит вручную, как средняя оценка экспертов по проекту, на основе критериев. Работа ведется в течение трех дней. Каждый этап оценивается отдельно.

Этап 2. Разработка архитектуры сервиса (100 баллов)

- Разработана диаграмма архитектуры системы — 10 баллов.
- Подготовлена диаграмма вариантов использования — 10 баллов.
- Выбраны технологии и обоснован их выбор — 10 баллов.
- Спроектирована структура API и определены методы — 10 баллов.
- Разработан прототип веб-интерфейса — 20 баллов.
- Реализован и обоснован механизм отправки уведомлений в Telegram — 15 баллов.
- Продуманы триггеры и алгоритмы для выявления подозрительных транзакций — 10 баллов.
- Описание архитектуры оформлено в документации — 10 баллов.
- Архитектурные решения позволяют дальнейшее масштабирование системы — 5 баллов.

Детализированная шкала оценивания для второго этапа (100 баллов)

1. Диаграмма архитектуры системы (максимально 10 баллов):
    - 0 баллов — диаграмма отсутствует.
    - 3 балла — диаграмма представлена, но неполная, отсутствуют ключевые компоненты.
    - 6 баллов — указаны основные блоки (БД, фронт, бэк), но взаимодействие между ними не отражено.
    - 8 баллов — указаны основные блоки (БД, фронт, бэк), потоки данные не документированы подробно, а представлены в общих чертах.

- 10 баллов — логичная, визуально чистая диаграмма, отражающая модули, взаимодействие, потоки данных и внешние сервисы (например, Telegram).

2. Диаграмма вариантов использования (максимально 10 баллов):
   - 0 баллов — диаграмма не представлена.
   - 5 баллов — диаграмма сделана, но не учитывает роли пользователя.
   - 8 баллов — охвачены базовые сценарии использования (оператор, админ).
   - 10 баллов — четко структурированная диаграмма с ролями, действиями, связями и комментариями.
3. Выбор технологий и обоснование (максимально 10 баллов):
   - 0 баллов — список технологий отсутствует.
   - 3 балла — указан список технологий без объяснения.
   - 6 баллов — дано обоснование выбора одной-двух ключевых технологий.
   - 8 баллов — приведены аргументы по стеку (БД, фронт, бэк, интеграции).
   - 10 баллов — логично обоснован и согласован полный технологический стек с учетом масштабируемости и совместимости компонентов.
4. Структура API и описание методов (максимально 10 баллов):
   - 0 баллов — API не описан.
   - 5 баллов — описана структура API и основные методы.
   - 10 баллов — полная спецификация API, соответствующая REST-принципам, описаны методы с параметрами, типами запросов и примерами.
5. Прототип веб-интерфейса (максимально 20 баллов):
   - 0 баллов — отсутствует.
   - 5 баллов — представлен общий макет (например, набросок в Paint).
   - 10 баллов — создан кликабельный прототип или интерфейс в Figma/HTML.
   - 15 баллов — прототип логичен, включает элементы фильтрации, уведомлений и отображения данных.
   - 20 баллов — полнофункциональный и хорошо продуманный прототип, приближенный к финальной реализации.
6. Механизм уведомлений в Telegram (максимально 15 баллов):
   - 0 баллов — не описан.
   - 7 баллов — описан в общих чертах (обозначено наличие).
   - 10 баллов — дано описание взаимодействия с системой.
   - 15 баллов — дано пошаговое описание взаимодействия, прописано с учетом команд, описана логика взаимодействия.
7. Триггеры и алгоритмы выявления подозрительных транзакций (максимально 10 баллов):
   - 0 баллов — отсутствуют.
   - 5 баллов — определены простые условия (например, сумма больше X).
   - 8 баллов — разработан набор бизнес-правил/триггеров на SQL или бэкенде.
   - 10 баллов — учтены разные сценарии и вариации мошенничества или

представлена подготовка к ML.

8. Документация по архитектуре (максимально 10 баллов):
   - 0 баллов — отсутствует.
   - 6 баллов — краткое текстовое описание.
   - 10 баллов — четкая, структурированная документация с примерами и визуализациями/оформлено в Markdown/PDF, содержит блоки по основным модулям.
9. Масштабируемость архитектуры (максимально 5 баллов):
   - 0 баллов — не учитывается масштабируемость.
   - 2 балла — описаны общие идеи (например, «можно будет расширить следующие части...»).
   - 5 баллов — предусмотрены конкретные механизмы масштабирования (например, микросервисы, очередь сообщений).

Этап 3. Реализация проекта (100 баллов)

1. Код системы запускается без ошибок, и система реагирует на изменения в базе данных (максимально 25 баллов):
   - Бэкенд реализован с использованием фреймворков — 10 баллов.
   - Веб-интерфейс оператора реализован с использованием фреймворков — 10 баллов.
   - Реализовано API для взаимодействия с веб-интерфейсом — 5 баллов.
   - Интеграция с Telegram успешно работает — 10 баллов.
   - Реализованы триггеры и/или алгоритмы машинного обучения как расширение базовой функциональности — 20 баллов.
   - Реализована возможность фильтрации данных (например, по периоду, сумме, клиенту) и визуализации данных — 10 баллов.
   - Код соответствует стандартам оформления — 5 баллов.
   - Проект содержит инструкцию по запуску — 5 баллов.
2. Код запускается без ошибок, система реагирует на изменения в БД (максимально 25 баллов):
   - 0 баллов — система не запускается, возникают критические ошибки/невозможно оценить реакцию на изменения в базу данных.
   - 15 баллов — работает корректно, не проверялась реакция на изменения.
   - 20 баллов — система стабильно запускается, корректно реагирует на обновления БД (например, по триггерам).
   - 25 баллов — запускается без ошибок, стабильная работа, включен механизм автоматической обработки новых данных и отправки уведомлений в бот.
3. Бэкенд реализован с использованием фреймворков (максимально 10 баллов):
   - 0 баллов — отсутствует или написан на чистом языке без структуры.
   - 5 баллов — используется фреймворк, но архитектура проекта не соблюдена/вызывает вопросы.
   - 10 баллов — использован фреймворк (например, Flask/FastAPI), код структурирован, соблюдены принципы проектирования, реализован REST.

4. Веб-интерфейс оператора реализован с использованием фреймворков (максимально 10 баллов):
    - 0 баллов — веб-интерфейс отсутствует или представлен статичной HTML-страницей.
    - 5 баллов — фреймворк не использован, но страница динамическая.
    - 10 баллов — использован фреймворк (например, React/Vue), интерфейс интерактивный, подключен к API, выполнен на фреймворке, адаптирован под задачи оператора.
5. Реализовано API для взаимодействия с фронтендом (максимально 5 баллов):
    - 0 баллов — API отсутствует.
    - 2 балла — API реализовано частично, нестабильно или не по REST.
    - 5 баллов — API полностью соответствует REST, стабильно работает, протестировано.
6. Интеграция с Telegram (максимально 10 баллов):
    - 0 баллов — интеграция отсутствует.
    - 5 баллов — Telegram-бот реализован, но не получается полностью подтвердить его корректность работу/не реагирует на изменения в базе автоматически.
    - 10 баллов — Telegram-бот реализован, бот отправляет сообщения по событиям из БД, есть логика форматирования и связи с основной системой.
7. Триггеры и/или алгоритмы машинного обучения (максимально 20 баллов):
    - 0 баллов — отсутствуют.
    - 10 баллов — реализованы простые SQL-триггеры или правила.
    - 20 баллов — подключена модель машинного обучения или сложная логика выявления аномалий.
8. Фильтрация данных во фронтенде (максимально 10 баллов):
    - 0 баллов — фильтрация/визуализация отсутствует.
    - 5 баллов — реализована базовая фильтрация по одному параметру, присутствует визуальное отображение одного параметра на странице.
    - 10 баллов — реализована фильтрация по нескольким параметрам (дата, сумма и т. д.), присутствует визуализация всех ключевых параметров.
9. Соответствие кода стандартам оформления (максимально 5 баллов):
    - 0 баллов — код не оформлен, отсутствуют комментарии.
    - 3 балла — частично соблюдены правила, есть структура.
    - 5 баллов — код структурирован, оформлен, читаем, прокомментирован, (PEP8 и др.), репозиторий оформлен корректно.
10. Инструкция по запуску проекта (максимально 5 баллов):
    - 0 баллов — инструкция отсутствует.
    - 3 балла — инструкция частичная или неполная.
    - 5 баллов — полное описание установки и запуска в файле `README.md` (или любом другом читаемом файле).

Балл технической части инженерного тура рассчитывается как среднее арифметическое трех этапов.

Максимальный балл за защиту проекта — 100 баллов.

Итоговая оценка инженерного тура (S) рассчитывается по формуле:

$$S = \frac{P_1 + P_2 + P_3}{3} \times 80\% + P_r \times 20\%,$$

где S — сумма, $P_{1,2,3}$ — оценки этапов инженерного тура, P_r — оценка презентации проекта.

4.3.7. Решение задачи

Этап 1. Аналитика — решение аналитической задачи

Дано: файлы базы данных банковских транзакций, содержащей параметры платежей и метаданные клиентов в формате `.tsv`.

На данном этапе необходимо:

- Проанализировать структуру базы данных банковских транзакций (предоставлена организаторами).
- Разработать алгоритм поиска ID мошенника на основе транзакционных данных.
- Оценить возможные критерии мошеннических действий (например, частота транзакций, географические аномалии, переводы на подозрительные счета).
- Подготовить программный код поиска мошенника и представить результат в виде ID мошенника.

Представленное решение является одним из возможных вариантов реализации и не претендует на статус единственно верного.

Оно иллюстрирует подход к решению задачи, но допускает альтернативные реализации, которые могут быть не менее корректными и эффективными.

Шаг 1. Импорт данных и построение таблиц базы данных

Для решения задачи нужно создать базу данных SQLite и автоматически загрузить в нее данные из нескольких `.tsv`-файлов, преобразовав их в таблицы.

Ниже представлен код на языке Python и SQL для загрузки данных из файлов и создание базы данных SQLite.

