



**НТО**

**МАТЕРИАЛЫ ЗАДАНИЙ**  
**Всероссийской междисциплинарной олимпиады**  
**школьников 8–11 класса**  
**«Национальная технологическая олимпиада»**  
**по профилю**  
**«Большие данные и машинное обучение»**

2024/25 учебный год

*[ntcontest.ru](http://ntcontest.ru)*

УДК 373.5.016:004.65  
ББК 74.263.02  
Б79

Авторы:

М. В. Бабушкин, Л. В. Борель, Н. В. Ведерников, И. А. Воронцов, А. А. Гаврилюк, Е. Н. Горечин, А. Р. Ершов, А. С. Забашта, О. В. Зубков, М. В. Лукина, М. В. Милованович, В. Ю. Тertyчный

**Б79** Всероссийская междисциплинарная олимпиада школьников 8–11 класса «Национальная технологическая олимпиада». Учебно-методическое пособие  
Том 6 **Большие данные и машинное обучение**  
— М.: Ассоциация участников технологических кружков, 2025. — 146 с.

ISBN 978-5-908021-05-0

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 978-5-908021-05-0



9 785908 021050 >

УДК 373.5.016:004.65  
ББК 74.263.02

# Оглавление

<b>1 Введение</b>	<b>5</b>
1.1 Национальная технологическая олимпиада	5
1.2 Большие данные и машинное обучение	13
<b>2 Первый отборочный этап</b>	<b>16</b>
2.1 Работа наставника НТО на этапе	16
2.2 Предметный тур. Информатика	17
2.2.1 Первая волна. Задачи 8–11 класса . . . . .	17
2.2.2 Вторая волна. Задачи 8–11 класса . . . . .	27
2.2.3 Третья волна. Задачи 8–11 класса . . . . .	37
2.2.4 Четвертая волна. Задачи 8–11 класса . . . . .	50
2.3 Предметный тур. Математика	65
2.3.1 Первая волна. Задачи 8–9 класса . . . . .	65
2.3.2 Первая волна. Задачи 10–11 класса . . . . .	68
2.3.3 Вторая волна. Задачи 8–9 класса . . . . .	72
2.3.4 Вторая волна. Задачи 10–11 класса . . . . .	75
2.3.5 Третья волна. Задачи 8–9 класса . . . . .	80
2.3.6 Третья волна. Задачи 10–11 класса . . . . .	85
2.3.7 Четвертая волна. Задачи 8–9 класса . . . . .	89
2.3.8 Четвертая волна. Задачи 10–11 класса . . . . .	93
2.4 Инженерный тур	98
<b>3 Второй отборочный этап</b>	<b>100</b>
3.1 Работа наставника НТО на этапе	100
3.2 Инженерный тур	102
<b>4 Заключительный этап</b>	<b>114</b>
4.1 Работа наставника НТО при подготовке к этапу	114

---

<b>4.2 Предметный тур</b>	<b>116</b>
4.2.1 Информатика. 8–11 классы . . . . .	116
4.2.2 Математика. 8–9 классы . . . . .	128
4.2.3 Математика. 10–11 классы . . . . .	133
<b>4.3 Инженерный тур</b>	<b>139</b>
4.3.1 Общая информация . . . . .	139
4.3.2 Легенда задачи . . . . .	139
4.3.3 Требования к команде и компетенциям участников . . . . .	139
4.3.4 Оборудование и программное обеспечение . . . . .	139
4.3.5 Описание задачи . . . . .	140
4.3.6 Система оценивания . . . . .	141
4.3.7 Решение задачи . . . . .	141
4.3.8 Материалы для подготовки . . . . .	143
<b>5 Критерии определения победителей и призеров</b>	<b>144</b>
<b>6 Работа наставника после НТО</b>	<b>146</b>

# 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. Большие данные и машинное обучение

Профиль Большие данные и машинное обучение посвящен решению задач по анализу данных. В связи с социально-экономическими и технологическими возможностями современной жизни в мире ежедневно собираются огромные массивы данных о деятельности человека, которые нужно обрабатывать, анализировать и структурировать. Такая работа помогает в прогнозировании, моделировании и выявлении закономерностей деятельности человека. Большие данные являются полезным инструментом, благодаря которому можно улучшать качество жизни людей. Алгоритмически обрабатывать их возможно благодаря машинному обучению.

Для подготовки участникам Олимпиады предоставлены материалы:

- по программированию на Python;
- по использованию основных библиотек для анализа данных, основам машинного обучения, теории вероятностей;
- практикумы;
- сборники задач с предыдущих соревнований.

Материалы выложены на странице профиля и на сайте проекта «Академия искусственного интеллекта для школьников».

Знакомство с профилем начинается с «Урока НТО» по профилю Большие данные и машинное обучение, который проводится в общеобразовательных учреждениях. Материалы для проведения урока находятся на сайте <https://nto-lesson.ru/> и доступны после регистрации на платформе «Талант».

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

**В рамках первого отборочного этапа** необходимо решить задачи по математике и информатике на предметном туре, освоить теорию машинного обучения через образовательный блок и развить свои компетенции в анализе данных на инженерном туре.

Задачи **второго этапа** готовят участников к заключительному этапу и представляют собой задания по олимпиадному программированию и машинному обучению, знакомят с платформой по проведению конкурса.

**На заключительном этапе** проходят соревнования в построении прогнозной модели. Здесь ставится цель — спрогнозировать, понравится ли предложенная запись пользователю социальной сети «Одноклассники» по имеющимся данным о пользователях, которые ежедневно просматривают ленту с записями и каким-либо образом взаимодействуют с записями. Для построения системы рекомендаций требуется предсказать тип взаимодействия.

Участникам предоставляется доступ к виртуальному серверу для вычислений (установлен Python 3.9 с Jupyter Notebook, SciPy, NumPy, Scikit Learn и Pandas), кроме того, они сами могут доустановить необходимые им библиотеки и языки.

Доступ к серверу осуществляется с личного или выданного ноутбука через веб-интерфейс Jupyter Notebook.

Наряду с этим, в распоряжении участников находится набор данных, подготовленный партнером профиля — компанией VK. Он состоит из нескольких CSV-файлов:

- `topics.csv`: — информация о записях социальной сети «Одноклассники». Каждая запись характеризуется признаками текста и изображения.
- `users.csv` — данные о пользователях социальной сети «Одноклассники». Каждый пользователь характеризуется датой рождения, полом и идентификатором города.
- `train.csv` — сведения о взаимодействии пользователей с записями социальной сети «Одноклассники». Тип взаимодействия равен L, если запись понравилась, а D — если запись не понравилась.
- `test.csv` — идентификаторы записей и пользователей, для которых требуется предсказать тип взаимодействия.

Решения загружаются на платформе Codeforces.

Участники загружают решение в тестирующую систему в виде текстового файла с ответами для объектов из проверочного набора данных (всего не более 50 решений на одну команду).

Для подготовки к заключительному этапу сформирована подборка материалов по машинному обучению и построению алгоритмов рекомендательных систем, на которые нацелена финальная задача.

Помимо этого, для эффективной подготовки к Олимпиаде проводятся вебинары по тематике профиля и разбор заданий второго этапа, которые доступны на канале университета.

Компетенции, приобретаемые благодаря участию в данном профиле Олимпиады, помогают участникам развить как *hard skills*:

- навыки программирования;
- умение строить алгоритмы и практические знания методов статистического анализа больших данных;
- знание как минимум языков программирования Python или R, основных библиотек, алгоритмов, их ограничений;
- базовые компетенции в теории вероятностей, статистике, математическом анализе и линейной алгебре,

так и *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 школьникам на пути

к становлению гармоничной и развитой личности.

Победители и призеры профиля Большие данные и машинное обучение поступают в ведущие вузы России на специальности, связанные с информационными технологиями и принимают участие в научно-технологических проектных школах.

Участие школьников в данном профиле заметно повышает 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  $0 \leq t \leq 10^6$ .

#### ***Формат выходных данных***

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

**Примеры***Пример №1*

<b>Стандартный ввод</b>
6
<b>Стандартный вывод</b>
120

*Пример №2*

<b>Стандартный ввод</b>
2
<b>Стандартный вывод</b>
0

*Пример №3*

<b>Стандартный ввод</b>
1000000
<b>Стандартный вывод</b>
999997000002000000

**Решение**

Ниже представлено решение на языке C++.

**C++**

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      int t;
6      cin >> t;
7      cout << ((t * (t - 1)) * (t - 2)) << endl;
8  }
```

**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  $1 \leq a, b, c, d \leq 10^9$ .

**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*

<b>Стандартный ввод</b>
5 10 9 6
<b>Стандартный вывод</b>
5

*Пример №2*

<b>Стандартный ввод</b>
4 10 9 6
<b>Стандартный вывод</b>
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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      double a, b, c, d;
6      cin >> a >> b >> c >> d;
7      double a0 = min({a, b, c, d});
8      double a1 = min(max(a, b) / 2.0, min(a, b));
9      double a2 = min(max(c, d) / 2.0, min(c, d));
10     double ans = max({a0, a1, a2});
11     if( (int)ans == ans ){
12         int ians = ans;
13         cout << ians << endl;
14         return 0;
15     }
16     cout.precision(1);
17     cout << fixed<< ans << endl;
18 }
```

**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  $1 \leq n \leq 10^5$ .

Во второй строке заданы  $2n$  целых чисел  $d_i$  — длины всех кусков, которые получились после «тренировки» внука, где  $1 \leq d_i \leq 10^9$ . 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

<b>Стандартный ввод</b>
3 4 8 2 3 6 7
<b>Стандартный вывод</b>
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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      int n;
6      cin >> n;
7      vector<pair<int, int> > v(2 * n);
8      for(int i = 0; i < 2 * n; i++){
9          int d;
10         cin >> d;
11         v[i] = {d, i + 1};
12     }
13     sort(v.begin(), v.end());
14     vector<pair<int, int> > ans(n);
15     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  $2 \leq a \leq 10^{18}$ ,  $1 \leq b, c \leq 10^{18}$ .

#### **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

<b>Стандартный ввод</b>
4 5 41
<b>Стандартный вывод</b>
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++
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  int f1(int M, int a){
5      int res = 0, z = 0;
6      while(1){
7          if(M == 0 && z < a){
8              return res;
9          }
10         res += M;
11         M = M + z;
12         z = M % a;
13         M = M / a;
14     }
15 }

```

```

16 int f2(int M, int b){
17     return M + M / b;
18 }
19 signed main(){
20     int a, b, c;
21     cin >> a >> b >> c;
22     int L = 0, R = 1;
23     while(f1(R, a) <= c){
24         R *= 2;
25     }
26     while(R - L > 1){
27         int M = (R + L) / 2;
28         if(f1(M, a) < c){
29             L = M;
30         }
31         else{
32             R = M;
33         }
34     }
35     int z = R;
36     L = 0, R = 1;
37     while(f2(R, b) <= z){
38         R *= 2;
39     }
40     while(R - L > 1){
41         int M = (R + L) / 2;
42         if(f2(M, b) < z){
43             L = M;
44         }
45         else{
46             R = M;
47         }
48     }
49     cout << R << endl;
50 }

```

### 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  $2 \leq n, m \leq 2 \cdot 10^5$ .

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.  $1 \leq A, B \leq n$ ,  $1 \leq d \leq 10^9$ . 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  $S$ ,  $1 \leq S \leq n$ .

### **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*

<b>Стандартный ввод</b>	
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++**

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  int n, m;
5  vector<vector<pair<int, int> > > G;
6  vector<int> order, used;
7  void dfs(int a){
8      used[a] = 1;
9      for(auto to : G[a]){
10         if(!used[to.first]){
11             dfs(to.first);
12         }
13     }
14     order.push_back(a);
15 }
16 signed main(){
17     cin >> n >> m;
18     G.resize(n + 1);
19     used.resize(n + 1, 0);
20     for(int i = 0; i < m; i++){
21         int a, b, d;
22         cin >> a >> b >> d;
23         G[a].push_back({b, d});
24     }

```

```

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

```

## 2.2.2. Вторая волна. Задачи 8–11 класса

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

### Задача 2.2.2.1. Игра на планшете (10 баллов)

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

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

**Ограничение по времени выполнения программы:** 1 с.

**Ограничение по памяти:** 64 Мбайт.

#### Условие

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

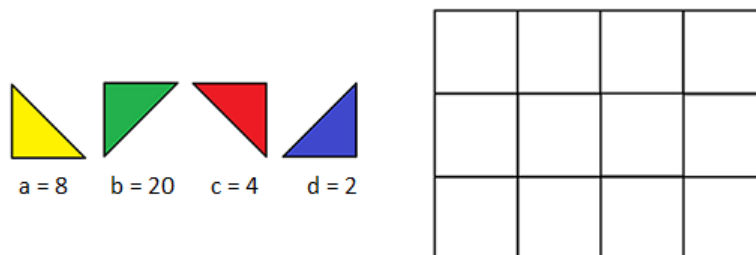


Рис. 2.2.1

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

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

### **Формат входных данных**

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

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

Все числа в пределах от 1 до  $10^9$ .

### **Формат выходных данных**

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

### **Примеры**

#### *Пример №1*

<b>Стандартный ввод</b>
8 20 4 2
3 4
<b>Стандартный вывод</b>
18

### **Примечания**

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

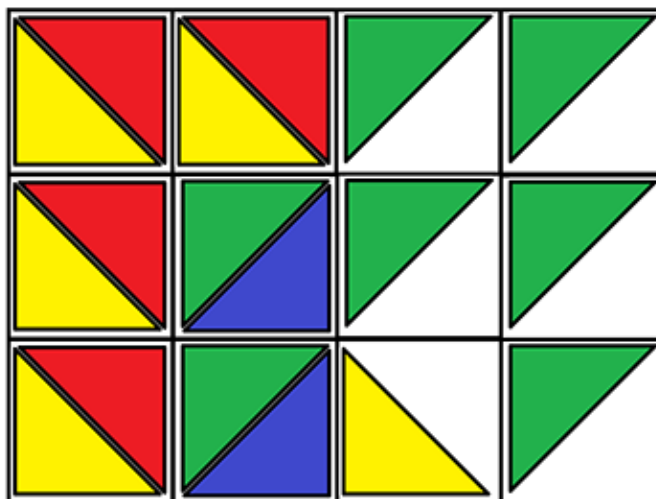


Рис. 2.2.2

**Решение**

Ниже представлено решение на языке C++.