Python

```python
import sqlite3
import pandas as pd
import os

# Путь к папке с файлами
data_dir = './samples'  # замените на нужную папку
db_file = 'bank_data.db'

# Список файлов
files = [
        'bank_clients.tsv',
```

```
        'bank_complaints.tsv',
        'bank_transactions.tsv',
        'ecosystem_mapping.tsv',
        'market_place_delivery.tsv',
        'mobile_build.tsv',
        'mobile_clients.tsv'
]

# Создание подключения к базе данных SQLite
conn = sqlite3.connect(db_file)
cursor = conn.cursor()

for file in files:
        table_name = file.replace('.tsv', '')

        file_path = os.path.join(data_dir, file)

        # Загружаем TSV-файл в DataFrame
        df = pd.read_csv(file_path, sep='\t')

        # Сохраняем таблицу в базу данных
        df.to_sql(table_name, conn, if_exists='replace', index=False)

        print(f"Таблица '{table_name}' успешно создана из файла
        ↪  {file}")

# Закрываем соединение
conn.close()
print(f"База данных '{db_file}' успешно создана.")
```

Вывод:

```
Таблица 'bank_clients' успешно создана из файла bank_clients.tsv
Таблица 'bank_complaints' успешно создана из файла bank_complaints.tsv
Таблица 'bank_transactions' успешно создана из файла
↪  bank_transactions.tsv
Таблица 'ecosystem_mapping' успешно создана из файла
↪  ecosystem_mapping.tsv
Таблица 'market_place_delivery' успешно создана из файла
↪  market_place_delivery.tsv
Таблица 'mobile_build' успешно создана из файла mobile_build.tsv
Таблица 'mobile_clients' успешно создана из файла mobile_clients.tsv
База данных 'bank_data.db' успешно создана.
```

Что делает код:

1. Устанавливает путь к папке с `.tsv`-файлами и создает подключение к SQLite-базе данных (`bank_data.db`).
2. Перебирает список файлов, загружает каждый в `pandas.DataFrame` с разделителем табуляция.
3. Преобразует имя файла в имя таблицы (без расширения `.tsv`).
4. Сохраняет каждую таблицу в базу данных (с заменой, если уже существует).
5. По завершении закрывает соединение и выводит сообщения об успешной загрузке.

Результат: готовая база данных SQLite, содержащая семь таблиц, соответствующих исходным `.tsv`-файлам.

Шаг 2. Формирование представлений (`views`) на основе загруженных данных и решение задачи

Предложим вариант решения — создать серию представлений (`views`) в базе данных SQLite на основе ранее загруженных таблиц, объединяя данные из разных источников в единую структуру для анализа связей между клиентами, звонками, транзакциями и доставками.

Python

```
# Подключение к базе данных
conn = sqlite3.connect('bank_data.db')
cursor = conn.cursor()

# SQL-скрипт для создания представлений
sql_script = """
DROP VIEW IF EXISTS client_profile;
CREATE VIEW client_profile AS
SELECT userId, fio, account, phone, mobile_user_id,
 ↪  market_plece_user_id
FROM (
        SELECT userId, fio, account, phone
        FROM bank_complaints INNER JOIN bank_clients USING (userId)
) INNER JOIN ecosystem_mapping ON userId = bank_id;

DROP VIEW IF EXISTS client_transactions;
CREATE VIEW client_transactions AS
SELECT userId, fio, account, account_out AS account_of_receiver,
 ↪  value, event_date
FROM client_profile INNER JOIN bank_transactions ON account =
 ↪  account_out;

DROP VIEW IF EXISTS client_calls;
CREATE VIEW client_calls AS
SELECT userId, fio, from_call AS incoming_call, duration_sec,
 ↪  event_date
FROM client_profile INNER JOIN mobile_build ON phone = to_call;

DROP VIEW IF EXISTS info_by_phone;
CREATE VIEW info_by_phone AS
SELECT phone, fio, address, bank_id, mobile_user_id,
 ↪  market_plece_user_id
FROM (
        SELECT client_id AS mobile_client_id, phone,
         ↪  mobile_clients.fio AS fio, address
        FROM client_calls INNER JOIN mobile_clients ON incoming_call =
         ↪  phone
) INNER JOIN ecosystem_mapping ON mobile_client_id = mobile_user_id;

DROP VIEW IF EXISTS answer;
CREATE VIEW answer AS
SELECT phone, contact_phone, fio, contact_fio, delivery.address,
 ↪  event_date
FROM info_by_phone INNER JOIN market_place_delivery AS delivery ON
 ↪  market_plece_user_id = user_id;
"""
# Выполнение SQL-скрипта
try:
        cursor.executescript(sql_script)
```

```
        conn.commit()
        print("Все представления успешно созданы.")
except Exception as e:
        print("Ошибка при выполнении SQL:", e)

# Выполняем запрос к финальному представлению и выводим результат
try:
        query = "SELECT * FROM answer;"  # можно убрать LIMIT, если
        ↪  нужно всё
        df = pd.read_sql_query(query, conn)
        print("\nРезультат запроса из представления 'answer':\n")
        print(df.to_string(index=False))
except Exception as e:
        print("Ошибка при выборке данных из представления 'answer':",
        ↪  e)
finally:
        conn.close()
```

Вывод:

```
Все представления успешно созданы.
Результат запроса из представления 'answer':
 phone    contact_phone fio       contact_fio            address
  ↪  event_date
79297654321    79297483321  Иванов В.\,Г.  Иванов В.\,Г. г. Москва,
  ↪  ул. Маршала Жукова, д. 7, офис 12  2020-05-07 15:21:45
79297654321    79297483321  Иванов В.\,Г.  Иванов В.\,Г. г. Москва,
  ↪  ул. Маршала Жукова, д. 7, офис 12  2020-05-12 16:01:23
79297654321    79297483321  Иванов В.\,Г. Иванова Д.\,Е.      г.
  ↪  Москва, Проспект мира, д. 12 кв. 175  2020-06-01 22:39:18
```

Что делает код:

1. Подключается к базе данных `bank_data.db`.
2. Выполняет SQL-скрипт, который:
   - создает пять логически связанных представлений (`client_profile`, `client_transactions`, `client_calls`, `info_by_phone`, `answer`);
   - объединяет данные из разных таблиц с помощью `INNER JOIN`.
3. Запрашивает все строки из финального представления answer, которое содержит сведения о клиентах, их контактах и адресах доставки.
4. Выводит результат запроса на экран.
5. Закрывает соединение с базой данных.

Ответ на первый этап: мошенник — клиент банка Иванов В.Г., номер телефона — 79297654321, возможные адреса нахождения — г. Москва, ул. Маршала Жукова, д. 7, офис 12, г. Москва, Проспект мира, д. 12 кв. 175.

Разработанные SQL-правила поиска мошенников на этом этапе лягут в основу построения системы уведомления о подозрительных транзакциях.

Этап 2. Разработка архитектуры сервиса

На данном этапе необходимо:

- Разработать диаграмму вариантов использования для операторов колл-центра.
- Спроектировать архитектуру системы (взаимодействие базы данных, бэкенда, фронтенда, Telegram-уведомлений).
- Выбрать технологии для реализации (база данных, серверный фреймворк, UI-библиотеки, интеграция с Telegram API).
- Разработать структуру API и определить методы для обработки транзакций.
- Создать макет веб-интерфейса оператора колл-центра.
- Спроектировать механизм отправки уведомлений в Telegram на основе событий в системе.

Представленное решение является одним из возможных вариантов реализации и не претендует на статус единственно верного.

Оно иллюстрирует подход к решению задачи, но допускает альтернативные реализации, которые могут быть не менее корректными и эффективными.

Шаг 1. Разработать диаграмму вариантов использования для операторов колл-центра

Диаграмма вариантов использования (Use Case Diagram) — это диаграмма UML, которая показывает, как разные пользователи (акторы) взаимодействуют с системой через основные сценарии использования (функции). Она помогает понять, что делает система с точки зрения пользователя, а не как она устроена внутри.

Контекст: система уведомлений о подозрительных транзакциях.

Акторы (или участники) в UML-диаграммах — это внешние по отношению к системе сущности, которые взаимодействуют с ней. Это могут быть люди, другие системы или устройства, выполняющие определенные действия по отношению к системе. Проще говоря, актор — это тот, кто что-то делает с системой.

Таблица 4.3.2. Акторы системы и их роли

Актор	**Роль в системе**
Клиент	Сообщает о подозрительной транзакции, получает уведомления и может просматривать историю
Оператор	Получает уведомления, просматривает инциденты, обновляет их статус
Аналитик	Оценивает риск, формирует отчеты, архивирует инциденты
Администратор	Управляет пользователями и назначает права доступа
Телеграм-бот	Получает уведомления, сгенерированные системой, и доставляет их пользователям

Варианты использования (use cases) — это действия или сценарии, которые система выполняет в ответ на действия акторов (пользователей или внешних систем). Проще говоря, варианты использования — это то, что система умеет делать.

Таблица 4.3.3. Варианты использования и описание действий

Use Case	**Описание**
UC1: Сообщить о подозрительной транзакции	Клиент инициирует проверку операции
UC2: Получить уведомление	Клиент или оператор получает сообщение о риске
UC3: Просмотреть уведомления	Доступ к журналу уведомлений
UC4: Оценить уровень риска	Аналитик проводит предварительную оценку операции
UC5: Сформировать отчет по инциденту	Подробный отчет по анализу подозрительной транзакции
UC6: Сделать пометку в системе	Добавление служебной записи или комментария к инциденту
UC7: Получить данные клиента	Сбор информации о клиенте (включается в UC1)
UC8: Подготовить сообщение для Telegram	Формирование текста и структуры уведомления
UC9: Отправить уведомление в Telegram	Система отправляет сформированное сообщение в Telegram-бот
UC10: Обновить статус инцидента	Изменение статуса (например, «проверено», «подтверждено» и т. д.)
UC11: Создать нового пользователя	Администратор регистрирует нового пользователя системы
UC12: Назначить права доступа	Назначение ролей и уровней доступа
UC13: Архивировать инцидент	Перевод завершенного инцидента в архив.

Ниже представлена текстовая структура взаимодействий между системой и ее пользователями, подготовленная для переноса в UML-диаграмму вариантов использования. Она включает в себя список акторов, основные сценарии работы системы (варианты использования) и связи между ними, что позволяет наглядно отразить функциональность проекта с точки зрения внешнего поведения.

PlantUML

```
@startuml
left to right direction
actor Клиент
actor Оператор
actor Аналитик
actor Администратор
actor "Телеграм-бот" as TelegramBot
rectangle Система {
 usecase "Сообщить о подозрительной транзакции" as UC1
```

```
  usecase "Получить уведомление" as UC2
  usecase "Просмотреть уведомления" as UC3
  usecase "Оценить уровень риска" as UC4
  usecase "Сформировать отчет по инциденту" as UC5
  usecase "Сделать пометку в системе" as UC6
  usecase "Получить данные клиента" as UC7
  usecase "Подготовить сообщение для Telegram" as UC8
  usecase "Отправить уведомление в Telegram" as UC9
  usecase "Обновить статус инцидента" as UC10
  usecase "Создать нового пользователя" as UC11
  usecase "Назначить права доступа" as UC12
  usecase "Архивировать инцидент" as UC13
  ' include relationships
  UC1 --> UC7 : <<include>>
  UC4 --> UC6 : <<include>>
  UC5 --> UC4 : <<include>>
  UC9 --> UC8 : <<include>>
  UC10 --> UC5 : <<include>>
 }
 Клиент --> UC1
 Клиент --> UC2
 Клиент --> UC3
 Оператор --> UC2
 Оператор --> UC3
 Оператор --> UC10
 Аналитик --> UC4
 Аналитик --> UC5
 Аналитик --> UC13
 Администратор --> UC11
 Администратор --> UC12
 TelegramBot <-- UC9
 @enduml
```

Данный код написан на языке разметки PlantUML — это специальный текстовый синтаксис для автоматической генерации UML-диаграмм.

Что происходит в этом коде:

- `@startuml` и `@enduml` — это рамки, внутри которых находится описание диаграммы.
- `actor` — обозначает актера, т. е. внешнего пользователя системы (в данном случае: клиент, оператор, система).
- `usecase "..." as UC#` — обозначает вариант использования (что пользователь может делать в системе).
- `->` — показывает связь между актером и вариантом использования (какое действие доступно какому пользователю).

Код применяется для создания UML-диаграмм в технической документации в инструментах вроде PlantUML online, draw.io, IntelliJ IDEA, VS Code с плагинами.

Ниже представлена диаграмма вариантов использования, оформленная по стандарту UML по описанному выше коду.

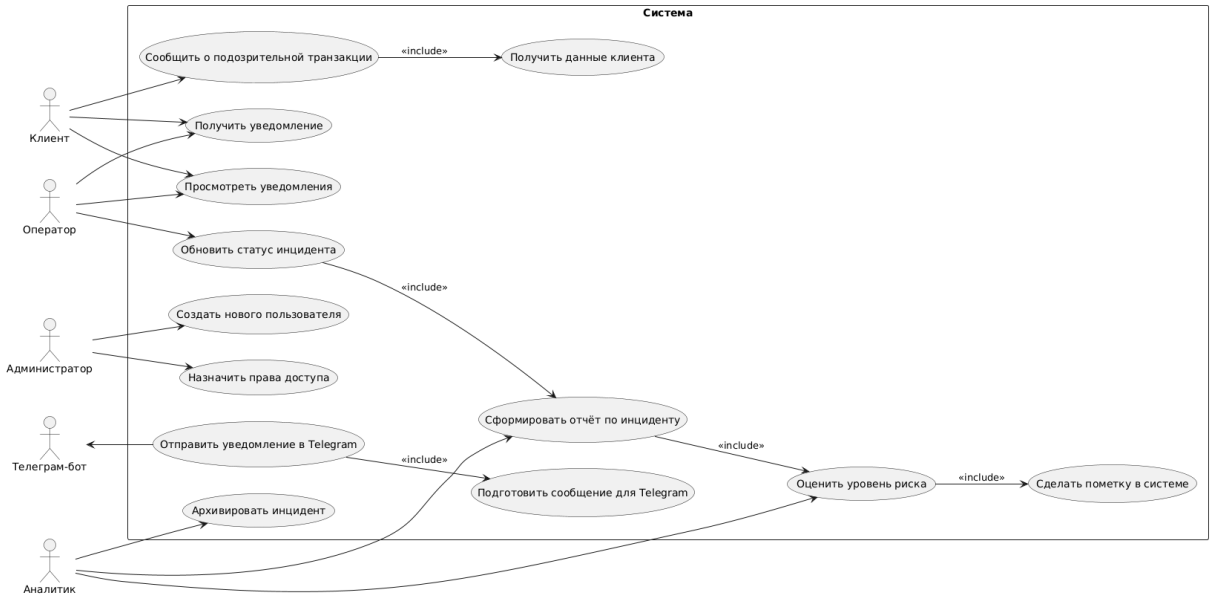


Рис. 4.3.1. Диаграмма вариантов использования системы обработки подозрительных транзакций с интеграцией Telegram-бота

Шаг 2. Спроектировать архитектуру системы (взаимодействие базы данных, бэкенда, фронтенда, Telegram-уведомлений)

На этом шаге нужно создать архитектуру системы, которую можно описать в виде компонентной диаграммы UML (Component Diagram). Она покажет, из чего состоит система, как взаимодействуют модули, и как данные проходят между ними.

Основные элементы системы

Представим основные элементы системы, участвующие в реализации функциональности: пользовательский интерфейс, серверная логика, база данных, аналитический модуль и компонент интеграции с Telegram. Описание этих компонентов помогает понять архитектуру решения и организацию потока данных между модулями.

Таблица 4.3.4. Основные компоненты системы

Компонент	**Описание**
Frontend (UI)	Веб-интерфейс для оператора (просмотр уведомлений, история, фильтрация)
Backend (API-сервер)	Обрабатывает логику: получает данные, вызывает ML/SQL, формирует ответы
База данных (DB)	Хранит транзакции, клиентов, события, статусы инцидентов
Модуль ML/правил	Компонент анализа данных (анализ аномалий, триггеры, правила)
Telegram Notification Module	Отвечает за отправку сообщений в Telegram через Bot API

Потоки данных:

1. Клиент инициирует запрос (через внешнюю систему или оператор).
2. Backend получает данные, передает в ML/SQL-анализ.
3. Анализ формирует инцидент → сохраняется в БД.
4. Backend отправляет результат оператору через фронтенд и Telegram.
5. Оператор работает через UI и может обновить статус → снова идет запрос в backend → БД.

Описание компонентов для UML, представленные в формате кода PlantUML

Ниже приведен текстовый шаблон архитектуры системы в формате PlantUML, который можно использовать для построения компонентной диаграммы. Он отражает ключевые модули системы — от пользовательского интерфейса до базы данных и внешних сервисов — и демонстрирует, как они взаимодействуют друг с другом в рамках решения задачи.

PlantUML

```
@startuml
package "Пользовательский интерфейс" {
  [Frontend UI] as UI
}

package "Серверная логика" {
  [Backend API-сервер] as BE
  [Аналитический модуль (ML/триггеры)] as ML
  [Модуль уведомлений Telegram] as TG
}

database "База данных" as DB

actor "Оператор" as Operator
actor "Клиент" as Client

Client --> UI : запрос (инцидент)
Operator --> UI : просмотр/действия

UI --> BE : HTTP-запрос (REST API)
BE --> DB : SQL-запросы
BE --> ML : анализ данных
ML --> BE : результат (риск)
BE --> TG : сформировать и отправить уведомление
TG --> "Telegram API"
@enduml
```

Архитектура системы и взаимодействие компонентов

Представим диаграмму архитектуры системы. Она показывает, как устроена система внутри: из каких частей она состоит, как они связаны между собой и как проходит поток данных — от пользователя до уведомлений в Telegram. Компонентная диаграмма поможет лучше представить, как взаимодействуют фронтенд, бэкенд, база данных, аналитика и модуль отправки уведомлений.

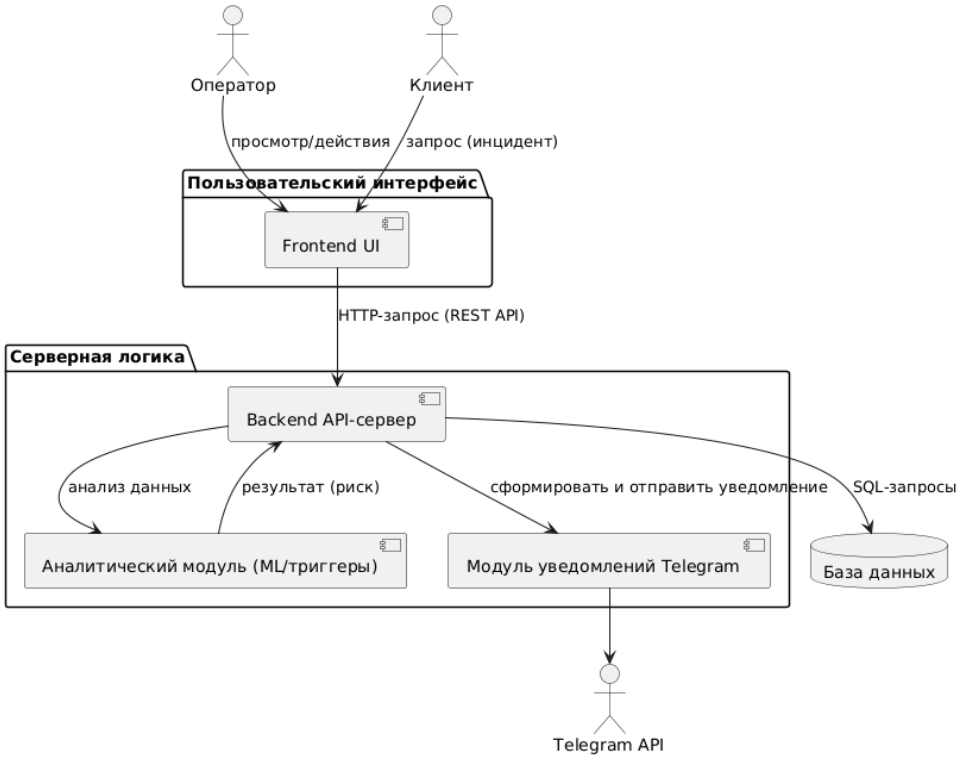


Рис. 4.3.2. Компонентная диаграмма системы обработки подозрительных транзакций с уведомлениями в Telegram

Обращаем внимание, что представленная архитектурная диаграмма — это лишь один из возможных вариантов реализации. Она отражает общее направление решения и может служить ориентиром, но не является единственно верным вариантом. У каждой команды может быть своя архитектура, в зависимости от выбранных технологий, распределения ролей и технических решений. Это соответствует духу заключительного этапа олимпиады, где ценится творческий подход, инженерное мышление и разнообразие идей.

Шаг 3. Выбрать технологии для реализации (база данных, серверный фреймворк, UI-библиотеки, интеграция с Telegram API)

Представляем рекомендованный стек технологий для реализации проекта с кратким обоснованием для каждой категории. Выбор технологий может варьироваться в зависимости от предпочтений и опыта команды. В таблице 4.3.5 представлены лишь рекомендованные инструменты, которые подходят для реализации проекта.