C++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      int a, b, c, d, n, m;
6      cin >> a >> b >> c >> d >> n >> m;
7      if(a > c){
8          swap(a, c);
9      }
10     if(b > d){
11         swap(b, d);
12     }
13     int f = a + b;
14     int k = n * m;
15     if(k <= f){
16         cout << k * 2;
17         return 0;
18     }
19     k -= f;
20     c -= a;
21     d -= b;
22     cout << f * 2 + min(k, c + d) << endl;
23 }
```

**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  $1 \leq n \cdot k \leq 10^{18}$ .

### **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*

<b>Стандартный ввод</b>
45 2
<b>Стандартный вывод</b>
8 12 5 20

#### *Пример №2*

<b>Стандартный ввод</b>
128 7
<b>Стандартный вывод</b>
7 21 2 98

**Решение**

Ниже представлено решение на языке C++.

```

C++
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      int n, k;
6      cin >> n >> k;
7      int x = (k * n) / (k * k + 2 * k + 1);
8      cout << x - k << ' ' << x + k << ' ' << x / k << ' ' << x * k << endl;
9  }

```

**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,  $2 \leq n, m \leq 25$ . Во второй строке через пробел находятся координаты  $(x, y)$  обязательной для посещения клетки, где  $1 \leq x \leq n, 1 \leq y \leq m$ . 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

<b>Стандартный ввод</b>
3 4 2 3
<b>Стандартный вывод</b>
6

## Примечания

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

## Комментарий

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

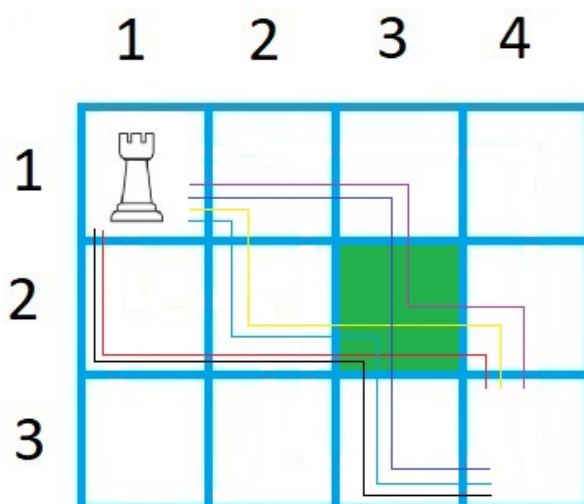


Рис. 2.2.3

## Решение

Ниже представлено решение на языке C++.

C++

```

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

```

```

9         bc[i][j] += bc[i - 1][j];
10        if(j - 1 >= 0){
11            bc[i][j] += bc[i - 1][j - 1];
12        }
13    }
14 }
15 int n, m, x, y;
16 cin >> n >> m >> x >> y;
17 int d1 = bc[x - 1 + y - 1][x - 1];
18 int d2 = bc[n - x + m - y][n - x];
19 int ans = d1 * d2;
20 cout << ans << endl;
21 }

```

### **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 появиться 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

<b>Стандартный ввод</b>
1456531355
<b>Стандартный вывод</b>
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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      string s;
6      cin >> s;
7      string mx = s, mn = s;
8
9      for(int i = 0; i < 5; i++){
10         for(int j = 0; j < 10; j++){
11             mx = max(mx, s);
12             mn = min(s, mn);
13             if(j < 9){
14                 s = s.substr(1) + s[0];
15             }
16         }
17         for(int j = 0; j < 5; j++){
18             swap(s[2 * j], s[2 * j + 1]);
19         }
20     }
21     stringstream ssmn;
22     ssmn << mn;
23     int imn;
24     ssmn >> imn;
25     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  $1, \dots, n$ .

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  $1 \leq n \leq 2 \cdot 10^5$ .

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

<b>Стандартный ввод</b>
9
2 9 1 5 6 4 3 8 7
<b>Стандартный вывод</b>
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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int LG = 19;
5  int N = (1 << LG);
6  vector<int> tr(2 * N, 0);
7  void upd(int pos, int x){
8      pos += N;
9      tr[pos] = x;
10     pos /= 2;
11     while(pos){
12         tr[pos] = {tr[2 * pos]+ tr[2 * pos + 1]};
13         pos /= 2;
14     }
15 }
16 int get(int l, int r){
17     l += N;
18     r += N;
19     int res = 0;
20     while(l <= r){
21         if(l % 2 == 1){
22             res += tr[l];
23         }
24         if(r % 2 == 0){
25             res += tr[r];
26         }
27         l = (l + 1) / 2;
28         r = (r - 1) / 2;
29     }
30     return res;
31 }
32 signed main(){
33     int n, a;
34     cin >> n;
35     int ans = 0;
36     for(int i = 0; i < n; i++){
37         cin >> a;
38         int sl = get(0, a - 1);
39         int sr = get(a + 1, N - 1);
40         ans += min(sl, sr);
41         upd(a, a);
42     }
43     cout << ans << endl;
44 }
```

**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

<b>Стандартный ввод</b>
1 5 10
<b>Стандартный вывод</b>
3

### Решение

Ниже представлено решение на языке C++.

C++

```
1 #include<bits/stdc++.h>
2 #define int long long
3 using namespace std;
4 signed main(){
5     int A, C, D;
6     cin >> A >> C >> D;
7     cout.precision(1);
8     cout << fixed << (A + C) / 2.0 << endl;
9 }
```

### 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

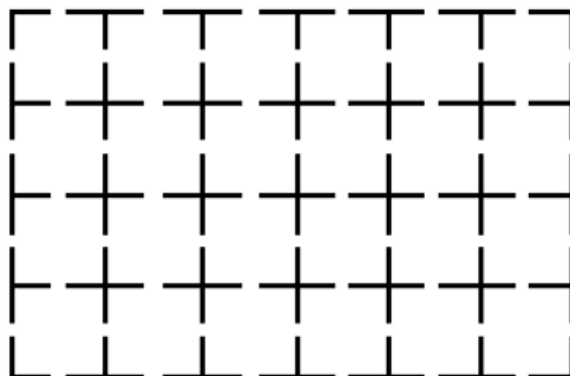


Рис. 2.2.4

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  $1 \leq n, m \leq 10^9$ .

#### Формат выходных данных

Требуется по числам  $a$  и  $b$  найти размеры исходной решетки  $n$  и  $m$ . Будем всегда считать, что  $n \leq m$ , 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

<b>Стандартный ввод</b>
16 15
<b>Стандартный вывод</b>
4 6

### Пример №2

<b>Стандартный ввод</b>
0 0
<b>Стандартный вывод</b>
1 1

## Комментарий

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

## Решение

Ниже представлено решение на языке C++ при помощи бинарного поиска.

C++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      int a, b;
6      cin >> a >> b;
7      int L = 0, R = a / 4 + 1;
8      while(R - L > 1){
9          int M = (R + L) / 2;
10         int D = a / 2 - M;
11         if(M * D <= b){
12             L = M;
13         }
14         else{
15             R = M;
16         }
17     }
18     cout << L + 1 << ' ' << a / 2 - L + 1 << endl;
19 }
```

### **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 свечку и предложил ему решить задачу.

«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  $2 \leq n, m \leq 1000$ .

Во второй строке находятся два числа  $x$  и  $y$  через пробел — координаты свечки, где  $1 \leq x \leq n - 1, 1 \leq y \leq m - 1$ , 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*

<b>Стандартный ввод</b>
8 5 7 2
<b>Стандартный вывод</b>
12.571

*Пример №2*

<b>Стандартный ввод</b>
2 2 1 1
<b>Стандартный вывод</b>
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.

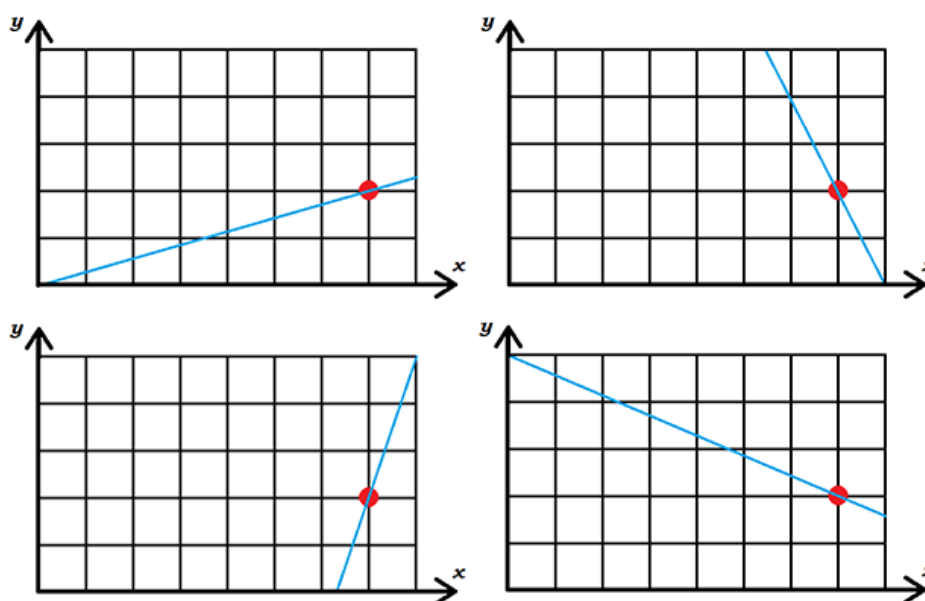


Рис. 2.2.5

## 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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int INF = 1e18;
5  double katy(double x, double y, double n){
6      return n * y / x;
7  }
8  double n, m, x, y;
9  double ans = INF;
10 double k1, k2;
11 void upd(){
12     if(k1 < m){
13         double st = k1 * n / 2;
14         ans = min(ans, n * m - 2 * st);
15     }
16     else{
17         double st = k2 * m / 2;
18         ans = min(ans, n * m - 2 * st);
19     }
20 }
21 signed main(){
22     cin >> n >> m >> x >> y;
23     k1 = katy(x, y, n);
24     k2 = katy(y, x, m);
25     upd();
26     k1 = katy(n - x, y, n);
27     k2 = katy(y, n - x, m);
28     upd();
29     k1 = katy(x, m - y, n);
30     k2 = katy(m - y, x, m);
31     upd();
32     k1 = katy(n - x, m - y, n);
33     k2 = katy(m - y, n - x, m);
34     upd();
35     cout.precision(3);
36     cout << fixed << ans << endl;
37 }
```

## 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  $2 \leq n \leq 2 \cdot 10^5$ . Во второй строке через пробел заданы  $n$  целых чисел  $a_i$  через пробел — длины полосок, где  $1 \leq a_i \leq 10^9$ .

### **Формат выходных данных**

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

### **Примеры**

#### *Пример №1*

<b>Стандартный ввод</b>
8 3 6 5 4 4 5 5 2
<b>Стандартный вывод</b>
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  $3 \times 4$ , и ее площадь 12. При ее создании Маша не должна использовать полоску номер 8, по этой причине неважно, как она раскрашена.

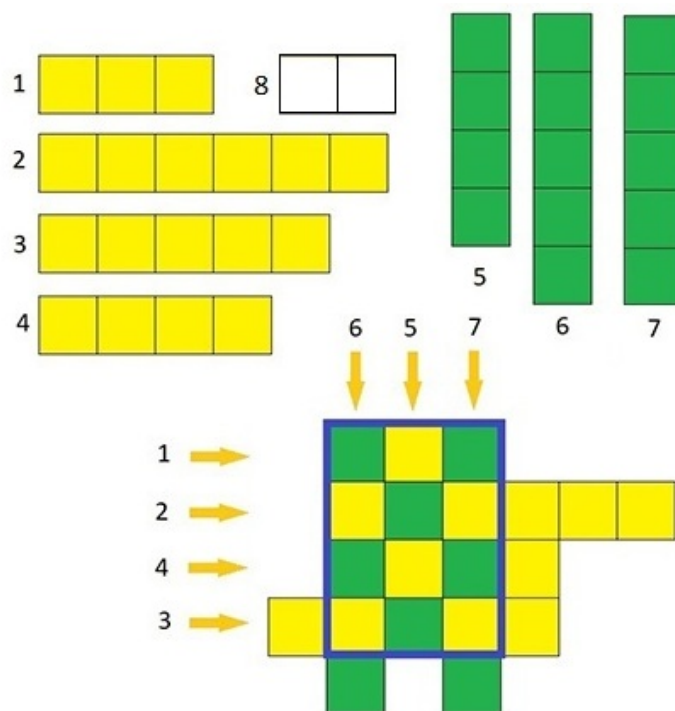


Рис. 2.2.6

**Решение**

Ниже представлено решение на языке C++.