Таблица 4.3.5. Рекомендуемые технологии для реализации системы уведомления о подозрительных транзакциях

Категория	**Технология**	**Обоснование использования**
База данных	PostgreSQL	Надежная, поддерживает сложные SQL-запросы, триггеры и связи между таблицами
	SQLite (альтернатива)	Подходит для прототипирования и локальной разработки

Категория	Технология	Обоснование использования
Бэкенд-фреймворк	FastAPI (Python)	Современный, быстрый, с автоматической документацией Swagger, идеален для REST API
	Flask (альтернатива)	Более простой, но менее структурированный фреймворк
Фронтенд (UI)	React.js	Гибкий, подходит для создания интерфейса оператора, большое сообщество и компоненты
	Vue.js (альтернатива)	Легче в освоении, удобно для небольших проектов
Интеграция Telegram	python-telegram-bot	Простая библиотека для отправки уведомлений, обработки сообщений, удобна для MVP
	aiogram (альтернатива)	Более гибкая и асинхронная библиотека
Контейнеризация	Docker	Упрощает запуск, обеспечивает одинаковую среду для всех участников
Работа с БД в Python	SQLAlchemy	ORM, упрощает взаимодействие с базой данных
ML и обработка данных	pandas, scikit-learn	Подходят для анализа транзакций и построения моделей для выявления аномалий

Шаг 4. Разработать структуру API и определить методы для обработки транзакций

В этом разделе представлена структура REST API, необходимая для взаимодействия между фронтендом, серверной частью, аналитическими модулями и системой Telegram-уведомлений. API обеспечивает доступ к данным о транзакциях, инцидентах и клиентах, а также позволяет инициировать анализ подозрительных операций и управлять уведомлениями.

Основные сущности API

Каждая сущность в API отвечает за определенный аспект системы — от работы с транзакциями до управления статусами инцидентов. Ниже перечислены ключевые сущности и их назначение.

Таблица 4.3.6. Сущности API и их назначение

Сущность	Описание
`/transactions`	Работа с транзакциями клиентов
`/alerts`	Создание и просмотр инцидентов
`/clients`	Получение информации о клиентах
`/analyze`	Запуск аналитики по транзакциям

Сущность	Описание
/notify	Отправка уведомлений в Telegram
/status	Получение и обновление статуса инцидента

Методы и структура запросов

В данном подразделе представлены REST-методы для работы с каждой сущностью API. Все запросы формируются в формате JSON и поддерживают обработку ошибок.

Таблица 4.3.7. REST-методы API и их назначение

Метод	URL	Назначение
GET	/transactions	Получить список транзакций
GET	/transactions/{id}	Получить информацию о конкретной транзакции
POST	/alerts	Зарегистрировать подозрительную операцию
GET	/alerts	Получить список активных инцидентов
POST	/analyze	Запустить анализ транзакции
POST	/telegram/notify	Отправить уведомление в Telegram
PATCH	/alerts/{id}/status	Обновить статус инцидента
GET	/clients/{id}	Получить данные о клиенте

Диаграмма взаимодействия компонентов через API

На диаграмме ниже представлен пример типового взаимодействия между оператором, пользовательским интерфейсом, API, аналитическим модулем и Telegram. Она отражает последовательность запросов и ответов, характерных для анализа подозрительной транзакции и оповещения оператора.

Представим сначала описание диаграммы на PlantUML, а далее — собственно диаграмму.

PlantUML

```
@startuml
actor Operator
participant "Frontend UI" as UI
participant "Backend API" as API
participant "ML-анализатор" as ML
participant "Telegram модуль" as TG
participant "База данных" as DB

Operator -> UI : выбирает транзакцию
UI -> API : GET /transactions/{id}
API -> DB : SQL-запрос
DB --> API : данные транзакции
API --> UI : JSON-ответ

Operator -> UI : нажимает "Анализировать"
UI -> API : POST /analyze
```

```
API -> ML : анализ
ML --> API : результат (риск высокий)
API -> TG : POST /telegram/notify
TG -> Telegram : уведомление
@enduml
```

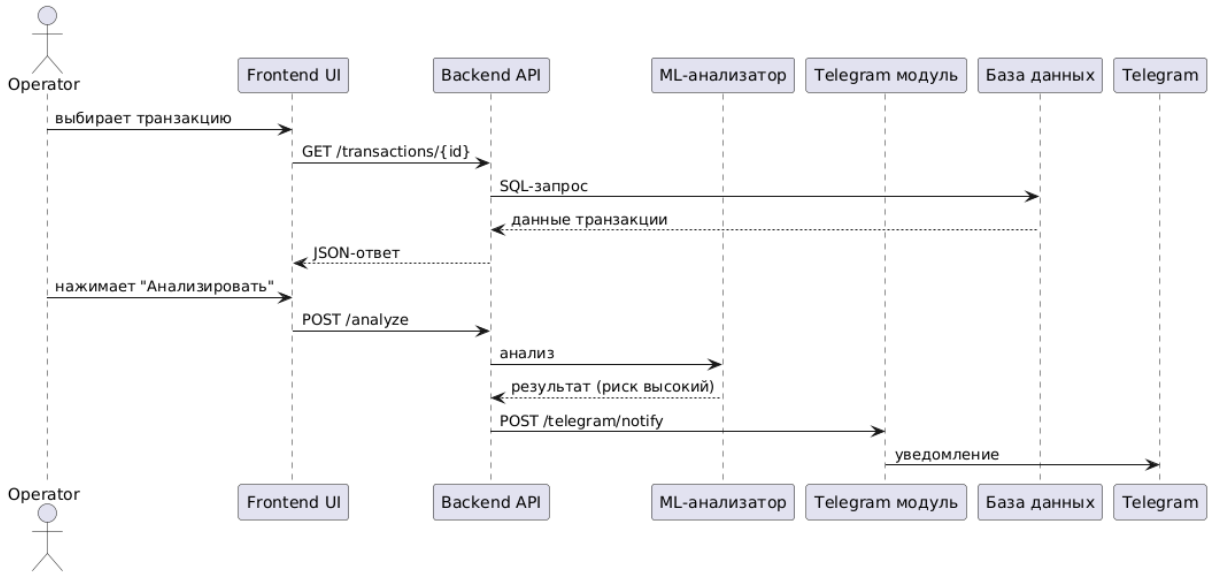


Рис. 4.3.3. Диаграмма последовательности обработки инцидента через API

Представленная структура API и описанные сущности являются примером одного из возможных вариантов реализации. Они иллюстрируют логику взаимодействия между модулями системы, но не являются обязательными или единственно правильными. Команды могут самостоятельно определять, какие сущности использовать, как организовать маршруты и методы API, а также как структурировать обмен данными — в зависимости от выбранных архитектурных решений, распределения ролей и технического стека.

Шаг 5. Создать макет веб-интерфейса оператора колл-центра

Цель этого шага — разработать макет веб-интерфейса, через который оператор сможет получать уведомления, просматривать детали инцидентов, фильтровать транзакции и обновлять их статусы.

Рекомендованный порядок действий:

1. Определить пользовательские сценарии — получение уведомлений, просмотр списка инцидентов, детальный просмотр транзакции, изменение статуса (мошенничество/безопасно), фильтрация и поиск.
2. Составить перечень элементов интерфейса — панель уведомлений / список инцидентов, карточка транзакции, панель фильтров (по дате, сумме, клиенту), например, кнопки: **Подробнее**, **Подтвердить**, **Отклонить**, **Архивировать**.
3. Создать макет (прототип) интерфейса в графическом редакторе (например, Figma) или непосредственно в коде — HTML + CSS + JS (например, с React/Vue).

4. Подготовить структуру компонентов — определить, какие данные приходят из API и как отображаются, настроить начальные состояния (пустой список, ошибка, загрузка).
5. Добавить базовую интерактивность — нажатие на карточку открывает подробности, фильтры обновляют список, кнопки вызывают запросы к API.

Таблица 4.3.8. Компоненты и их функции

Компонент	**Назначение**
`Header`	Верхняя панель с названием системы, иконками, кнопкой выхода
`FilterPanel`	Панель фильтрации транзакций по различным параметрам
`IncidentList`	Главный список инцидентов, полученных из API
`AlertCard`	Отдельная карточка транзакции (краткая информация)
`IncidentDetails`	Подробная информация по инциденту и кнопки управления

На рис. 4.3.4 представлен фрагмент макета веб-интерфейса в приложении Figma, разработанного командой ber, вошедшей в число победителей и призеров олимпиады. Полная версия проекта доступна на официальном сайте олимпиады НТО в разделе финальных работ. Макет иллюстрирует один из возможных вариантов реализации интерфейса оператора колл-центра.

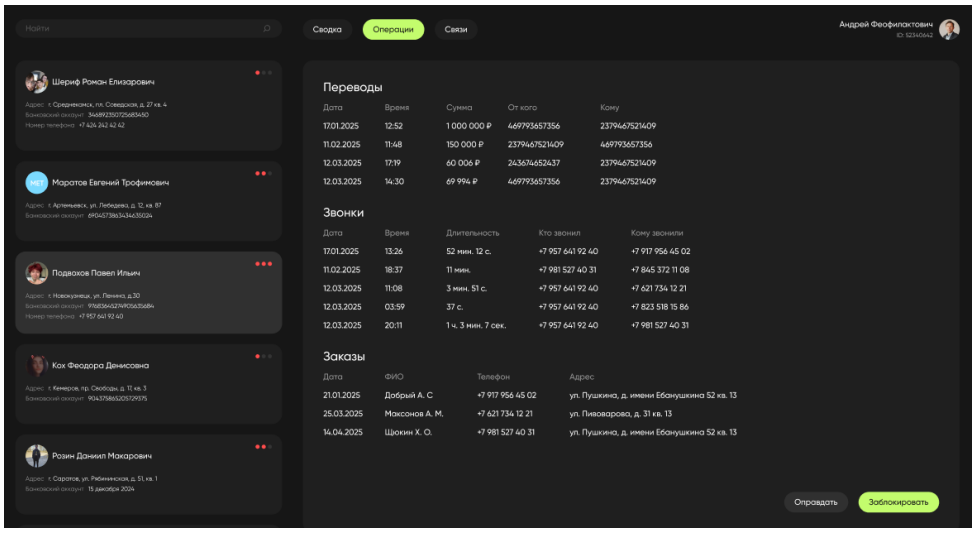


Рис. 4.3.4. Фрагмент макета веб-интерфейса, разработанного командой ber. Страница отображения операций конкретных пользователей. Реализовано в приложении Figma

Порядок шагов по созданию интерфейса и выбор технологий могут отличаться в зависимости от подхода и опыта каждой команды. Макет веб-интерфейса — это творческая часть решения, и каждая команда вправе реализовать его по-своему. В ходе заключительного этапа оценивается не соответствие какому-либо шаблону, а качество реализации, удобство для пользователя, логика отображения информации и общая понятность интерфейса.