**C++**

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      int n;
6      cin >> n;
7      deque<int> v(n);
8      for(int i = 0; i < n; i++){
9          cin >> v[i];
10     }
11     sort(v.begin(), v.end());
12     int ans = 0;
13     int cnth = 0, minh;
14     while(1){
15         if(v.size() == 0){
16             break;
17         }
18         cnth++;
19         minh = v.back();
20         v.pop_back();
21         while(v.size() > 0 && v[0] < cnth){
22             v.pop_front();
23         }
24         ans = max(ans, cnth * min(minh, (int)v.size()));
25     }
26     cout << ans << endl;
27 }
```

### **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  $20 \leq n \leq 100$ , 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,  $20 \leq n \leq 100$ .

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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  string frs;
5  int n;
6  vector<string> dict;
7  vector<int> msk(26, 0);
8  int cnt = 0;
9  vector<vector<int>> amsk;
10 vector<string> ans;
11 bool bigok = 0;
12 void p(int pos){
13     if(!bigok){
14         if(cnt == 0){
15             sort(ans.begin(), ans.end());
16             bigok = 1;
17             return;
18         }
19         for(int i = pos; i < n; i++){
20             string ts = dict[i];
21             bool ok = 1;
22             for(int j = 0; j < 26; j++){
23                 if(amsk[i][j] > msk[j]){
24                     ok = 0;
25                 }
26             }
27             if(ok){
28                 ans.push_back(ts);
29                 for(int j = 0; j < 26; j++){
30                     msk[j] -= amsk[i][j];
31                     cnt -= amsk[i][j];
32                 }
33                 p(i + 1);
34                 if(!bigok){
35                     for(int j = 0; j < 26; j++){
36                         msk[j] += amsk[i][j];
37                         cnt += amsk[i][j];
38                     }
39                 }
40                 ans.pop_back();
41             }
42         }
43     }
44 }
45 signed main(){
46     cin >> frs;
47     cin >> n;
48     amsk.resize(n, vector<int>(26, 0));
49
50     string ts;
51     for(int i = 0; i < n; i++){
52         cin >> ts;
53         dict.push_back(ts);
54     }
55     for(int i = 0; i < n; i++){
56         for(auto el : dict[i]){
57             amsk[i][el - 'a']++;
58         }
59     }

```

```
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  $1 \leq a, b, c, d \leq 10^9$ .

#### **Формат выходных данных**

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

**Примеры***Пример №1*

<b>Стандартный ввод</b>
2 4
3 6
<b>Стандартный вывод</b>
4

*Пример №2*

<b>Стандартный ввод</b>
2 2
3 6
<b>Стандартный вывод</b>
3

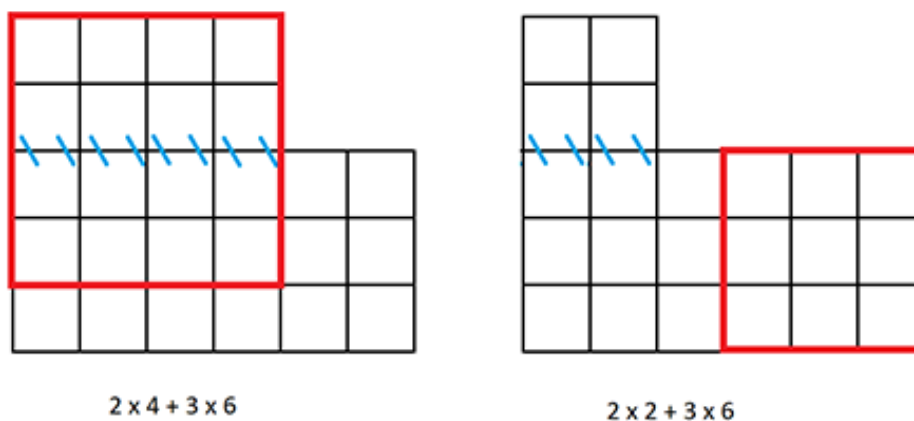
**Примечания**

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++

```

1 #include<bits/stdc++.h>
2 #define int long long
3 using namespace std;
4 signed main(){
5     int a, b, c, d;
6     cin >> a >> b >> c >> d;
7     int ans = max(min(a, b), min(c, d));
8     int p1 = min(a + c, min(b, d));
9     int p2 = min(a + d, min(b, c));
10    int p3 = min(b + c, min(a, d));
11    int p4 = min(b + d, min(a, c));
12    ans = max({ans, p1, p2, p3, p4});
13    cout << ans << endl;
14 }

```

### 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 А, С, G, Т.

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  $D_1 = \mathbf{AACGGTAGGT}$ , а  $D_2 = \mathbf{TCCCGGAACA}$ . Выберем  $i = 4$  и поменяем местами префиксы длины 4. Получим две новые ДНК, одна из которых будет иметь вид  $\mathbf{AACGGGAACA}$ , а вторая —  $\mathbf{TCCCGTAGGT}$ . 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  $10 \leq n \leq 100$ ,  $2 \leq m \leq 100$ .

В следующих  $m$  строках содержится описание исходной популяции ДНК, каждая задается строкой длины  $n$ , состоящей из символов А, С, G и Т.

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*

<b>Стандартный ввод</b>
10 2
AACGGTAGGT
TCCCGGAACA
-----
TCCCGTAGGT

<b>Стандартный вывод</b>
AACGGGAACA

*Пример №2*

<b>Стандартный ввод</b>
10 4
AACCGGTТАА
ACGTACGTAC
AAACCCGGGT
САТТАСТGGA
-----
AAGCGCTТАА
ССАСАСGТGC
ААСТАGGGGT

<b>Стандартный вывод</b>
ААТТССТGAA

## 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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  signed main(){
5      int n, m;
6      cin >> n >> m;
7      vector<string> v1(m);
8      for(int i = 0; i < m; i++){
9          cin >> v1[i];
10     }
11     string d;
12     cin >> d;
13     vector<string> v2(m - 1);
14     for(int i = 0; i < m - 1; i++){
15         cin >> v2[i];
16     }
17     for(int j = 0; j < n; j++){
18         int ss = 0;
19         for(int i = 0; i < m; i++){
20             ss ^= (int)(v1[i][j]);
21         }
22         for(int i = 0; i < m - 1; i++){
23             ss ^= (int)(v2[i][j]);
24         }
25         cout << (char)(ss);
26     }
27     cout << endl;
28 }
```

### 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  $1 \leq A, B, C, D \leq 1000$ ,  $0 \leq E \leq 1000$ .

### **Формат выходных данных**

Вывести одно число — минимальное время прохождения туннеля всей группой.

### **Примеры**

#### *Пример №1*

<b>Стандартный ввод</b>
8 9 10 1
3
<b>Стандартный вывод</b>
44

#### *Пример №2*

<b>Стандартный ввод</b>
8 9 10 1
0
<b>Стандартный вывод</b>
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++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int INF = 1e18;
5  vector<int> v(4);
6  int e, ans = INF;
7  void p(vector<int> &v1, vector<int> &vr, int tv){
8      if(v1.size() == 2){
9          ans = min(ans, tv + *max_element(v1.begin(), v1.end()));
10         return;
11     }
12     for(int i = 0; i < v1.size() - 1; i++){
13         for(int j = i + 1; j < v1.size(); j++){
14             vector<int> v11;
15             for(int k = 0; k < v1.size(); k++){
16                 if(k != i && k != j){
17                     v11.push_back(v1[k]);
18                 }
19             }
20             vector<int> vr1 = vr;
```

```

21         vrl.push_back(vl[i] + e);
22         vrl.push_back(vl[j] + e);
23         int tmp = max(vl[i], vl[j]);
24         sort(vrl.rbegin(), vrl.rend());
25         vll.push_back(vrl.back() + e);
26         vrl.pop_back();
27         p(vll, vrl, tv + tmp + vll.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  $2 \leq a \leq 10^6$ ,  $0 \leq b \leq 10^6$ .

### 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

<b>Стандартный ввод</b>
4 3
<b>Стандартный вывод</b>
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++

```

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

```

```

7  int binpow (int a, int n) {
8      int res = 1;
9      while (n > 0) {
10         if (n % 2 == 1)
11             (res *= a) %= MOD;
12         (a *= a) %= MOD;
13         n /= 2;
14     }
15     return res;
16 }
17
18 int bc(int n, int k){
19     int res = f[n];
20     int p1 = binpow(f[k], MOD - 2);
21     int p2 = binpow(f[n - k], MOD - 2);
22     (res *= p1) %= MOD;
23     (res *= p2) %= MOD;
24     return res;
25 }
26 signed main(){
27     for(int i = 1; i <= 2e6; i++){
28         f[i] = (f[i - 1] * i) % MOD;
29     }
30     int a, b;
31     int ans = 0;
32     cin >> a >> b;
33     a--;
34     for(int i = 0; i < a + 1; i++){
35         if(2 * i <= b){
36             int d = bc(a, i);
37             if(b - 2 * i <= a - i){
38                 (d *= bc(a - i, b - 2 * i) ) %= MOD;
39                 (ans += d) %= MOD;
40             }
41         }
42     }
43     cout << ans << endl;
44 }

```

### 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 прогуливающих джентльменов, где  $1 \leq d \leq 200$ ,  $2 \leq n \leq 2000$ .

Во второй строке находятся  $n$  целых чисел  $v_i$  через пробел — скорости каждого джентльмена, где  $1 \leq v_i \leq 2000$ . Гарантируется, что все скорости попарно различны. Скорости даны в порядке возрастания, то есть  $v_1 < v_2 < \dots < v_n$ .

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

### **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*

<b>Стандартный ввод</b>
5 4 2 5 8 10 6
<b>Стандартный вывод</b>
0.000

*Пример №2*

<b>Стандартный ввод</b>
5 4 2 5 8 10 7
<b>Стандартный вывод</b>
0.556

*Пример №3*

<b>Стандартный ввод</b>
5 4 2 5 8 10 11
<b>Стандартный вывод</b>
1.000

*Пример №4*

<b>Стандартный ввод</b>
5 4 2 5 8 10 15
<b>Стандартный вывод</b>
1.429

*Пример №5*

<b>Стандартный ввод</b>
5 4 2 5 8 10 17
<b>Стандартный вывод</b>
1.667

*Пример №6*

<b>Стандартный ввод</b>
5 4 2 5 8 10 19
<b>Стандартный вывод</b>
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.

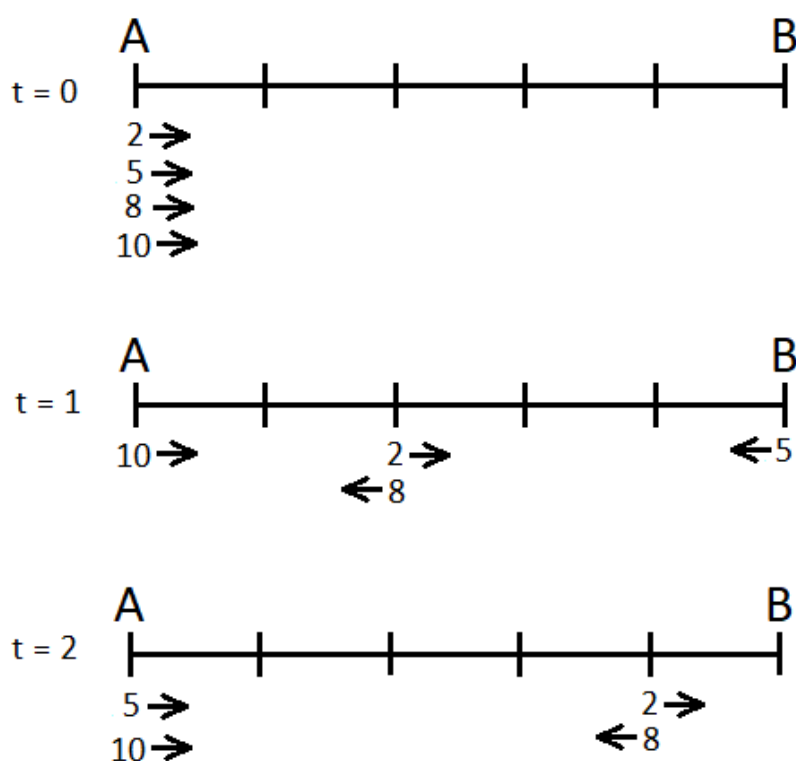


Рис. 2.2.8

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 бинарного поиска с квадратичным нахождением ответа в каждой его итерации.

### Решение

Ниже представлено решение на языке C++.

C++

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const double EPS = 1e-7;
5  double x(double M, int V, int d){
6      double dst = V * M;
7      int cnt = floor((dst + EPS) / d);
8      double pin = dst - cnt * d;
9      if(cnt % 2 == 0){
10         return pin;
11     }
12     else{
13         return d - pin;
14     }
15 }
16 int F(double M, vector<int> &v, int d){
17     int res = 0;
18     for(int i = 0; i < v.size(); i++){
19         double dst = v[i] * M;

```