Шаг 6. Спроектировать механизм отправки уведомлений в Telegram на основе событий в системе

На этом шаге необходимо проработать механизм отправки уведомлений в Telegram на основе событий в системе. Цель — обеспечить автоматическую отправку уведомлений операторам или администраторам через Telegram-бот, когда в системе фиксируется подозрительная транзакция или происходит важное событие.

Рекомендованный порядок действий:

1. Создать Telegram-бота:

   перейти в Telegram → найти @BotFather;

   использовать команду `/newbot` → задать имя и получить токен.

2. Подключить Telegram API в бэкенд: установить библиотеку

   ```
   pip install python-telegram-bot
   ```

   или использовать `requests` для отправки HTTP-запросов вручную.

3. Определить события для уведомлений:

   пример: выявлена подозрительная транзакция, изменение статуса инцидента, отклонение оператором и т. д.

4. Создать обработчик событий на сервере:

   в коде бэкенда прописать логику: если событие `X` произошло → вызвать функцию `send_telegram_message()`.

5. Сформировать текст уведомления:

   включить ключевые параметры: ID транзакции, клиент, сумма, уровень риска, ссылка на интерфейс.

6. Отправить уведомление оператору:

   передать сообщение через Telegram Bot API → в чат или в группу.

Рекомендуемые технологии для построения механизма отправки уведомлений в Telegram представлены в таблице 4.3.9.

Таблица 4.3.9. Рекомендуемые технологии для построения механизма отправки уведомлений в Telegram

Назначение	**Технология**
Интеграция с Telegram	python-telegram-bot, aiogram, requests
События в системе	Сигналы/триггеры в backend (FastAPI/Flask)
Очереди (опц.)	Celery, Redis (если нагрузка большая)

Пример схемы взаимодействия компонентов механизма отправки уведомлений в Telegram приведен ниже.

Представим сначала описание схемы взаимодействия компонентов на PlantUML, а далее — собственно схему.

PlantUML

```
@startuml
actor Backend
participant "Trigger / Event System"
participant "Telegram Notification Module"
participant "Telegram API"
Backend -> "Trigger / Event System" : фиксирует событие
"Trigger / Event System" -> "Telegram Notification Module" : вызывает
↪ отправку уведомления
"Telegram Notification Module" -> "Telegram API" : HTTP-запрос с
↪ сообщением
Telegram API --> "Telegram Notification Module" : статус 200 OK
@enduml
```

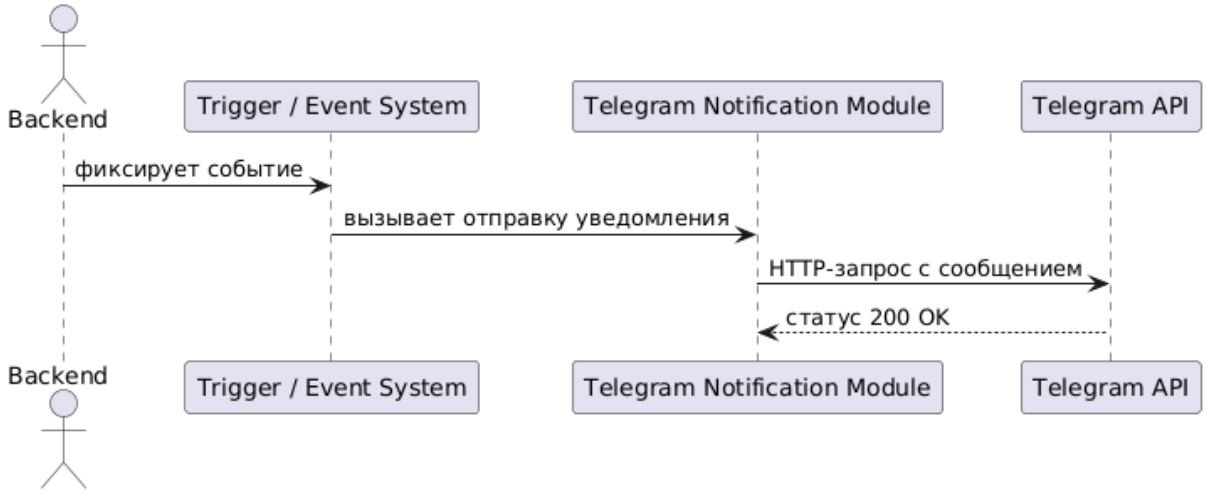


Рис. 4.3.5. Схема взаимодействия компонентов механизма отправки уведомлений в Telegram

Команды могут реализовать механизм уведомлений по-разному — использовать асинхронную отправку, очереди или даже `webhooks` (способ, с помощью которого одна система может автоматически отправлять уведомление другой системе, когда происходит определенное событие). Главное — чтобы уведомление формировалось по событию и доставлялось оперативно и информативно.

Шаг 7. Триггеры и алгоритмы выявления подозрительных транзакций

Для своевременного обнаружения потенциально мошеннических действий система должна обладать встроенными механизмами реагирования на аномальные транзакции. Такие механизмы могут быть реализованы в виде триггеров, наборов бизнес-правил, а также моделей машинного обучения. Выбор подхода зависит от целей проекта, доступных данных и глубины решения. В основу этого шага ложится ваша работа первого этапа.

Пример 1: бизнес-правила и триггеры

Один из распространенных способов — задание четких правил, по которым система автоматически помечает транзакции как подозрительные. Такие правила можно реализовать как на уровне базы данных (SQL), так и в серверной логике (Python, Java и др.).

Примеры правил:

- сумма перевода превышает 500 000 руб;
- клиент совершает более двух переводов за 1 мин;
- переводы на одни и те же счета из разных городов;
- транзакция происходит ночью при отсутствии активности днем.

Реализация может включать:

- SQL-триггеры на события `INSERT` или `UPDATE` (реакция на событие обновления);
- Python-функции, реагирующие на поступление данных;
- отправку уведомления через Telegram при срабатывании правила.

Пример 2: использование машинного обучения

Более продвинутый вариант — построение ML-модели для анализа аномалий в транзакциях. Такой подход позволяет учитывать скрытые зависимости и паттерны поведения, которые сложно формализовать вручную.

Что можно сделать:

- построить классификатор на основе обучающей выборки (например, `Random Forest`);
- использовать алгоритмы аномалий (`Isolation Forest`, `OneClassSVM`);
- обучить модель на временных рядах активности клиента;
- построить граф аффилированных клиентов и находить кластеры.

Для обучения могут использоваться признаки:

- время;
- сумма;
- частота;
- география переводов;
- идентификаторы получателей;
- история обращений или жалоб.

Упрощенная схема взаимодействия бизнес-правил и алгоритмов ML

Представим сначала описание схемы взаимодействия бизнес-правил и алгоритмов ML на PlantUML, а далее — собственно схему.

PlantUML

```
@startuml
actor "Клиент"
participant "Backend"
participant "ML / Бизнес-правила"
database "База данных"
participant "Telegram Модуль"
"Клиент" -> "Backend" : инициирует транзакцию
"Backend" -> "ML / Бизнес-правила" : анализирует данные
"ML / Бизнес-правила" --> "Backend" : возвращает статус
"Backend" --> "База данных" : записывает результат
"Backend" -> "Telegram Модуль" : отправляет уведомление
```

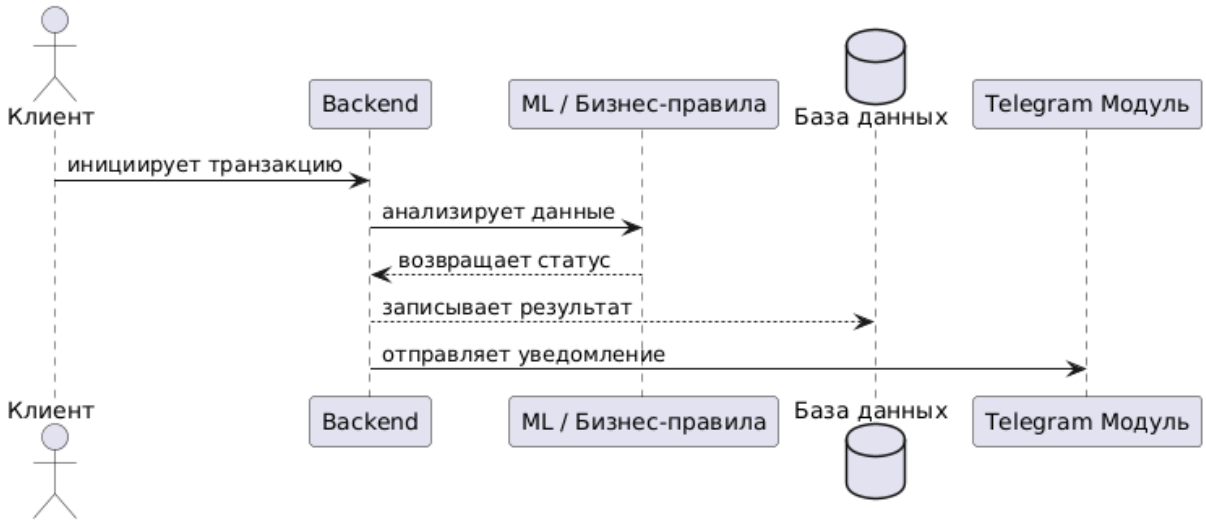


Рис. 4.3.6. Схема взаимодействия бизнес-правил и алгоритмов ML

Каждая команда самостоятельно выбирает подход к реализации механизма выявления подозрительных операций, и это — одна из точек различия в решениях. В рамках оценки учитывается, насколько продуман и обоснован выбранный подход.

На базовом уровне рассматриваются простые условия, например, превышение заданного порога суммы транзакции или превышение количества операций за короткий период. Более продвинутые решения включают разработку набора бизнес-правил или триггеров, например, срабатывание при переводах на один счет с разных устройств, при ночной активности без дневной или при частом повторении операций между одними и теми же клиентами.

Наиболее комплексные подходы предусматривают использование алгоритмов машинного обучения — команды предлагают применение моделей для выявления аномалий, таких как:

- классификация операций;
- построение поведенческого профиля клиента;
- кластеризация;
- графовые модели;
- работа с временными рядами.

В этих случаях также оценивается наличие перечня признаков (`features`), описание выбранного метода (например, `Random Forest`, `Isolation Forest`) и обоснование параметров модели. Критерием является не наличие кода, а детальность проработки подхода — насколько реализация связана с анализом и пониманием сути проблемы.

Шаг 8. Документация по архитектуре

Документация по архитектуре проекта помогает понять, как устроена система, какие модули в ней присутствуют и как они взаимодействуют друг с другом. Она особенно важна, когда систему разрабатывает команда и требуется согласованное понимание ее структуры. Архитектурная документация могла быть представлена участниками в свободной форме — от краткого описания до полноценного технического

отчета с диаграммами, списками компонентов, маршрутами API и технологиями.

Что рекомендовано включать в архитектурную документацию:

- описание основных компонентов (фронтенд, бэкенд, БД, ML-модуль, Telegram-модуль);
- схемы взаимодействия между ними (например, в формате PlantUML);
- примеры запросов (если есть API);
- обоснование выбора технологий;
- визуализация архитектуры в виде диаграммы компонентов или последовательности.

Пример краткого описания архитектуры

Система состоит из frontend-интерфейса, backend-сервера на FastAPI, базы данных PostgreSQL и модуля отправки уведомлений в Telegram. Интерфейс обращается к REST API, который взаимодействует с БД и по заданным условиям отправляет уведомления через Telegram Bot API.

Пример структурированной архитектурной документации

Архитектура проекта

Технологии:

- Backend: FastAPI (Python).
- Frontend: React.
- База данных: PostgreSQL.
- Уведомления: python-telegram-bot.

Компоненты:

- **client-ui**: веб-интерфейс оператора, реализован на React, получает данные через REST API.
- **incident-api**: обрабатывает обращения, проверяет транзакции, передает результат в ML-модуль.
- **ml-analyzer**: модуль анализа транзакций на основе обученной модели.
- **tg-sender**: микросервис отправки сообщений в Telegram.

Схема взаимодействия:

вставлена диаграмма UML.

Дополнительно:

- API описан в Swagger.
- Настройка окружения через docker-compose.
- Все модули документированы в формате Markdown.

Команды имеют свободу в оформлении архитектурной документации — от простого описания в файле `README.md` до полноценного отчета. В рамках оценки учитывается, представлена ли документация вообще, носит ли она краткий и общий характер, или содержит четкую структуру, примеры, визуализации и технические детали. Кроме того, оценивается оформление в читаемом и практичном формате (Markdown, PDF, DOCX).

Шаг 9. Описать потенциальное масштабирование системы

Система уведомлений и анализа транзакций может использоваться как в прототипе, так и в более масштабной промышленной версии. При росте нагрузки, увеличении объема данных или числа пользователей необходимо предусмотреть архитектурные решения, обеспечивающие устойчивость, производительность и гибкость. На этом шаге представлены возможные направления масштабирования системы (таблица 4.3.10).

Таблица 4.3.10. Возможные направления масштабирования системы

Направление	**Описание**
Микросервисная архитектура	Разделение системы на независимые компоненты (API, ML, Telegram и др.) для гибкой поддержки и масштабирования
Очереди сообщений	Использование брокеров (RabbitMQ, Redis, Kafka) для асинхронной обработки событий и разгрузки API
Горизонтальное масштабирование	Развертывание компонентов на нескольких серверах или контейнерах для увеличения производительности
Вертикальное масштабирование	Увеличение вычислительных ресурсов сервера (CPU, RAM) при начальной нагрузке
Кеширование	Использование кеша (например, Redis) для хранения часто запрашиваемых данных и ускорения работы
Репликация и шардирование БД	Повышение надежности и масштабируемости хранения данных за счет распределения и копирования

Представим схему масштабируемой архитектуры (в PlantUML).

PlantUML

```
@startuml
node "API Gateway" {
  [Frontend UI]
}
node "Backend Services" {
  [Аналитика (ML)]
  [Telegram Sender]
  [Инциденты API]
}
database "Основная база данных" as DB
queue "Очередь сообщений (RabbitMQ)" as MQ
[Frontend UI] --> [API Gateway]
[API Gateway] --> [Инциденты API]
[Инциденты API] --> DB
[Инциденты API] --> MQ
MQ --> [Telegram Sender]
MQ --> [Аналитика (ML)]
@enduml
```

Построим диаграмму по этому коду.

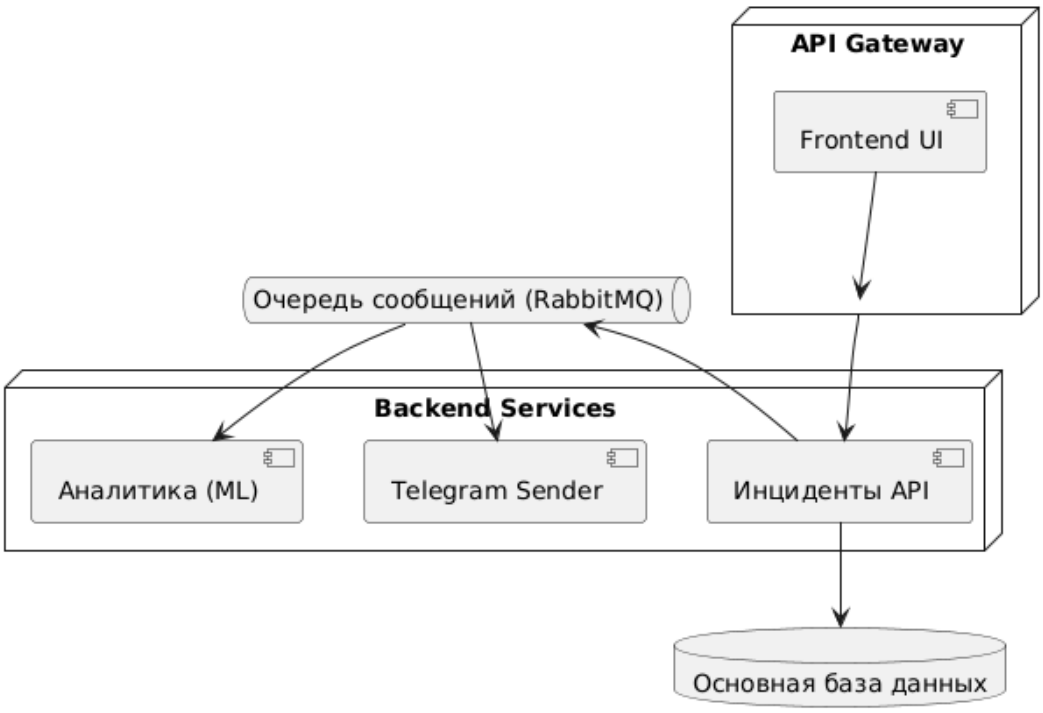


Рис. 4.3.7. Схема масштабируемой архитектуры

Представленные подходы — лишь один из вариантов развития архитектуры. Каждая команда может предложить собственное решение масштабирования: будь то переход на микросервисную архитектуру, внедрение очередей сообщений или оптимизация хранения и обработки данных. В ходе оценки учитывались именно предложенные команды механизмы масштабирования, а не строгость их реализации.

Этап 3. Реализация проекта

На этом этапе участники переходят от проектирования к полноценной реализации системы. Основная задача — воплотить разработанную архитектуру в виде работающего программного решения: написать код, реализовать интерфейсы, настроить взаимодействие между компонентами и убедиться, что система корректно реагирует на события (например, подозрительные транзакции).

Система должна быть протестирована, запускаться без ошибок и демонстрировать реакцию на изменения в базе данных или пользовательские действия. Реализация включает бэкенд, фронтенд, API, Telegram-интеграцию, алгоритмы анализа данных и базовую визуализацию.

В рамках этого методического пособия этап реализации проекта будет описан в виде двух последовательных шагов:

1. реализация компонентов — включает описание создания бэкенда, интерфейса, API, интеграции с Telegram и алгоритмов обработки данных;
2. код и документация — содержит требования к качеству кода и инструкции по запуску проекта.

Далее приводятся фрагменты решений команд, вошедших в число победителей и призеров олимпиады чьи решения получили высокие оценки. Они могут служить ил-

люстрацией ключевых шагов реализации проекта, подходов к решению задач и особенностей архитектуры, использованных участниками.

Шаг 1. Реализация компонентов

Реализация бэкенда с использованием фреймворка

Участники могут выбрать любой современный серверный фреймворк (FastAPI, Flask, Django, Express и т. д.) для построения API, обработки логики анализа и работы с базой данных.

В качестве примера ниже представлен фрагмент кода с использованием фреймворка Django (от участников команды Naumov22, которая входит в число победителей и призеров профиля).

Python

```python
from django.contrib import admin
from .models import FraudAnalysisResult

@admin.register(FraudAnalysisResult)
class FraudAnalysisResultAdmin(admin.ModelAdmin):
        list_display = ('identifier', 'reason', 'id_type', 'fio', 'phone',
        ↪  'address', 'mobile_id', 'marketplace_id')
        list_filter = ('reason', 'id_type', 'identifier')
        search_fields = ('identifier', 'reason', 'id_type')
```

Этот код регистрирует модель `FraudAnalysisResult` в административной панели Django и настраивает, как она будет отображаться в интерфейсе администратора. Импортируются модуль `admin` и модель `FraudAnalysisResult`, которая описывает результаты анализа подозрительных операций. Регистрирует модель в админке и связывает ее с классом настройки `FraudAnalysisResultAdmin` декоратор `@admin.register(...)`. `list_display` определяет столбцы, которые будут показаны в списке записей на странице модели. `list_filter` добавляет фильтры справа в интерфейсе администратора — для удобной фильтрации записей по указанным полям. `search_fields` позволяет осуществлять поиск по этим полям в строке поиска вверху интерфейса администратора.

В качестве примера ниже представлен фрагмент кода с использованием фреймворка Django (от участников команды Naumov22, которая входит в число победителей и призеров профиля).

Python

```python
from django.db import models

class FraudAnalysisResult(models.Model):
        identifier = models.CharField("Идентификатор", max_length=255)
        reason = models.CharField("Причина подозрения",
        ↪  max_length=255)
        id_type = models.CharField("Тип идентификатора",
        ↪  max_length=50, blank=True)
        created_at = models.DateTimeField("Когда проанализировано",
        ↪  auto_now_add=True)
        fio = models.CharField(max_length=255, blank=True)
        phone = models.CharField(max_length=50, blank=True)
```

```
        address = models.TextField(blank=True)
        mobile_id = models.CharField(max_length=100, blank=True)
        marketplace_id = models.CharField(max_length=100, blank=True)

        def __str__(self):
            return f"{self.identifier} — {self.reason}"
```

Данный код определяет модель `FraudAnalysisResult` в рамках фреймворка Django, предназначенную для хранения информации о результатах анализа подозрительных транзакций. Модель содержит поля для идентификатора объекта (`identifier`), причины, по которой он был признан подозрительным (`reason`), типа идентификатора (`id_type`), а также времени создания записи (`created_at`).