```
20     int cnt = floor((dst + EPS) / d);
21     res += cnt * i;
22     double tx = x(M, v[i], d);
23     for(int j = 0; j < i; j++){
24         double txj = x(M, v[j], d);
25         if(cnt % 2 == 0){
26             res += txj <= tx + EPS;
27         }
28         else{
29             res += txj >= tx - EPS;
30         }
31     }
32 }
33 return res;
34 }
35 signed main(){
36     int d, n;
37     cin >> d >> n;
38     vector<int> v(n);
39     for(int i = 0; i < n; i++){
40         cin >> v[i];
41     }
42     int k;
43     cin >> k;
44     double L = 0, R = 1;
45     while(F(R, v, d) <= k){
46         R *= 2;
47     }
48     R /= 2;
49     while(R - L > 1e-4){
50         double M = (R + L) / 2.0;
51         if(F(M, v, d) < k){
52             L = M;
53         }
54         else{
55             R = M;
56         }
57     }
58     cout.precision(10);
59     cout << fixed << L << endl;
60 }
```

## 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://contests.yandex.ru/contest/63459/enter/>.

#### **Задача 2.3.1.1. (15 баллов)**

Тема: арифметика.

##### **Условие**

Оля в каждую клетку таблицы  $3 \times 3$  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  $2 \times 2$  равна 32. Какое число Оля записала в центральную клетку таблицы?

##### **Решение**

Проанализируем исходную таблицу и увидим, что при построении всех возможных квадратов  $2 \times 2$ :

- 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  $2 \times 2$  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

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

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  $\frac{192}{900\,000} \approx 0,0002$ .

**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 съесть  $60 \cdot 0,35 = 21$  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 получают 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)**

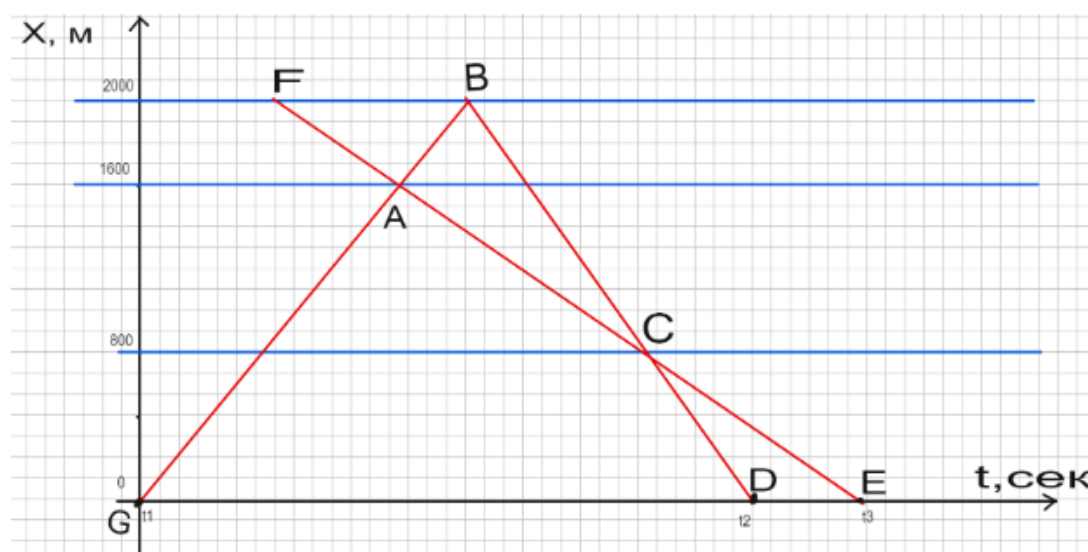
Темы: текстовые задачи и логика, графическое изображение движения, теорема Менелая.

**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 скоростью. 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:

$$h_1 = 2000 - 400 = 1600, \quad h(2) = 800, \quad H = 2000, \quad t_1 = 6, \quad t_2 = 12.$$

По теореме Менелая получаем:

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

Выразим эти отношения из разных пар подобных треугольников:

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

Подставим числа:

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

**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  $0 - 1 - 2 - 3 - 4 - 0 - \dots$  (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 выбрать 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,  $\angle ABC = \angle ABD = \angle AED = 90^\circ$ .

**Ответ:**  $90^\circ$ .

### **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  $a + b = 11(x + y) : 5$  и числа 5 и 11 взаимно просты, то

$$(x + y) : 5. \quad (2.3.1)$$

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

$$(x - y) : 3. \quad (2.3.2)$$

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,  $y : 11$  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  $Q = \frac{P}{2}$  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  $Q = \frac{P}{2}$ . 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 размножиться, 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**

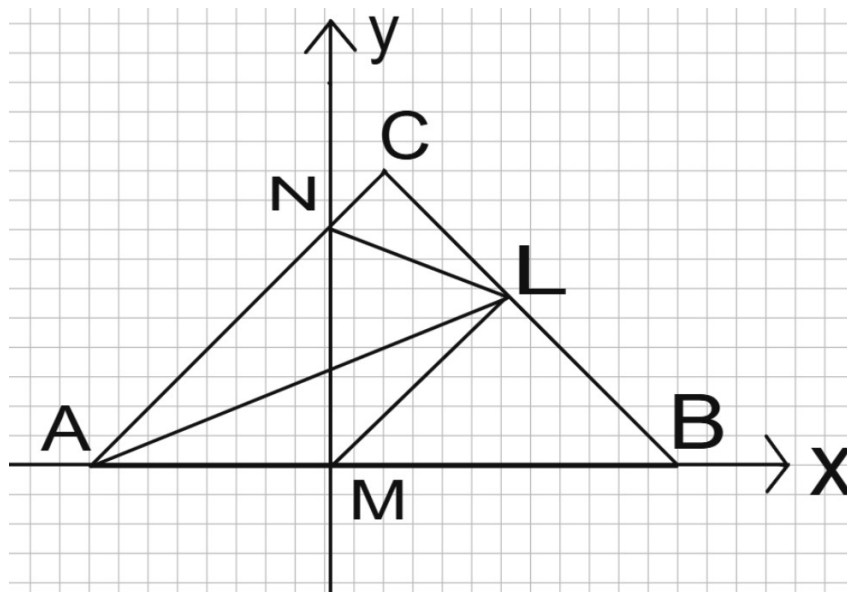
$MN = AM$ , значит, угол  $ANM$  тоже равен  $45^\circ$  и  $NM$  перпендикулярно  $AB$ .

Тогда углы  $NML$  и  $BML$  тоже равны по  $45^\circ$ .

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:

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

Тогда угол  $ANL$  равен  $180^\circ$  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:

$$\begin{aligned} 16 \cdot x^2 &= a \cdot (x - x_0) + 16 \cdot (x_0)^2; \\ 16 \cdot x^2 - a \cdot x + x_0 \cdot a - 16 \cdot (x_0)^2 &= 0; \\ D = a^2 - 4 \cdot 16 \cdot (x_0 \cdot a - 16 \cdot x_0^2) &= (a - 32 \cdot x_0)^2 = 0; \\ a &= 32 \cdot x_0. \end{aligned}$$

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  $x_1 = \frac{x_0}{2}$ .

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:

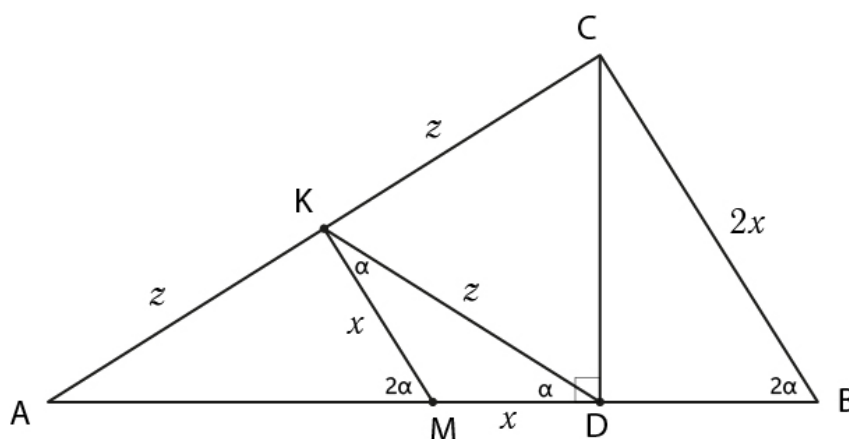
$$71 = x(n - 2k + m).$$

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  $ADC$ :  $DK = \frac{1}{2}AC = AK$ .

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  $\Phi$  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  $\Phi$ , равно дроби, в числителе которой шесть, а в знаменателе квадратный корень из числа  $\pi$ .

Чему равна площадь фигуры  $\Phi$ ?

#### Решение

- 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  $\Phi$ .
  - 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  $\Phi$  находятся на одинаковом расстоянии  $R$  от точки  $O$ . По определению, это окружность.
- Так как границей является окружность, то сама фигура — круг.
- 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:

$$\begin{array}{l} 20\% - 900, \\ 100\% - ? \end{array}$$

Значит, к концу первого часа оставалось  $900 : 0,2 = 4500$  г анобтаниума.

Найдем массу топлива к началу первого часа.

Не было израсходовано главным двигателем к этому моменту  $4500 + 300 = 4800$  г, что составляет  $100 - 40 = 60\%$ .

Составим пропорцию и решим ее:

$$\begin{array}{l} 60\% - 4800, \\ 100\% - ? \end{array}$$

Значит, к началу первого часа было  $4800 : 0,6 = 8000$  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)

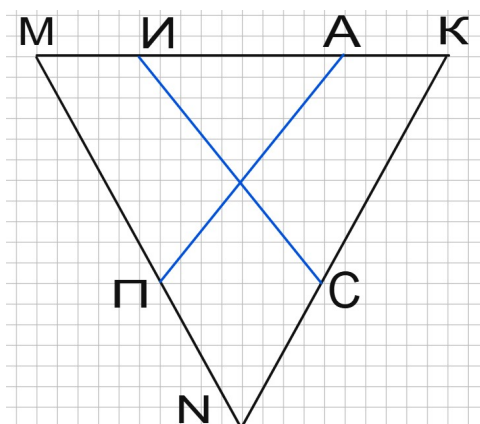
*Темы: равносторонний треугольник, первый признак равенства треугольников.*

#### 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:

$$MA + MP = KI + KC, \text{ а значит, } MI + MP = KA + KC = 1000 \text{ м.}$$

$$MI + KA = IA = 500 \text{ м. Кроме того, длина KM равна } 1000 \text{ м.}$$

Отсюда выходит, что  $KA = 1000 - KC = 1000 - MA$ , а значит,  $KC = MA$ . Аналогично выходит, что  $KI = MP$ .

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

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

**Ответ:** 250.

### **Задача 2.3.6.5. (30 баллов)**

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

#### **Условие**

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

#### **Решение**

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

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

Третья часть всех делителей не делится на 3.

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

Половина четных делителей делится на 5.

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

Если перемножим  $2^5$  на  $3^2$  и на 5, то получим 1440. Минимальное число, подходящее под условия выше, но большее этого числа, равно  $7 \cdot 1440 = 10080 > 10000$ .

Следовательно, под все условия подходит только число 1440.

**Ответ:** 1440.

## **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

$$\begin{aligned} x^2 + y^2 &= (x + y)^2 - 2xy = (2a + 1)^2 - 2(4a^2 + 8a - 4) = 4a^2 + 4a + 1 - 8a^2 - 16a + 8 = \\ &= -4a^2 - 12a + 9 = -(4a^2 + 12a + 9) + 18 = -(2a + 3)^2 + 18. \end{aligned}$$

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.

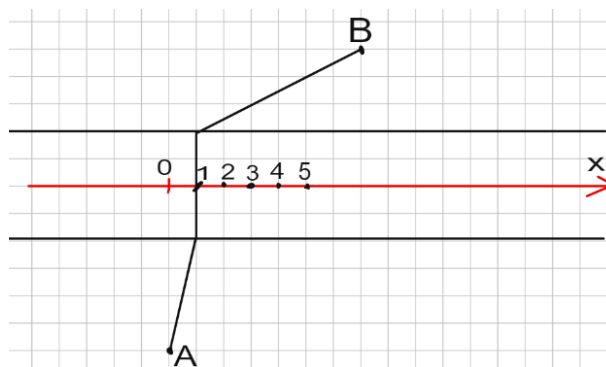


Рис. 2.3.1

**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.

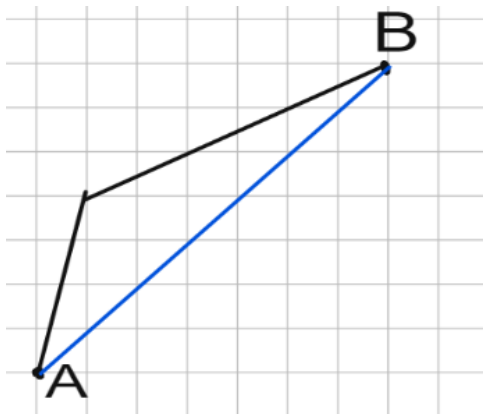


Рис. 2.3.2

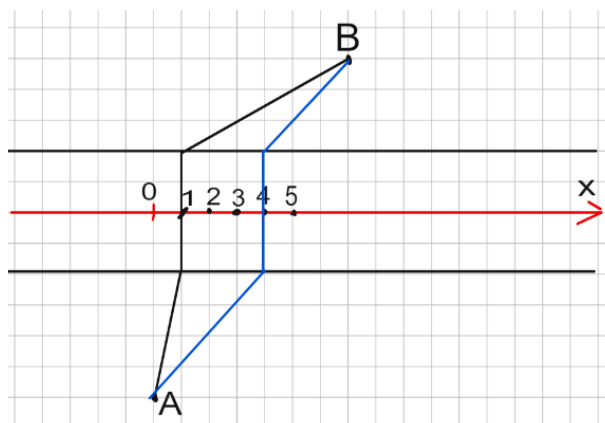


Рис. 2.3.3

**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 пятёрки, 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.

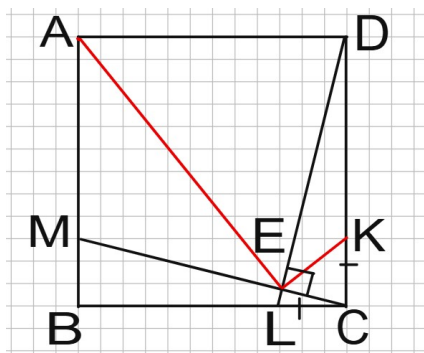


Рис. 2.3.4

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

Тогда

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

Так как вектора  $LE$  и  $LD$  должны быть сонаправлены, то

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

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

Посчитаем скалярное произведение:  $\vec{AE} \cdot \vec{EK} = 0$ . 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. Машинное обучение (100 баллов)

Тема: классификация.

#### Условие

Требуется построить модель на объектах из тренировочного набора данных и с ее помощью предсказать класс для объектов из тестового набора данных. Решение будет оценено при помощи F-меры.

Материалы задачи: <https://disk.yandex.ru/d/900jxWTev0v-NQ>.

Файл `train.tsv` содержит объекты из тренировочного набора данных и состоит из следующих колонок:

- `id` — идентификатор объекта;
- $x_1 \dots x_{1000}$  — числовые признаки, среди их значений могут быть пропуски;
- `y` — класс объекта  $P$  или  $N$ .

Файл `test.tsv` содержит объекты из тестового набора данных. Он схож по формату с `train.tsv`, но не содержит класс `y`.

Требуется загрузить `tsv`-файл с предсказанием классов для объектов из тестового набора данных.

#### Решение

Для решения задачи необходимо считать тренировочную часть набора данных, заполнить пропуски в данных и нормализовать их. Затем обучить модель классификации и построить предсказание для тестовой части набора данных, произведя те же операции по преобразованию с ней.

Ниже представлено решение на языке Python.

#### Python

```
1 from sklearn.dummy import DummyClassifier
2 import pandas as pd
3
4 train = pd.read_csv("train.tsv", sep='\t')
5 test = pd.read_csv("test.tsv", sep='\t')
6
7 trainX = train.drop(columns=["y", "id"])
8 trainY = train["y"]
9 testX = test.drop(columns=["id"])
10 submission = test[["id"]].copy()
11
12 clsf = DummyClassifier(random_state=0)
13 clsf.fit(trainX, trainY)
14 prediction = clsf.predict(testX)
```