Дополнительно предусмотрены поля для хранения персональных данных, таких как Ф. И. О., номер телефона, адрес, а также идентификаторов мобильного пользователя и пользователя маркетплейса. Метод `__str__` определяет строковое представление экземпляра модели, которое используется в интерфейсах отображения и логирования. Эта модель будет отображаться в базе данных как таблица и может использоваться для обработки, хранения и анализа результатов в рамках приложения.

Реализация веб-интерфейса оператора

Интерфейс должен позволять оператору видеть список инцидентов, фильтровать данные, просматривать детали транзакций и управлять их статусом. Использование фреймворков (React, Vue, Bootstrap и др.) приветствуется.

Пример кода интерфейса на фреймфорке Vue (файл `DashboardView.vue`, фрагмент кода команды «Троичный код», которая входит в число победителей и призеров профиля), представлен ниже.

Html

```
<template>
  <main class="col-md-9 ms-sm-auto col-lg-10 px-md-4">
    <div class="d-flex justify-content-between flex-wrap
    ↪  flex-md-nowrap align-items-center pt-3 pb-2 mb-3
    ↪  border-bottom">
      <h1 class="h2">Dashboard</h1>
      <div class="btn-toolbar mb-2 mb-md-0 buttons">
        <button type="button" class="btn btn-sm btn-outline-secondary
        ↪  dropdown-toggle d-flex align-items-center gap-1">
          <svg class="bi" aria-hidden="true"><use
          ↪  xlink:href="#calendar3"></use></svg>
          Эта неделя
        </button>
      </div>
    </div>

    <canvas class="my-4 w-100" id="myChart" width="100%" height=""
    ↪  style="display: block; box-sizing: border-box; height: 440px;
    ↪  width: 1042px;"></canvas>
    <h2>Заголовок секции</h2>
    <div class="table-responsive small">
      <table class="table table-striped table-sm">
        <thead>
          <tr>
            <th scope="col">Количество транзакций</th>
```

```
            <th scope="col">Количество мошеннических транзакций</th>
            <th scope="col">Сумма украденных денег</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td>{{ dashboard_data.txns_count }}</td>
            <td>{{ dashboard_data.fraud_txns_count }}</td>
            <td>{{ dashboard_data.stolen_money }}</td>
          </tr>
        </tbody>
      </table>
    </div>
  </main>

</template>
<style>
  .buttons button{
      height: 100px;
  }

</style>
```

Этот код представляет собой шаблон Vue-компонента, реализующего основное содержимое панели управления (`Dashboard`) веб-приложения, предназначенного для мониторинга подозрительных транзакций. Разметка использует классы `Bootstrap` для стилизации и адаптивного расположения элементов.

В верхней части отображается заголовок `Dashboard` и кнопка с выпадающим меню, обозначающая текущий временной диапазон («Эта неделя»). Ниже размещен график (`<canvas id="myChart»`), предназначенный для визуализации статистики, например, числа транзакций по дням или уровня риска — предполагается, что график будет построен с использованием библиотеки `Chart.js` или аналогичной.

Под графиком размещена таблица с заголовком «Заголовок секции», содержащая три ключевых показателя:

- общее количество транзакций;
- количество выявленных мошеннических транзакций;
- общая сумма украденных средств.

Эти значения динамически подставляются из объекта `dashboard_data`, который, вероятно, поступает из API или хранилища данных (Vuex, Pinia и др.). Таким образом, компонент визуализирует ключевые метрики и предоставляет оператору колл-центра или аналитику общее представление о текущей ситуации с мошенническими операциями в системе.

Интеграция с Telegram

Система должна уметь автоматически отправлять уведомления через Telegram при срабатывании правил или алгоритма. Интеграция может быть реализована через Telegram Bot API.

Пример функции отправки сообщения в Телеграм (фрагмент кода от участников команды Naumov22, которая входит в число победителей и призеров профиля) представлен ниже.

Python

```python
def send_report_via_telegram(report_text):
 url = f"https://api.telegram.org/bot{BOT_TOKEN}/sendMessage"
 for chat_id in CHAT_IDS:
     print(f"!!!!!ЗДЕСЬ Отправка отчёта пользователю {chat_id}...")
     for i in range(0, len(report_text), 4000):  # Telegram ограничение
         chunk = report_text[i:i + 4000]
         requests.post(url, data={"chat_id": chat_id, "text": chunk})
  printf(f"Отправлено пользователю {char_id}")
```

Этот код реализует функцию отправки текстового отчета в Telegram нескольким получателям. Функция `send_report_via_telegram(report_text)` принимает текст отчета и для каждого идентификатора чата из списка `CHAT_IDS` отправляет его через Telegram Bot API.

Поскольку Telegram ограничивает длину одного сообщения 4 096 символами, текст предварительно разбивается на части по 4 000 символов, чтобы избежать ошибок при передаче. Каждая часть отправляется отдельно с помощью POST-запроса на API-адрес, сформированный с использованием токена бота `BOT_TOKEN`. В процессе выполнения выводятся сообщения о начале и завершении отправки для каждого пользователя.

Реализация алгоритмов и триггеров

Расширенная реализация может включать машинное обучение или продвинутые SQL/логические триггеры для анализа транзакций и генерации инцидентов.

Пример кода детектора аномалий с использование алгоритмов машинного обучения (фрагмент кода от участников команды Naumov22, которая входит в число победителей и призеров профиля) представлен ниже.

Python

```python

# detect_suspicious_by_model.py
import pandas as pd
import sqlite3
import joblib

# Загрузка обученной модели
print("ℹ️ Описание признаков:")
print("Признак 1 — номер получателя доставки используется более чем у 3 клиентов за 1 час")
print("Признак 2 — первый перевод на сумму больше 10 000 рублей")
print("Признак 3 — после звонка был перевод в течение 1–10 минут")
print("Признак 4 — перевод ночью (с 22:00 до 06:00)")
print("Признак 5 — получатель получил переводы от более чем 3 разных клиентов за 1 час")
print("")
model = joblib.load("fraud_model.pkl")

# Подключение к базе
import argparse

parser = argparse.ArgumentParser(description="🚨 Проверка транзакций через обученную модель")
```

```
parser.add_argument("--db", type=str, default="anti_fraud_ml.db", help="Путь к SQLite базе данных")
args = parser.parse_args()

DB_PATH = args.db
conn = sqlite3.connect(DB_PATH)
bank_clients = pd.read_sql("SELECT * FROM bank_clients", conn)
bank_transactions = pd.read_sql("SELECT * FROM bank_transactions", conn)
mobile_build = pd.read_sql("SELECT * FROM mobile_build", conn)
ecosystem_mapping = pd.read_sql("SELECT * FROM ecosystem_mapping", conn)
market_place = pd.read_sql("SELECT * FROM market_place_delivery", conn)

bank_transactions['event_date'] = pd.to_datetime(bank_transactions['event_date'])
mobile_build['event_date'] = pd.to_datetime(mobile_build['event_date'])
market_place['event_date'] = pd.to_datetime(market_place['event_date'])

# Подготовка индексов
acc_to_user = bank_clients.set_index('accout')['userId'].to_dict()
user_to_id = ecosystem_mapping.set_index('bank_id')['id'].to_dict()
acc_to_phone = bank_clients.set_index('accout')['phone'].to_dict()

# Создание признаков для всех id
print("🔍 Формируем признаки для модели...")
features = []
for _, eco in ecosystem_mapping.iterrows():
    eid = eco['id']
    bank_id = eco['bank_id']
    if not bank_id:
        continue
    accs = bank_clients[bank_clients['userId'] == bank_id]['accout'].tolist()
    phones = bank_clients[bank_clients['userId'] == bank_id]['phone'].tolist()

    f = {
        'id': eid,
        'is_phone_shared': 0,
        'is_first_large_transfer': 0,
        'is_call_before_transfer': 0,
        'is_night_transfer': 0,
        'is_many_senders': 0
    }

    # Признак 1: номер доставки встречается у >3 пользователей в 1 час
    mpd_rows = market_place[market_place['user_id'] == eco['market_plece_user_id']]
    if not mpd_rows.empty:
        for _, row in mpd_rows.iterrows():
            block = market_place[(market_place['contact_phone'] == row['contact_phone']) &
                (market_place['event_date'].dt.floor('h') == row['event_date'].floor('h'))]
            if block['user_id'].nunique() > 3:
                f['is_phone_shared'] = 1
                break

```

```
    # Признак 2: первый перевод > 10_000
    user_acc = accs[0] if accs else None
    if user_acc:
        txs = bank_transactions[bank_transactions['account_in'] == user_acc]
        if not txs.empty:
            first = txs.sort_values('event_date').iloc[0]
            if first['value'] > 10_000:
                f['is_first_large_transfer'] = 1

    # Признак 3: звонок → перевод через 1–10 минут
    if phones:
        for phone in phones:
            calls = mobile_build[mobile_build['to_call'] == phone]
            for _, call in calls.iterrows():
                after = call['event_date'] + pd.Timedelta(minutes=1)
                before = call['event_date'] + pd.Timedelta(minutes=10)
                txs = bank_transactions[
                    (bank_transactions['account_out'].isin(accs)) &
                    (bank_transactions['event_date'] >= after) &
                    (bank_transactions['event_date'] <= before)
                ]
                if not txs.empty:
                    f['is_call_before_transfer'] = 1
                    break

    # Признак 4: перевод ночью
    night = bank_transactions[(bank_transactions['account_in'].isin(accs)) &
                              ((bank_transactions['event_date'].dt.hour < 6) |
                               (bank_transactions['event_date'].dt.hour >= 22))]
    if not night.empty:
        f['is_night_transfer'] = 1

    # Признак 5: >3 разных отправителя за 1 час
    txs = bank_transactions[bank_transactions['account_in'].isin(accs)].copy()
    txs['hour'] = txs['event_date'].dt.floor('h')
    grouped = txs.groupby(['hour'])['account_out'].nunique().reset_index()
    if any(grouped['account_out'] > 3):
        f['is_many_senders'] = 1

    features.append(f)

# Предсказание
print("🤖 Предсказание модели...")
df_features = pd.DataFrame(features)
X = df_features.drop(columns=['id'])
df_features['predicted_fraud'] = model.predict(X)

# Вывод
print("\n📋 Результаты модели:")
suspects = df_features[df_features['predicted_fraud'] == 1].copy()
suspects['fraud_score'] = model.predict_proba(X[df_features['predicted_fraud'] == 1])[:, 1]
suspects = suspects.sort_values(by='fraud_score', ascending=False)
```

```
for _, row in suspects.iterrows():
    reasons = []
    if row['is_phone_shared']: reasons.append("Признак 1")
    if row['is_first_large_transfer']: reasons.append("Признак 2")
    if row['is_call_before_transfer']: reasons.append("Признак 3")
    if row['is_night_transfer']: reasons.append("Признак 4")
    if row['is_many_senders']: reasons.append("Признак 5")
    print(f"ID {row['id']}: модель предсказала фрод (вероятность:
    ↪  {row['fraud_score']:.2f}) → {', '.join(reasons)}")
```

Этот скрипт `detect_suspicious_by_model.py` реализует автоматическую проверку пользователей на предмет возможного мошенничества с использованием ранее обученной модели машинного обучения.