```
15
16 submission["y"] = prediction
17
18 submission.to_csv("submission.tsv", sep='\t')
```

## 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](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

### Задача 3.2.1. Нормальная регрессия (100 баллов)

Темы: линейная регрессия, нормализация.

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

Игорю и Олегу поручили построить линейную регрессию на наборе данных.

Сначала Игорь преобразовал все признаки, включая целевой минимакс нормализацией, а затем Олег построил линейную регрессию на нем. Оказалось, что полученное уравнение прямой не подходит для дальнейшего предсказания, так как оно аппроксимирует нормализованные, а не оригинальные данные. Тогда Олег решил построить уравнение прямой на оригинальном наборе данных, но оказалось, что он потерялся.

Так как Игорь занимался нормализацией, он разобрался, как восстановить оригинальное уравнение прямой. Но Олег не поверил ему, он решил, что Игорь как-то запомнил оригинальный набор данных и просто построил уравнения прямой для него. Тогда Игорь предложил Олегу как-нибудь изменить информацию о нормализации и уравнении прямой, чтобы подтвердить, что оригинальное уравнение прямой возможно восстановить и без набора данных.

Олег согласился на предложение Игоря, но теперь ему самому нужно узнать правильное решение задачи для проверки Игоря.

#### Формат входных данных

Первая строка содержит одно натуральное число  $N$  ( $1 \leq N \leq 10^5$ ) — число признаков в наборе данных без учета целевого признака.

Вторая строка содержит  $(N + 1)$  разделенное пробелом целое число  $l_i$  ( $|l_i| \leq 10^9$ ) — минимальное значение для каждого признака до нормализации. Первые  $N$  из этих чисел соответствуют обычным признакам, а последнее — целевому.

Третья строка содержит  $(N + 1)$  разделенное пробелом целое число  $u_i$  ( $|u_i| \leq 10^9$ ) — максимальные значения в аналогичном формате. Гарантируется, что все максимумы строго больше соответствующих минимумов.

Четвертая строка содержит  $(N + 1)$  разделенное пробелом целое число  $c_i$  ( $|c_i| \leq 10^9$ ) — коэффициенты прямой, полученной после нормализации. Первые  $N$  из этих чисел соответствуют признакам, а последнее — свободный коэффициент.

Пятая строка содержит одно натуральное число  $M$  ( $1 \leq M \leq 10^5$ ) — число

запросов Олега.

Далее идут  $M$  строк, каждая содержит описание соответствующего запроса. Каждый запрос начинается с одной маленькой латинской буквы  $l$ ,  $u$ ,  $s$  или  $q$  в зависимости от его вида:

- запрос вида  $l\ i\ v$  — запрос на изменение минимального значения до нормализации для  $i$ -го признака на  $v$ ;
- запрос вида  $u\ i\ v$  — запрос на изменение максимального значения до нормализации для  $i$ -го признака на  $v$ ;
- запрос вида  $s\ i\ v$  — запрос на изменение  $i$ -го коэффициента прямой на  $v$ ;
- запрос вида  $q\ i$  — запрос значения  $i$ -го коэффициента оригинальной прямой.

Гарантируется, что  $i$  — целое число и  $1 \leq i \leq n + 1$ . Если  $i$  равняется  $(n + 1)$  в запросах  $l$  и  $u$ , то это значит, что в запросе изменялся минимум или максимум целевого признака, а в запросах  $s$  и  $q$  имелся в виду свободный коэффициент прямой. Гарантируется, что  $v$  — целое число и  $|v| \leq 10^9$ , а после запросов  $l$  и  $u$  минимальное значение соответствующего признака по-прежнему строго меньше максимального.

### Формат выходных данных

Для каждого запроса коэффициента оригинальной прямой выведите его значение — число с плавающей точкой.

Решение будет засчитано, если абсолютная или относительная погрешность не будет превышать  $10^{-9}$ .

### Примеры

#### Пример №1

Стандартный ввод
4
1 1 1 1 0
2 2 2 2 1
1 2 3 4 5
5
q 5
l 1 0
u 2 6
s 3 4
q 5
Стандартный вывод
-5
-3.4

### Решение

После минимакс-нормализации уравнение для предсказания выглядит как:

$$y(x) = \sum_{i=1}^n c_i x_i + c_{n+1}.$$

Подставим вместо  $x_i$  нормализованную версию  $x_i = \frac{x'_i - l_i}{u_i - l_i}$ , где  $x'$  — оригинальные данные до нормализации:

$$y(x) = \sum_{i=1}^n c_i \frac{x'_i - l_i}{u_i - l_i} + c_{n+1}.$$

Но полученное предсказание будет для нормализованного целевого признака, поэтому его нужно преобразовать обратным преобразованием:

$$y_d(x) = (u_{n+1} - l_{n+1}) \cdot y(x) + l_{n+1} = (u_{n+1} - l_{n+1}) \cdot \left( \sum_{i=1}^n c_i \frac{x'_i - l_i}{u_i - l_i} + c_{n+1} \right) + l_{n+1}.$$

Так как нормализация и обратное к ней преобразование являются линейными, их можно скомбинировать в новое линейное уравнение:

$$y_d(x) = \sum_{i=1}^n \frac{(u_{n+1} - l_{n+1})c_i}{u_i - l_i} x'_i + (u_{n+1} - l_{n+1}) \left( - \sum_{i=1}^n \frac{c_i l_i}{u_i - l_i} + c_{n+1} \right) + l_{n+1}.$$

В этом уравнении коэффициент при  $x'_i$  равен:  $\frac{(u_{n+1} - l_{n+1})c_i}{u_i - l_i}$ , его можно просто вычислить за  $\mathcal{O}(1)$  при ответе на соответствующий запрос.

В том же уравнении свободный коэффициент равен

$$(u_{n+1} - l_{n+1}) \left( c_{n+1} - \sum_{i=1}^n \frac{c_i \cdot l_i}{u_i - l_i} \right) + l_{n+1}.$$

Наивное его вычисление потребует  $\mathcal{O}(N)$  времени, так как содержит в себе сумму  $\sum_{i=1}^n \frac{c_i \cdot l_i}{u_i - l_i}$ . Но эту сумму можно предподсчитать, а затем обновлять соответствующие слагаемое при запросе на изменение. Тогда на запрос значения свободного коэффициента можно отвечать за  $\mathcal{O}(1)$ .

### Пример программы-решения

Ниже представлено решение на языке Python.

#### Python

```

1  n = int(input())
2
3  l = list(map(int, input().split()))
4  u = list(map(int, input().split()))
5  c = list(map(int, input().split()))
6
7  total_sum = 0

```

```

8
9  for i in range(n):
10     total_sum += l[i] * c[i] / (u[i] - l[i])
11
12  m = int(input())
13
14  for _ in range(m):
15     row = list(input().split())
16
17
18     t = row[0]
19     i = int(row[1]) - 1
20
21     if t == 'q':
22         if i == n:
23             print((c[n] - total_sum) * (u[n] - l[n]) + l[n])
24         else:
25             print(c[i] * (u[n] - l[n]) / (u[i] - l[i]))
26     else:
27         v = int(row[2])
28         if i != n:
29             total_sum -= l[i] * c[i] / (u[i] - l[i])
30
31         if t == 'l':
32             l[i] = v
33         elif t == 'u':
34             u[i] = v
35         else:
36             c[i] = v
37
38         if i != n:
39             total_sum += l[i] * c[i] / (u[i] - l[i])

```

### Задача 3.2.2. Дикий лес (100 баллов)

Тема: сортировки.

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

Владимир решил быть ближе к природе и поэтому приехал на выходные в деревню рядом с лесом. Там он обнаружил несколько странно растущих в один ряд деревьев, при этом листва у них была внизу, а сверху они были обтесаны.

Владимиру стало интересно, и он решил исследовать этот ряд деревьев. Так как у Владимира есть всего два дня на исследование, он разбил последовательность из деревьев на две части в зависимости от их высот.

Для определения оптимального разбиения Владимир хочет вычислить средне-квадратичную разность высот деревьев в каждой из частей и между ними. К сожалению, только на эти вычисления у него могут уйти все выходные, поэтому Владимир

попросил помощи.

### **Формат входных данных**

Первая строка содержит натуральное число  $N$  ( $4 \leq N \leq 10^5$ ) — число деревьев в ряду.

Вторая строка содержит  $N$  натуральных чисел  $h_i$  ( $1 \leq h_i \leq 10^9$ ) — высоты соответствующих деревьев.

### **Формат выходных данных**

Выведите  $(N - 3)$  оценки для всех возможных разбиений. Крайние разбиения не учитываются, чтобы избежать деления на 0.

Каждая оценка должна состоять из трех вещественных чисел с плавающей точкой: значение для первой части разбиения, между частями и для второй части. Ответ будет засчитан, если его абсолютная или относительная погрешность не будет превышать  $10^{-9}$ .

### **Примеры**

#### *Пример №1*

<b>Стандартный ввод</b>
5 1 3 8 9 12
<b>Стандартный вывод</b>
2.0 7.916228058025278 2.943920288775949 5.0990195135927845 7.291547618075786 3.0

### **Примечания**

В примере для первого оцениваемого разбиения деревья разбиваются на две части: 1 3 и 8 9 12.

Значение оценки для первой части равно  $\sqrt{\frac{(1-3)^2}{1}}$ .

Значение оценки между частями равно  $\sqrt{\frac{7^2+8^2+11^2+5^2+6^2+9^2}{6}}$ .

Значение оценки для второй части равно  $\sqrt{\frac{1^2+3^2+4^2}{3}}$ .

### **Решение**

Среднее квадратическое вычисляется как корень из суммы квадратов, деленной на число слагаемых в этой сумме. Рассмотрим  $p$ -е разбиение. Оно будет состоять из двух частей. В первой части будет  $l = p + 1$  элемент, а во второй —  $r = n - l = n - p - 1$ . Тогда сумма для оценки среднеквадратической разности для первой

части будет содержать  $l(l-1)/2$  слагаемых, для второй будет  $r(r-1)/2$ , а между частями будет  $lr$ . Потому что ровно столько будет различных пар разностей для соответствующих средних квадратических.

Теперь необходимо научиться быстро вычислять сумму квадратов разностей для различных разбиений и между ними. Для начала решим задачу только для первой части каждого разбиения. Если рассматривать все разбиения в порядке возрастания номера, то каждый раз к первой части будет добавляться один элемент из второй. Получается, что сумма квадратов разностей будет состоять из предыдущей суммы плюс сумма квадратов разности добавляемого элемента  $h_l$  с существующими  $\{h_i\}_{i=1}^{l-1}$ :

$$\sum_{i=1}^l \sum_{j=1}^{i-1} (h_i - h_j)^2 = \sum_{i=1}^{l-1} \sum_{j=1}^{i-1} (h_i - h_j)^2 + \sum_{i=1}^{l-1} (h_i - h_l)^2.$$

Добавляемую сумму можно переписать как:

$$\begin{aligned} \sum_{i=1}^{l-1} (h_i - h_l)^2 &= \sum_{i=1}^{l-1} (h_i^2 - 2h_i h_l + h_l^2) = \sum_{i=1}^{l-1} h_i^2 - \sum_{i=1}^{l-1} 2h_i h_l + \sum_{i=1}^{l-1} h_l^2 = \\ &= \sum_{i=1}^{l-1} h_i^2 - 2h_l \sum_{i=1}^{l-1} h_i + (l-1)h_l^2. \end{aligned}$$

В свою очередь, суммы  $\sum_{i=1}^{l-1} h_i$  и  $\sum_{i=1}^{l-1} h_i^2$  также можно не вычислять заново, а воспользоваться уже вычисленными суммами  $\sum_{i=1}^{l-2} h_i$  и  $\sum_{i=1}^{l-2} h_i^2$  с предыдущего разбиения:  $\sum_{i=1}^{l-1} h_i = \sum_{i=1}^{l-2} h_i + h_{l-1}$  и  $\sum_{i=1}^{l-1} h_i^2 = \sum_{i=1}^{l-2} h_i^2 + h_{l-1}^2$ . Получается, что каждое обновление каждой суммы производится за  $\mathcal{O}(1)$ .

Для вычисления суммы квадратов разностей второй части каждого разбиения можно развернуть массив и применить алгоритм, который использовался для первых частей. Заметим, что в конце этого же алгоритма при добавлении всех элементов в одну часть получится сумма квадратов разностей между всеми элементами  $\sum_{i=1}^n \sum_{j=1}^{i-1} (h_i - h_j)^2$ , которая также пригодится в дальнейшем.

Для любого разбиения сумму квадратов разностей между частями можно вычислить как разницу суммы квадратов разностей между всеми элементами и между элементами первой и второй частей, так как оба элемента из каждой пары лежат либо в одной части, либо в разных:

$$\begin{aligned} \sum_{i=1}^n \sum_{j=1}^{i-1} (h_i - h_j)^2 &= \sum_{i=1}^l \sum_{j=1}^{i-1} (h_i - h_j)^2 + \sum_{i=1}^l \sum_{j=l+1}^n (h_i - h_j)^2 + \sum_{i=l+1}^n \sum_{j=i+1}^n (h_i - h_j)^2 \Rightarrow \\ \Rightarrow \sum_{i=1}^l \sum_{j=l+1}^n (h_i - h_j)^2 &= \sum_{i=1}^n \sum_{j=1}^{i-1} (h_i - h_j)^2 - \sum_{i=1}^l \sum_{j=1}^{i-1} (h_i - h_j)^2 - \sum_{i=l+1}^n \sum_{j=i+1}^n (h_i - h_j)^2. \end{aligned}$$

Получается, что для каждого разбиения сумму квадратов разностей между частями также можно вычислить за  $\mathcal{O}(1)$  из уже вычисленных сумм.

### **Пример программы-решения**

Ниже представлено решение на языке Python.

## Python

```

1  import math
2
3  def solve_in_half(n, h, res):
4      sum1 = h[0]
5      sum2 = h[0] * h[0]
6      ans = 0
7
8      for p in range(1, n):
9          res[p - 1] = ans
10         hh = h[p] * h[p]
11         ans += sum2
12         ans += hh * p
13         ans -= 2 * h[p] * sum1
14         sum1 += h[p]
15         sum2 += hh
16     return ans
17
18 def solve(n, h):
19     h = h[:]
20     l = [0] * (n - 1)
21     m = [0] * (n - 1)
22     r = [0] * (n - 1)
23
24     solve_in_half(n, h, l)
25     reverse(n, h)
26     total = solve_in_half(n, h, r)
27     reverse(n - 1, r)
28
29     for i in range(n - 1):
30         m[i] = total - l[i] - r[i]
31
32     for i in range(1, n - 2):
33         cntL = i + 1
34         cntR = n - cntL
35         cntM = cntL * cntR
36         cntL = cntL * (cntL - 1) / 2
37         cntR = cntR * (cntR - 1) / 2
38         l[i] = math.sqrt(l[i] / cntL)
39         m[i] = math.sqrt(m[i] / cntM)
40         r[i] = math.sqrt(r[i] / cntR)
41
42     return l, m, r
43
44 def reverse(n, a):
45     l, r = 0, n - 1
46     while l < r:
47         a[l], a[r] = a[r], a[l]
48         l += 1
49         r -= 1
50
51
52 n = int(input())
53 h = list(map(int, input().split()))
54
55 res = solve(n, h)
56 l, c, r = res
57
58 for i in range(1, n - 2):
59     print(l[i], c[i], r[i])

```

### Задача 3.2.3. Выборы (100 баллов)

Темы: машинное обучение, классификация, графы.

#### Условие

Скоро в округе Самая Шуточная Агломерация должны пройти выборы нового мэра. За этот пост борются два кандидата: Даниил Вичсенд и Камила Маклис. Для анализа предвыборной кампании был нанят программист Марк Ящеров, который решил выяснить, как голосуют избиратели в зависимости от их предпочтений. Он провел опрос в одной социальной сети и выяснил, как собираются голосовать люди, но не смог узнать никаких характеристик о них. Вместо этого у него есть информация о некотором коэффициенте схожести между пользователями. Но как на основе него можно построить предсказание, Марк не разобрался. Помогите ему с этим.

Набор данных состоит из двух таблиц в csv формате: `train.csv` и `sim.csv`.

- `train.csv` состоит из столбцов `id` и `class`, он содержит информацию о том, за кого избиратели собираются голосовать.
- `sim.csv` состоит из столбцов `id1`, `id2` и `sim`, он содержит информацию о схожести между измерителями с индексами `id1` и `id2`. Схожесть симметрична.

Необходимо загрузить в тестирующую систему csv-файл со столбцами `id` и `class`, в котором будут предсказания результата голосования для избирателей с `id` от 10 000 до 14 999.

#### Решение

Данная задача не имеет точного решения. Для решения можно было использовать алгоритм, который строит предсказание на основе близости относительно заданной меры схожести. Частный случай такого алгоритма используется в базовом решении.

Более продвинутый подход основывается на предварительном извлечении признаков.

#### Пример программы-решения

Ниже представлено решение на языке Python.

#### Python

```
1 import pandas as pd
2 from sklearn.model_selection import train_test_split
3 from sklearn.decomposition import PCA
4 from sklearn.linear_model import LogisticRegression
5
6 # Загрузка данных
7 train_data = pd.read_csv("train.csv")
8 sim_data = pd.read_csv("sim.csv")
9
10 # Построение полной матрицы схожести
```

```

11 similar = pd.pivot_table(sim_data, values='sim', index='id1',
    ↪ columns='id2').fillna(0)
12
13 X = similar.loc[train_data['id']].fillna(0)
14 y = train_data['class']
15 X_train, X_val, y_train, y_val = train_test_split(X, y,
    ↪ test_size=0.05, random_state=42)
16
17 # Извлечение признаков
18 pca = PCA(n_components=95, random_state=2750)
19 pca.fit_transform(X_train)
20 X_train = pca.transform(X_train)
21 X_val = pca.transform(X_val)
22
23 # Обучение модели и построение предсказаний
24 model = LogisticRegression(C=0.844, max_iter=4000)
25 model.fit(X_train, y_train)
26
27 test_id = range(10000, 15000)
28 test_data = similar.loc[test_id]
29 predictions = model.predict(pca.transform(test_data.fillna(0)))
30
31 ans = pd.DataFrame({'id': test_id, 'class': predictions})
32 ans.to_csv('predictions.csv', index=False)

```

### Критерии оценивания

Для оценки решения используется F-мера.

Баллы, которые получатся в результате, будут вычислены по формуле:

$$\max\left(0; 100 \frac{S - B}{M - B}\right),$$

где  $S$  — качество решения задачи командой,  $B$  — качество базового решения, а  $M$  — качество наилучшего среди всех команд решения.

### Задача 3.2.4. Беспорядочные категории (100 баллов)

Темы: машинное обучение, извлечение признаков.

#### Условие

Однажды Роман попал в клуб настольных игр «Куб Ведер», где надумал посетить зал карточных игр. Роман там был в первый раз и решил разобраться в правилах всех игр. Он не хотел выглядеть новичком, поэтому попытался понять правила, просто наблюдая за играми. Это оказалось не так уж и просто. Для каждой игры задействуются несколько различных колод карт. В каждом раскладе используется ровно по одной карте из каждой колоды, которые выбираются в зависимости от других уже выложенных карт, но независимо от предыдущих раскладов. Более того, это специальные фантазийные карты, на которых не указано достоинство, а просто изображено какое-то сказочное существо.

Роман подумал, что для начала стоит разобраться в том, как должны быть упорядочены карты. Он решил купить в киоске новую колоду карт, но оказалось, что

они там уложены в произвольном порядке, да и колод ему пришлось бы купить слишком много. Тогда Роман захотел собрать статистику игр. Для удобства он сопоставил каждой картинке какое-то число от одного до числа карт в соответствующей колоде.

Помогите восстановить порядок карт по истории раскладов для каждой игры.

Дано множество `csv`-файлов. Каждый из них содержит информацию о раскладах для соответствующей игры. Название каждого файла имеет формат `gid_n_m.csv`, где `gid` — индекс игры, `n` — число выписанных раскладов в этой игре, а `m` — число используемых колод и карт в соответствующем раскладе. Каждый расклад содержит для каждой колоды условный номер карты, которая была выбрана для этого расклада.

В тестирующую систему необходимо загрузить `csv`-файл со столбцами: `game`, `deck` и `order`, где `game` — индекс игры, `deck` — номер колоды, `order` — предполагаемый порядок карт.

### **Решение**

В данной задаче необходимо было упорядочить значения категорий. Для этого можно было решить задачу построения обучаемых эмбендингов. При этом эмбендинги должны быть одномерными, чтобы на основе соответствующих значений можно было упорядочить значения категорий.

Так как наборы данных не содержали информации о целевой задаче, для обучения эмбендингов необходимо было использовать самообучение. В качестве суррогатной задачи можно было использовать задачу обучения автокодировщика, который по значению эмбендингов должен предсказать их же самих. Архитектура автокодировщика должна была содержать регуляризирующий фактор, например, «бутылочное горлышко», чтобы тождественная функция, которую аппроксимирует автокодировщик, не была тривиальной. Другая важная часть автокодировщика — пакетная нормализация, которая призвана предотвратить вырождение эмбендингов, поскольку тривиальным решением будет заполнение всех эмбендингов нулями

### **Пример программы-решения**

Ниже представлено решение на языке Python.

#### **Python**

```
1 import os
2 import numpy as np
3 import pandas as pd
4 import torch
5 import torch.nn as nn
6
7 class AutoEncoder(nn.Module):
8     def __init__(self, dims):
9         super(AutoEncoder, self).__init__()
10
11         self.k = len(dims)
12         self.embeddings = []
13         self.dims = dims
```

```

14
15     for dim in dims:
16         self.embeddings.append(nn.Embedding(dim, 1))
17
18     self.embeddings = nn.ModuleList(self.embeddings)
19
20     self.input_layer = nn.Linear(self.k, 5)
21     self.relu = nn.ReLU()
22     self.output_layer = nn.Linear(5, self.k)
23     self.bn = nn.BatchNorm1d(self.k, affine=False)
24
25     def embded(self, x):
26         s = torch.zeros((x.size()[0], self.k))
27
28         for i in range(self.k):
29             s[:, i] = self.embeddings[i](x[:, i]).flatten()
30
31         return s
32
33     def forward(self, x):
34         out = self.embded(x)
35         out = self.relu(self.input_layer(out))
36         out = self.output_layer(out)
37         out = self.bn(out)
38         return out
39
40     directory = "games"
41
42     game_id = []
43     deck_id = []
44     order = []
45
46     for filename in os.listdir(directory):
47         f = os.path.join(directory, filename)
48         data = pd.read_csv(f) - 1
49         g, n, m = map(int, filename[:-4].split("_"))
50
51         t = torch.tensor(data.to_numpy())
52         dims = data.max() + 1
53         model = AutoEncoder(dims)
54
55         learning_rate = 0.1
56         criterion = nn.MSELoss()
57         optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)
58
59         for epoch in range(100):
60             optimizer.zero_grad()
61             output_train = model(t)
62             loss = criterion(output_train, model.embded(t))
63             loss.backward()
64             optimizer.step()
65
66         for d, embd in enumerate(model.embeddings):
67             game_id.append(g)
68             deck_id.append(d + 1)
69             order.append(" ".join(map(str,
70                                     ↪ np.argsort(embd.weight.detach().numpy().flatten()) + 1)))
71
72     pd.DataFrame({"game" : game_id, "deck" : deck_id, "order" :
73                 ↪ order}).to_csv("submission.csv", index=False)

```

**Критерии оценивания**

Для оценки решения используется усредненный по колодам, а затем — по играм модуль коэффициента корреляции Кенделла.

Баллы, которые получатся в результате, будут вычислены по формуле:

$$\max \left( 0; 100 \frac{S - B}{M - B} \right),$$

где  $S$  — качество решения задачи командой,  $B$  — качество базового решения, а  $M$  — качество наилучшего среди всех команд решения.

## 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. Високосный год (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**

Как известно, количество дней в году, согласно Григорианскому календарю, определяется следующим образом:

- если номер года не делится на 4, то количество дней в году равно 365;
- если номер года делится на 4, но не делится на 100, то количество дней в году равно 366;
- если номер года делится на 100, но не делится на 400, то количество дней в году равно 365;
- если номер года делится на 400, то количество дней в году равно 366.

Требуется по номеру года определить, сколько в нем минут согласно Григорианскому календарю.

#### **Формат входных данных**

Первая строка содержит единственное целое число  $y$  ( $1\,583 \leq y \leq 2\,025$ ) — номер года.

#### **Формат выходных данных**

Выведите одно целое число — количество минут в  $y$  году.

#### **Примеры**

*Пример №1*

<b>Стандартный ввод</b>
2025

**Стандартный вывод**

525600

**Решение**

Используем условный оператор для реализации логики, описанной в условии задачи. Чтобы определить, делится ли  $x$  на  $y$ , можно проверить, выполняется ли условие  $x\%y == 0$ . Здесь  $x\%y$  обозначает остаток от деления  $x$  на  $y$ , где  $\%$  — оператор, который находит остаток.

Если  $x$  делится на  $y$ , то остаток от деления  $x$  на  $y$  равен 0, поэтому приведенный выше код определяет делимость.

В нашем случае может быть несколько условных операторов, что может привести к глубокой вложенности. В таких ситуациях можно сразу вывести результат и завершить выполнение функции, как только условие выполнится, чтобы избежать глубокой вложенности.

**Пример программы-решения**

Ниже представлено решение на языке Python.

**Python**

```

1  y = int(input())
2  if y % 4:
3      print(365 * 60 * 24)
4  else:
5      if y % 100:
6          print(366 * 60 * 24)
7      else:
8          if y % 400:
9              print(365 * 60 * 24)
10         else:
11             print(366 * 60 * 24)

```

**Задача 4.2.1.2. Очень сложная математика (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**

В задаче требуется по данным целым числам  $a$ ,  $b$  и  $c$  определить значение следующего выражения:  $\sum_{i=1}^a \sum_{j=1}^b \sum_{k=1}^c i \cdot j \cdot k$ .

Иными словами, необходимо вычислить сумму произведений  $i \cdot j \cdot k$  по всем тройкам целых чисел  $(i, j, k)$  таких, что  $1 \leq i \leq a$ ,  $1 \leq j \leq b$  и  $1 \leq k \leq c$ .

### Формат входных данных

Единственная строка содержит три целых числа  $a$ ,  $b$  и  $c$  ( $1 \leq a, b, c \leq 10^9$ ).

### Формат выходных данных

Выведите одно целое число — значение искомой суммы. Так как ответ может быть достаточно большим, выведите остаток от деления суммы на число 998 244 353.

### Примеры

#### Пример №1

<b>Стандартный ввод</b>
1 2 3
<b>Стандартный вывод</b>
18

#### Пример №2

<b>Стандартный ввод</b>
1000000000 987654321 123456789
<b>Стандартный вывод</b>
951633476

### Примечания

В первом примере искомая сумма вычисляется следующим образом:

$$(1 \cdot 1 \cdot 1) + (1 \cdot 1 \cdot 2) + (1 \cdot 1 \cdot 3) + (1 \cdot 2 \cdot 1) + (1 \cdot 2 \cdot 2) + (1 \cdot 2 \cdot 3) = 1 + 2 + 3 + 2 + 4 + 6 = 18.$$

Ответ во втором примере очень большой. Убедитесь в том, что остаток от деления ответа на число 998 244 353 был взят правильно.

### Решение

Имеем:

$$\sum_{i=1}^a \sum_{j=1}^b \sum_{k=1}^c i \cdot j \cdot k = \sum_{i=1}^a i \cdot \sum_{j=1}^b j \cdot \sum_{k=1}^c k = \frac{a(a+1)}{2} \cdot \frac{b(b+1)}{2} \cdot \frac{c(c+1)}{2}.$$

### Пример программы-решения

Ниже представлено решение на языке Python.

#### Python