Он загружает модель из файла `fraud_model.pkl` и применяет ее к данным, извлеченным из базы SQLite, путь к которой можно передать через аргумент `-db`.

Скрипт подключается к базе данных и извлекает таблицы с транзакциями, клиентами, звонками, сопоставлением ID и доставками. Затем для каждого пользователя формируются признаки (`features`), отражающие подозрительные шаблоны поведения:

1. Один и тот же номер телефона доставки используется более, чем у трех клиентов в течение одного часа.
2. Первый перевод пользователя превышает 10 000 руб.
3. После звонка клиент совершает перевод в течение 1–10 мин.
4. Перевод был совершен ночью (с 22:00 до 6:00).
5. Один получатель получает переводы от более чем трех разных отправителей в течение одного часа.

После генерации признаков скрипт передает их в обученную модель, которая делает предсказание о том, является ли поведение пользователя подозрительным. Для пользователей, помеченных как потенциальные мошенники (предсказание = 1), также вычисляется оценка уверенности модели (`fraud score`).

В конце скрипт выводит в консоль список ID подозрительных пользователей, вероятности мошенничества и список сработавших признаков, благодаря которым модель приняла такое решение. Таким образом, скрипт автоматизирует процесс анализа поведения клиентов и может использоваться для поддержки операторов и аналитиков в задачах поиска и прогнозирования мошенничества.

Шаг 2. Код и документация

Качественный код и понятная документация — неотъемлемая часть любого инженерного проекта. В условиях командной олимпиады это особенно важно: эксперты оценивают не только работоспособность системы, но и насколько легко ее прочитать, воспроизвести и доработать. Хорошо оформленный код и наличие инструкций по запуску проекта помогают быстро протестировать решение в едином окружении и убедиться в его корректности.

Таблица 4.3.11. Основные требования к коду проекта

Категория	**Требования**
Читаемость	Используйте понятные имена переменных и функций. Разделяйте код на логические блоки и модули.
Структура проекта	Разделяйте проект на модули (frontend/backend), упорядочивайте файловую структуру проекта.
Стиль кода	Избегайте дублирования кода (принцип DRY). Придерживайтесь единого стиля: отступы, длина строк, соглашения по наименованию. Рекомендуемые гайды: PEP8 (Python): https://peps.python.org/pep-0008/. JavaScript Style Guide (Airbnb): https://github.com/airbnb/javascript. Google Java Style Guide: https://google.github.io/styleguide/javaguide.html.
Комментирование	Добавляйте комментарии к ключевым фрагментам кода, особенно если логика неочевидна. Используйте docstring в функциях и классах (Python). Подпишите нестандартные настройки, параметры, переменные.

Проект должен содержать файл `README.md` или аналог, в котором описано:

- что именно делает проект;
- указано, как его запустить;
- перечислены зависимости и способы их установки;
- приведен пример использования или тестирования;
- по возможности — указаны команды для запуска через Docker (если используется).

Пример содержимого файла `README.md` (из документации проекта команды ber, которая входит в число победителей и призеров профиля) представлен ниже.

md

````
### Автоматическое создание таблиц
Когда в файле `.env` установлено значение `NODE_ENV=development`, при запуске бэкенда таблицы в базе данных создаются автоматически с использованием ORM. Если значение `NODE_ENV` будет отличаться от `development` (например, `production`), автоматическое создание таблиц не выполняется, и их нужно будет создать вручную.
### Создание администратора
Для создания администратора в системе выполните следующую команду:
```bash
pnpm db:create-admin
```
Или с передачей логина и пароля в качестве аргументов:
```bash
pnpm db:create-admin admin securepassword
```
Если логин и пароль не указаны в аргументах, скрипт запросит их во время выполнения.
### Заполнение базы данных тестовыми данными
````

```
14 Мы дополнили стандартный датасет небольшим количеством собственных
    ↪ данных для более полной демонстрации функционала. Если вы хотите
    ↪ протестировать сервис только с исходными данными, переименуйте
    ↪ директорию `apps/migrations/samples-old` в
    ↪ `apps/migrations/samples`.
15 Для заполнения базы данных тестовыми данными выполните:
16 ```bash
17 pnpm db:insert-defaults
18 ```
19 **Важно**: Перед запуском скриптов создания администратора и
    ↪ заполнения базы убедитесь, что:
20 1. Бекенд был запущен хотя бы один раз в режиме разработки для
    ↪ создания необходимых таблиц в базе данных
21 2. База данных настроена и доступна по URL, указанному в .env файле
```

Фрагмент документации команды ber оформлен грамотно и соответствует требованиям к технической документации. Информация логично структурирована по смысловым блокам, команды снабжены пояснениями, язык изложения простой и понятный. Пошаговое описание процесса создания администратора и заполнения базы данных позволяет быстро воспроизвести действия даже тем, кто не знаком с проектом. Хорошо, что сделан акцент на важных условиях запуска, например, необходимости запуска бэкенда в режиме разработки до выполнения скриптов — это предотвращает типичные ошибки.

В качестве небольшого улучшения можно было бы добавить шаблон `.env` и пояснение по структуре переменных окружения, чтобы пользователь сразу понимал, где и как указать путь к базе. Кроме того, было бы полезно добавить краткое описание, содержащее информацию о том, как проверить успешность действий (например, через вход в интерфейс администратора). Тем не менее данный фрагмент является одним из лучших примеров качественной документации среди команд и существенно упрощает процесс тестирования и запуска проекта.

Например, проект команды «Троичный код» имеет хорошо организованную и логичную файловую структуру. Есть разделение по каталогам, и оно соответствует основным компонентам системы: отдельно выделены папки для фронтенда и бэкенда с указанием используемых фреймворков, присутствует отдельный модуль для Telegram-бота, а также блок, связанный с реализацией машинного обучения. Благодаря такому подходу легко ориентироваться в проекте, понимать назначение каждого элемента и находить нужные файлы. Команда включила в проект файл `README.md` с инструкциями запуска. Эта структура является одной из лучших среди представленных решений финала, это отличный пример оформления проекта.

Пример файловой структуры проекта команды «Троичный код», которая входит в число победителей и призеров профиля, приведен на рисунке 4.3.8.

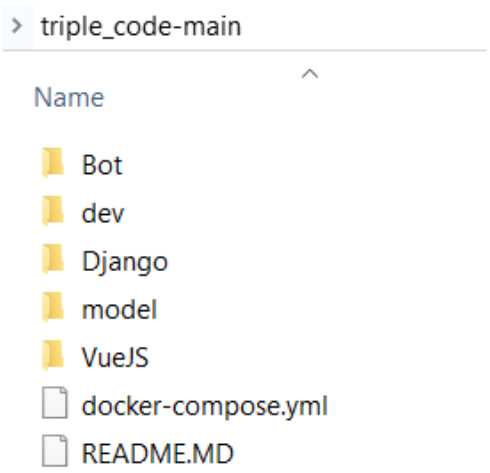


Рис. 4.3.8. Файловая структура проекта команды «Троичный код»

Заключение

Каждая команда реализует проект в соответствии со своей архитектурой, возможностями и выбором технологий. Оценка проводится не по единообразию решений, а по фактической работоспособности системы, качеству реализации, наличию связей между модулями, поддержке уведомлений, фильтрации данных и оформлению проекта.

Кроме того, каждая команда представляет видеозапись демонстрации проекта — это дает возможность всем участникам подробно рассказать о том, что было ими сделано, показать запуск системы, объяснить особенности реализации и подтвердить авторство решения.

Участники успешно демонстрируют работу полноценной системы по отправке уведомлений и анализу подозрительных транзакций, решив задачу, поставленную партнером профиля — ПАО Сбербанк (https://sberstudent.ru/). Лучшие команды отмечаются наградами от организаторов и партнера профиля.

4.3.8. Материалы для подготовки

1. Python и основы разработки на Python. Официальная документация. URL: https://docs.python.org/3/.
2. Stepik: Программирование на Python. URL: https://stepik.org/course/67.
3. Основы работы с базами данных и SQL. Бесплатный курс, Яндекс Практикум. URL: https://start.practicum.yandex/sql-database-basics.
4. PostgreSQL. Официальная документация. URL:https://www.postgresql.org/docs/.
5. Mode SQL Tutorial — практикум на английском языке. URL: https://mode.com/sql-tutorial/.
6. FastAPI. Официальная документация. URL: https://fastapi.tiangolo.com/.
7. Stepik: Веб-разработка на Flask. URL: https://stepik.org/course/512.

8. Telegram Bot API. Официальная документация. URL: https://core.telegram.org/bots/api.
9. Создание Telegram-бота на Python / Habrahabr. URL: https://habr.com/ru/post/262247/.
10. REST API — простое объяснение / Habrahabr. URL: https://habr.com/ru/articles/590679/.
11. Swagger / OpenAPI Docs. URL: https://swagger.io/docs/.
12. The Twelve-Factor App. URL: https://12factor.net/ru/.
13. Diagrams.net (draw.io) — инструмент для создания схем. URL: https://app.diagrams.net/.
14. UML Use Case Diagrams — Lucidchart Guide. URL: https://www.lucidchart.com/pages/uml-use-case-diagram.
15. Git Handbook / GitHub Guides. URL: https://guides.github.com/introduction/git-handbook/.
16. Stepik: Git + GitHub. URL: https://stepik.org/course/3145.
17. Введение в Data Science и машинное обучение. Stepik. URL: https://stepik.org/course/62243/promo.
18. Scikit-learn. Руководство пользователя. URL: https://scikit-learn.org/stable/user_guide.html.

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 50,0 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 1011,0 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 4 победителя и 8 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	73,76 и выше
Призеры	От 66,20 до 71,64

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».