```
1 a, b, c = map(int, input().split())
2 sum = a * (a + 1) // 2 * b * (b + 1) // 2 * c * (c + 1) // 2
3 print(sum % 998244353)
```

### Задача 4.2.1.3. Угощение (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

На планете Метароботов праздник.

Роботы готовят очень большие угощения к празднику. Угощение  $R$ -ранга ( $R$  — натуральное число или 0):

- Угощение 0-ранга — это одна «Чистая энергия».
- Угощение  $R$ -ранга, где  $R > 0$ , — представляет собой железяку, угощение ( $R - 1$ )-ранга, «Чистую энергию», угощение ( $R - 1$ )-ранга и железяку, сложенные друг на друга.

Например, угощение 1-ранга и угощение 2-ранга выглядят следующим образом: МЕЕЕМ и ММЕЕЕМЕМЕЕЕММ, где М — железяка, а Е — «Чистая энергия».

На праздник приготовили угощение  $N$ -ранга. Один хитрый робот съел  $X$  элементов сверху. Сколько «Чистой энергии» съел робот?

#### Формат входных данных

Даны два числа  $N$  и  $X$ ,  $1 \leq N \leq 50$ ,  $1 \leq X \leq$  (максимальное число слоев в угощении  $N$ -ранга).

#### Формат выходных данных

Выведите количество съеденной «Чистой энергии».

**Примеры***Пример №1*

<b>Стандартный ввод</b>
2 7
<b>Стандартный вывод</b>
4

*Пример №2*

<b>Стандартный ввод</b>
1 1
<b>Стандартный вывод</b>
0

*Пример №3*

<b>Стандартный ввод</b>
50 4321098765432109
<b>Стандартный вывод</b>
2160549382716056

**Решение**

Пусть:

- $a_i$  — общее количество элементов (толщина) угощения ранга  $i$  ( $i \geq 0$ ),
- $p_i$  — количество элементов «Чистая энергия» (E) в угощении ранга  $i$  ( $i \geq 0$ ).

Эти величины вычисляются рекуррентно:

$$\begin{cases} a_0 = 1 & \text{(только E),} \\ a_i = 2a_{i-1} + 3 & \text{при } i \geq 1, \end{cases} \quad \begin{cases} p_0 = 1 & \text{(только E),} \\ p_i = 2p_{i-1} + 1 & \text{при } i \geq 1. \end{cases}$$

Искомое количество элементов «Чистая энергия» в верхних  $X$  слоях угощения ранга  $N$  обозначим как  $f(N, X)$ .

Угощение ранга 0 — это просто E:

$$f(0, X) = \begin{cases} 1 & \text{при } X \geq 1, \\ 0 & \text{при } X = 0. \end{cases}$$

Угощение ранга  $N$  имеет структуру:

$$M + \text{Угощение}_{N-1} + E + \text{Угощение}_{N-1} + M.$$

Разберем возможные значения  $X$ :

1.  $X = 1 : f(N, 1) = 0$  (верхний слой —  $M$ ).
2.  $1 < X \leq 1 + a_{N-1} : f(N, X) = f(N - 1, X - 1)$ .
3.  $X = 2 + a_{N-1} : f(N, X) = p_{N-1} + 1$ .
4.  $2 + a_{N-1} < X \leq 2 + 2a_{N-1} : f(N, X) = p_{N-1} + 1 + f(N - 1, X - (2 + a_{N-1}))$ .
5.  $X \geq 3 + 2a_{N-1} : f(N, X) = 2p_{N-1} + 1$ .

Если реализовать это как рекурсивную функцию, то ответ может быть найден за  $N + 1$  шагов вычислений.

### Пример программы-решения

Ниже представлено решение на языке Python.

#### Python

```

1 N, X = map(int, input().split())
2 a, p = [1], [1]
3 for i in range(N):
4     a.append(a[i] * 2 + 3)
5     p.append(p[i] * 2 + 1)
6 def f(N, X):
7     if N == 0:
8         return 0 if X <= 0 else 1
9     elif X <= 1 + a[N-1]:
10        return f(N-1, X-1)
11    else:
12        return p[N-1] + 1 + f(N-1, X-2-a[N-1])
13
14 print(f(N, X))

```

### Задача 4.2.1.4. Сумма цифр (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

Для целых чисел  $b$  ( $b \geq 2$ ) и  $n$  ( $n \geq 1$ ) определим функцию  $f(b, n)$  следующим образом:

- $f(b, n) = n$ , где  $n < b$ ;
- $f(b, n) = f(b, \text{floor}(\frac{n}{b})) + (n \bmod b)$ , где  $n \geq b$ .

$\text{floor}(n/b)$  — наибольшее целое число, не превосходящее  $n/b$ , а  $(n \bmod b)$  — остаток от деления числа  $n$  на число  $b$ .

Менее формально  $f(b, n)$  равно сумме цифр числа  $n$ , записанного в системе счисления с основанием  $b$ .

Например:

- $f(10, 87654) = 8 + 7 + 6 + 5 + 4 = 30$ ;
- $f(100, 87654) = 8 + 76 + 54 = 138$ .

Даны два числа  $n$  и  $s$ . Требуется определить, существует ли целое число  $b$  ( $b \geq 2$ ) такое, что  $f(b, n) = s$ . Если ответ существует, то найдите наименьшее такое  $b$ .

### Формат входных данных

Даны два числа  $n$  ( $1 \leq n \leq 10^{11}$ ) и  $s$  ( $1 \leq s \leq 10^{11}$ ).

### Формат выходных данных

Если существует целое число  $b$  ( $b \geq 2$ ) такое, что  $f(b, n) = s$ , выведите наименьшее такое  $b$ . Если такого числа не существует, то выведите  $(-1)$ .

### Примеры

#### Пример №1

<b>Стандартный ввод</b>
87654
30
<b>Стандартный вывод</b>
10

#### Пример №2

<b>Стандартный ввод</b>
87654
138
<b>Стандартный вывод</b>
100

### Решение

Очевидно, что в случае  $s = n$  ответом будет  $n + 1$ .

В остальных случаях сначала проводится полный перебор для нахождения целого числа  $b$  в диапазоне  $2 \leq b \leq \sqrt{n}$ , удовлетворяющего условию  $f(b, n) = s$ . Если такое  $b$  существует, то минимальное из них будет ответом.

Если такого  $b$  не найдено, то необходимо проверить наличие целого числа  $b$  в диапазоне  $\sqrt{n} < b \leq n$ , удовлетворяющего условию  $f(b, n) = s$ . Заметим, что при  $b > \sqrt{n}$  число  $n$  в системе счисления с основанием  $b$  будет двузначным. Пусть старшая цифра будет  $p$ , а младшая —  $q$  (где  $1 \leq q < b$ ,  $0 \leq q < b$ ), тогда

$$n = p \cdot b + q. \quad (4.2.1)$$

Согласно условию задачи,

$$p + q = s. \quad (4.2.2)$$

Поскольку  $b > \sqrt{n}$ , из уравнения (4.2.1) следует, что  $n = p \cdot b + q \geq p \cdot b > p^2$ , откуда получаем  $p < \sqrt{n}$ . Таким образом можно провести полный перебор по старшей цифре  $p$ . Из уравнений (4.2.1) и (4.2.2) следует, что  $b = \frac{n - s}{p} + 1$ , поэтому  $b$  однозначно определяется по  $p$ . Наконец, для такого  $b$  необходимо проверить, выполняется ли условие  $f(b, n) = s$ .

Общая сложность алгоритма составляет  $\mathcal{O}(\sqrt{n})$ , что позволяет получить максимальный балл.

### Пример программы-решения

Ниже представлено решение на языке Python.

#### Python

```

1 def sum_digit(b, n):
2     if n < b:
3         return n
4     return sum_digit(b, n // b) + n % b
5
6 N = int(input())
7 S = int(input())
8
9 if S > N:
10    print(-1)
11 elif S == N:
12    print(N + 1)
13 else:
14    sqrt = int(pow(N, 0.5))
15    for b in range(2, sqrt + 1):
16        if sum_digit(b, N) == S:
17            print(b)
18            quit()
19    for k in reversed(range(1, sqrt + 1)):
20        if (N - S) % k == 0:
21            b = (N - S) // k + 1
22            if sum_digit(b, N) == S:
23                print(b)
24                quit()
25    print(-1)

```

### **Задача 4.2.1.5. Исследование космоса (25 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:** 256 Мбайт.

#### **Условие**

Для наблюдения за космосом нужно подняться как можно выше над уровнем моря.

Территория для наблюдения представляет собой полосу шириной  $w$ , разбитую на единичные квадраты. В  $i$ -м из них находится столбец земли высотой  $h_i$ . Можно добавить не более  $n$  клеток с песком, чтобы поднять территорию как можно выше. Однако песок нестабилен, и клетку с песком можно разместить только при условии, что клетка под ней, а также клетки по диагонали слева снизу и справа снизу заполнены землей или песком.

Определите максимальную высоту над уровнем моря, которую можно создать для наблюдения за космосом.

#### **Формат входных данных**

Первая строка содержит два целых числа  $w$  ( $1 \leq w \leq 100\,000$ ) — ширина территории и  $n$  ( $0 \leq n \leq 10^9$ ) — максимальное количество добавляемых клеток песка.

Следующие  $w$  строк содержат одно целое число  $h_i$  — высоту  $i$ -го столбца земли ( $1 \leq h_i \leq 10^9$ ).

#### **Формат выходных данных**

Определите максимальную высоту над уровнем моря, которую можно создать для наблюдения за космосом.

#### **Примеры**

##### *Пример №1*

<b>Стандартный ввод</b>	
8	5
3	
3	
3	
2	
3	
1	
2	
4	

<b>Стандартный вывод</b>
5

Пример №2

<b>Стандартный ввод</b>
3 10 1 2 1
<b>Стандартный вывод</b>
2

**Решение**

Найдем ответ бинарным поиском по ответу следующим образом. Предположим, что сможем получить высоту  $h$ . Тогда посчитаем, сколько понадобится песка, чтобы ее построить. Если есть необходимое количество, такая высота горы устраивает.

Чтобы по данному  $h$  определить необходимое количество песка, переберем всевозможные  $i$ , где может находиться вершина горы. Чтобы считать количество песка быстрее, заведем три вспомогательных массива: массив префиксных сумм  $psum$  (для быстрого подсчета суммы высот каменных столбов), массив  $left$  и массив  $right$ . Два последних массива понадобятся пересчитывать для каждого  $h$ .

Массив  $left$  — это такой массив, что  $left[i]$  — координата левой границы насыпи из песка, если вершина насыпи высоты  $h$  находится на координате  $i$ . Будем строить его следующим образом: для каждой координаты  $i$  посчитаем, для какой координаты  $sfrom$  она может быть левой границей насыпи (чтобы  $left[sfrom] = i$ ). Если будем перебирать  $i$  слева, то для каждой координаты  $sfrom$  значение  $left[sfrom]$  будет максимальным, что и требуется. Аналогично построим массив  $right$  — координаты правых границ насыпей.

После этого остается перебрать всевозможные координаты вершины и для каждой посчитать размер горы высоты  $h$  (учитывая камни). Это можно сделать по формуле, так как известно, где каждая гора начинается (по массиву  $left$ ) и заканчивается (по массиву  $right$ ). Затем остается вычесть количество камней в полученной горе (с помощью  $psum$ ) и найти минимум среди полученных значений.

**Пример программы-решения**

Ниже представлено решение на языке Python.

```

Python
1 import sys
2
3 inf = int(2e18)
4 N = 123456
5

```

```

6  h = [0] * N
7  ev = [[] for _ in range(N)]
8  ans = [0] * N
9  sum = [0] * N
10
11 def main():
12     n, m = map(int, sys.stdin.readline().split())
13     for i in range(n):
14         h[i] = int(sys.stdin.readline().strip())
15
16     low = 0
17     for i in range(n):
18         low = max(low, h[i])
19
20     high = int(1.01e9)
21     while low < high:
22         mid = (low + high + 1) // 2
23         for i in range(n):
24             ans[i] = 0
25
26         for rot in range(2):
27             sum[0] = h[0]
28             for i in range(1, n):
29                 sum[i] = sum[i - 1] + h[i]
30
31             for i in range(n):
32                 ev[i].clear()
33
34             for i in range(n):
35                 j = i + mid - h[i]
36                 if j < n:
37                     ev[j].append(i)
38
39             mx = -1
40             for i in range(n):
41                 for j in range(len(ev[i])):
42                     mx = max(mx, ev[i][j])
43                 if mx >= 0:
44                     from_val = mid - 1
45                     to = mid - (i - mx) + 1
46                     ans[i] += (to + from_val) * (from_val - to + 1)
47                     ↪ // 2
48                     ans[i] -= (sum[i - 1] - sum[mx])
49                 else:
50                     ans[i] += inf
51
52             for i in range(n // 2):
53                 h[i], h[n - i - 1] = h[n - i - 1], h[i]
54                 ans[i], ans[n - i - 1] = ans[n - i - 1], ans[i]
55
56     found = False
57     for i in range(n):
58         ans[i] += mid - h[i]
59         if ans[i] <= m:
60             found = True
61             break
62
63     if found:
64         low = mid
65     else:

```

```
65         high = mid - 1
66
67     print(low)
68
69     if __name__ == "__main__":
70         main()
```

## 4.2.2. Математика. 8–9 классы

### Задача 4.2.2.1. (15 баллов)

Темы: оценка плюс пример, четность и нечетность.

#### Условие

В левом верхнем углу квадратного клетчатого поля  $4 \times 4$  находится фишка. За один ход фишку можно перемещать по полю на одну клетку вправо, влево, вверх или вниз. При этом каждую клетку можно посетить не более одного раза. Фишку перемещают в правый нижний угол и подсчитывают сумму всех чисел в клетках, где побывала фишка. Какую наибольшую сумму можно получить?

2	4	3	2
3	3	1	1
1	2	4	3
4	1	2	4

#### Решение

2	4	3	2
3	3	1	1
1	2	4	3
4	1	2	4

Рис. 4.2.1

Заметим, что сумма всех чисел на поле равна

$$(1 + 2 + 3 + 4) \cdot 4 = 40.$$

Однако при соблюдении правил фишка не может побывать во всех клетках. Действительно, пусть фишку переместили  $r$  раз вправо,  $l$  раз влево,  $u$  раз вверх и  $d$  раз вниз. Тогда

$$r - l = 3, \quad d - u = 3.$$

Следовательно, количество клеток, в которых побывала фишка, равно

$$1 + r + l + d + u = 1 + (l + 3) + l + (u + 3) + u = 2(l + u) + 7,$$

то есть нечетному числу.

Значит, хотя бы одна клетка останется непосещенной. Поскольку наименьшее число на поле равно 1, то наибольшая сумма не превзойдет  $40 - 1 = 39$ .

Сумму 39 можно набрать, проведя фишку, как показано на рис. 4.2.1.

**Ответ:** 39.

### **Задача 4.2.2.2. (20 баллов)**

*Темы: проценты, система линейных уравнений, текстовая задача.*

#### **Условие**

Чтобы лучше подготовиться к ОГЭ, Андрей смотрит обучающий видеоролик. В начале ролик был достаточно простой, поэтому Андрей увеличил скорость просмотра в 1,5 раза. Начиная с некоторого места начался разбор трудных задач. Тогда Андрей установил скорость, в 1,25 раза большую оригинальной, и так досмотрел ролик до конца. Известно, что продолжительность ролика и время, затраченное на его просмотр, относятся как 7 : 5. Сколько процентов от всего времени просмотра затратил Андрей на просмотр первых 75% ролика?

#### **Решение**

Пусть единица измерения времени такова, что на все видео приходится 7 единиц. Обозначим через  $t_1$  время просмотра ролика на скорости 1,5х, а через  $t_2$  — на скорости 1,25х. Тогда по условию задачи имеем систему

$$\begin{cases} t_1 + t_2 = 7, \\ 1,5t_1 + 1,25t_2 = 7. \end{cases}$$

Решая ее, находим

$$t_1 = 3, \quad t_2 = 2.$$

На скорости 1,5х школьник посмотрел  $1,5t_1 = 9/2$  единиц времени, то есть  $\frac{9/2}{7} = \frac{9}{14}$  от всего ролика. Так как  $\frac{9}{14} < \frac{3}{4}$ , то школьник поменял скорость просмотра раньше, чем просмотрел 75% ролика. Значит, оставшиеся 25% были просмотрены на скорости 1,25х. Таким образом, на оставшуюся четверть ролика было потрачено

$$\frac{\frac{1}{4} \cdot 7}{1,25} = \frac{7}{5}$$

единиц времени, то есть  $100 \cdot \frac{7/5}{7} = 28\%$  всего времени просмотра. Тогда на первые 75% приходится  $100 - 28 = 72\%$  всего времени просмотра.

**Ответ:** 72%.

### **Задача 4.2.2.3. (20 баллов)**

*Темы: признаки делимости, комбинаторика.*

**Условие**

Сколько существует различных пятизначных чисел, обладающих следующими свойствами:

- последние четыре цифры различны;
- число делится на 4.

**Решение**

На первое место можно поставить любую цифру, кроме 0 — всего 9 способов. На второе место можно поставить любую цифру, кроме тех, которые будут находиться на последних двух позициях — 8 способов. На третьем — так же, за исключением еще той, которая стоит на втором месте — 7 способов.

Число делится на 4, если делится на 4 число, образованное его двумя последними цифрами. Среди чисел  $0, 1, \dots, 99$  имеется 25 делящихся на 4. Необходимо исключить 0, 44 и 88, поскольку в их записи одинаковые цифры. Таким образом, общее количество таких чисел равно

$$9 \cdot 8 \cdot 7 \cdot (25 - 3) = 11\,088.$$

**Ответ:** 11 088.

**Задача 4.2.2.4. (20 баллов)**

Темы: высота, биссектриса, медиана, площадь.

**Условие**

В треугольнике  $ABC$  с острым углом  $B$  проведена высота  $BD$ ,  $DE$  — биссектриса треугольника  $BDA$ , а  $EF$  — медиана треугольника  $DEA$ . Найти площадь четырехугольника  $FEBD$ , если  $AB = 5$ ,  $BC = 8$ ,  $\sin \angle ABC = \frac{3}{5}$ .

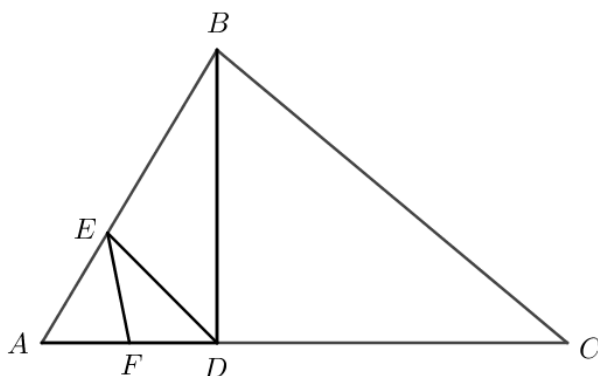
**Решение**

Рис. 4.2.2

Площадь  $\triangle ABC$

$$S_{ABC} = \frac{1}{2} AB \cdot BC \cdot \sin B = \frac{1}{2} \cdot 5 \cdot 8 \cdot \frac{3}{5} = 12.$$

Так как  $\sin^2 B + \cos^2 B = 1$  и угол  $B$  по условию острый, то  $\cos B = \frac{4}{5}$ . По теореме косинусов

$$AC^2 = AB^2 + BC^2 - 2 \cdot AB \cdot BC \cos B = 25 + 64 - 2 \cdot 5 \cdot 8 \cdot \frac{4}{5} = 25.$$

То есть  $AC = 5$ .

Так как  $S_{ABC} = \frac{1}{2} AC \cdot BD$  и  $S_{ABC} = 12$ , то  $BD = \frac{2 \cdot 12}{5} = \frac{24}{5}$ . По теореме Пифагора

$$AD^2 = AB^2 - BD^2 = 25 - \frac{24^2}{25} = \frac{49}{25}.$$

Тогда  $AD = \frac{7}{5}$ . Далее

$$S_{ABD} = \frac{1}{2} BD \cdot AD = \frac{1}{2} \cdot \frac{24}{5} \cdot \frac{7}{5} = \frac{84}{25}.$$

Рассмотрим  $\triangle ABD$ . Так как  $DE$  — биссектриса, то

$$\frac{AE}{EB} = \frac{AD}{BD} = \frac{7}{5} \cdot \frac{5}{24} = \frac{7}{24}.$$

Треугольники  $AED$  и  $ABD$  имеют общую высоту, проведенную из точки  $D$ . Следовательно,

$$S_{AED} = \frac{AE}{AB} S_{ABD} = \frac{7}{31} \cdot \frac{84}{25}.$$

Так как  $EF$  — медиана в  $\triangle AED$ , то

$$S_{AEF} = \frac{S_{AED}}{2} = \frac{7}{31} \cdot \frac{42}{25}.$$

Искомая площадь

$$S_{FEED} = S_{ABD} - S_{AEF} = \frac{84}{25} - \frac{7 \cdot 42}{31 \cdot 25} = \frac{462}{155}.$$

**Ответ:**  $\frac{462}{155}$ .

### **Задача 4.2.2.5. (25 баллов)**

Темы: неравенство с модулем, неравенство с квадратным трехчленом, арифметическая прогрессия.

**Условие**

Определите, сколько целых решений имеет неравенство

$$\sum_{k=1}^{2025} \left( |x - k| - \frac{1012}{2025} k \right) \leq 1\,000\,000.$$

**Решение**

Преобразуем левую часть неравенства:

$$\sum_{k=1}^{2025} \left( |x - k| - \frac{1012}{2025}k \right) = \sum_{k=1}^{2025} |x - k| - \frac{1012}{2025} \sum_{k=1}^{2025} k.$$

Сумма арифметической прогрессии

$$\sum_{k=1}^{2025} k = \frac{1 + 2025}{2} \cdot 2025 = 1013 \cdot 2025.$$

Тогда

$$\frac{1012}{2025} \sum_{k=1}^{2025} k = 1012 \cdot 1013.$$

Значит, исходное неравенство равносильно

$$\sum_{k=1}^{2025} |x - k| \leq 1012 \cdot 1013 + 10^6.$$

Если  $x \leq 0$  или  $x \geq 2026$ , то

$$\sum_{k=1}^{2025} |x - k| \geq \sum_{k=1}^{2025} k = \frac{1 + 2025}{2} \cdot 2025 = 1013 \cdot 2025.$$

Имеем

$$1013 \cdot 2025 = 1013 \cdot (1012 + 1013) = 1013 \cdot 1012 + 1013^2 > 1013 \cdot 1012 + 10^6.$$

Поэтому числа  $x \leq 0$  и  $x \geq 2026$  не являются решениями.

Рассмотрим сумму  $S = \sum_{k=1}^{2025} |x - k|$ , когда  $x$  — целое число от 1 до 2025. Имеем:

$$x = 1: \quad S = 0 + 1 + 2 + 3 + \dots + 2024,$$

$$x = 2: \quad S = 1 + 0 + 1 + 2 + \dots + 2023,$$

$$x = 3: \quad S = 2 + 1 + 0 + 1 + \dots + 2022,$$

...

$$x = 2025: \quad S = 2024 + 2023 + 2022 + \dots + 0.$$

То есть число  $S$  — сумма двух арифметических прогрессий,

$$S = \frac{(x-1)+0}{2} \cdot x + \frac{1+(2025-x)}{2} \cdot (2025-x) = x^2 - 2026x + 1013 \cdot 2025.$$

Таким образом, требуется найти количество целых решений неравенства

$$x^2 - 2026x + 1013 \cdot 2025 \leq 1012 \cdot 1013 + 10^6,$$

которое равносильно

$$x^2 - 2026x + 1013^2 - 10^6 \leq 0.$$

Дискриминант, деленный на 4, трехчлена в левой части равен

$$\frac{D}{4} = 1013^2 - (1013^2 - 10^6) = 10^6.$$

Поэтому решения неравенства лежат в промежутке  $[1013 - 10^3, 1013 + 10^3]$ . На этом отрезке находится 2001 целое число.

**Ответ:** 2001.

### 4.2.3. Математика. 10–11 классы

#### Задача 4.2.3.1. (15 баллов)

Темы: тригонометрическое уравнение, показательное уравнение, экстремум функции.

##### Условие

Найдите все решения уравнения

$$4^{\sin^2 x} + 4^{\cos^2 x} + \operatorname{tg} x + \operatorname{ctg} x = 6.$$

на интервале  $\left(0, \frac{\pi}{2}\right)$ .

##### Решение

Рассмотрим функцию

$$f(x) = 4^{\sin^2 x} + 4^{\cos^2 x} = 4^{\sin^2 x} + \frac{4}{4^{\sin^2 x}} = t + \frac{4}{t},$$

где  $t = 4^{\sin^2 x}$ . Функция  $f$  достигает наименьшего значения при  $t = 2$ , то есть при  $x = \frac{\pi}{4}$ :  $f\left(\frac{\pi}{4}\right) = 4$ . Во всех остальных точках интервала  $\left(0, \frac{\pi}{2}\right)$  ее значения строго больше 4.

Рассмотрим функцию

$$g(x) = \operatorname{tg} x + \operatorname{ctg} x = \operatorname{tg} x + \frac{1}{\operatorname{tg} x} = u + \frac{1}{u},$$

где  $u = \operatorname{tg} x$ . Функция  $g$  достигает наименьшего значения при  $u = 1$ , то есть при  $x = \frac{\pi}{4}$ :  $g\left(\frac{\pi}{4}\right) = 2$ . Во всех остальных точках интервала  $\left(0, \frac{\pi}{2}\right)$  ее значения строго больше 2.

Таким образом, левая часть уравнения достигает значения 6 при  $x = \frac{\pi}{4}$ , а во всех остальных точках интервала  $\left(0, \frac{\pi}{2}\right)$  ее значения строго больше 6. Следовательно, имеется единственный корень  $x = \frac{\pi}{4}$ .

**Ответ:**  $\frac{\pi}{4}$ .

**Задача 4.2.3.2. (20 баллов)**

Темы: комбинаторика, сочетания с повторениями.

**Условие**

20 одинаковых роз надо расставить по 5 различным вазам. Сколькими способами это можно сделать, если могут оставаться пустые вазы?

**Решение**

Неважно, в каком порядке будем заполнять вазы цветами, важен только окончательный результат: сколько цветов будет в каждой из различных ваз. Поэтому речь идет о выборке без учета порядка, то есть о сочетаниях. Далее: цветы не выбираем, они одинаковые. Беря розу, выбираем вазу, в которую поставим эту розу. Одну и ту же вазу можно выбрать несколько раз, поэтому речь идет о схеме с повторениями. Таким образом, имеем сочетания с повторениями. Формула для числа сочетаний с повторениями имеет вид

$$\overline{C}_n^m = C_{n+m-1}^m = \frac{(n+m-1)!}{m!(n-1)!},$$

где  $n$  — число различных объектов, а  $k$  — число их выбора. В нашем случае  $n = 5$ ,  $k = 20$ , таким образом, будем иметь

$$\overline{C}_5^{20} = C_{20+5-1}^{20} = \frac{24!}{20!4!} = \frac{24 \cdot 23 \cdot 22 \cdot 21}{4 \cdot 3 \cdot 2 \cdot 1} = 10\,626.$$

**Ответ:** 10 626.

**Задача 4.2.3.3. (20 баллов)**

Темы: система неравенств, площадь сектора, уравнения прямой и окружности.

**Условие**

Найдите площадь фигуры, заданной системой неравенств

$$\begin{cases} x^2 + y^2 \leq 2(y - x) + 34, \\ x + \sqrt{3}|y - 1| \leq -1. \end{cases}$$

**Решение**

Преобразуем систему следующим образом:

$$\begin{cases} (x + 1)^2 + (y - 1)^2 \leq 36, \\ x + 1 + \sqrt{3}|y - 1| \leq 0. \end{cases}$$

Введем замену  $\tilde{x} = x + 1$ ,  $\tilde{y} = y - 1$ . Тогда система примет вид:

$$\begin{cases} \tilde{x}^2 + \tilde{y}^2 \leq 36, \\ \tilde{x} + \sqrt{3}|\tilde{y}| \leq 0. \end{cases}$$

Система задает сектор круга радиуса  $R = 6$ , заключенный между прямыми  $\tilde{y} = \frac{1}{\sqrt{3}}\tilde{x}$  и  $\tilde{y} = -\frac{1}{\sqrt{3}}\tilde{x}$  (при  $\tilde{y} \geq \frac{1}{\sqrt{3}}\tilde{x}$  и  $\tilde{y} \leq -\frac{1}{\sqrt{3}}\tilde{x}$ ).

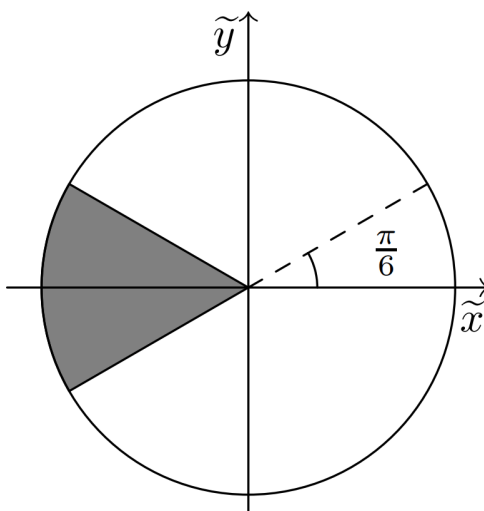


Рис. 4.2.3

Угол сектора круга при этом равен  $\alpha = 2 \cdot \frac{\pi}{6} = \frac{\pi}{3}$  радиан (прямая  $\tilde{y} = \frac{1}{\sqrt{3}}\tilde{x}$  проходит под углом  $\operatorname{arctg} \frac{1}{\sqrt{3}} = \frac{\pi}{6}$  к оси  $\tilde{x}$ ), а значит, площадь кругового сектора

$$S = \frac{1}{2}R^2 \cdot \alpha = \frac{1}{2} \cdot 6^2 \cdot \frac{\pi}{3} = 6\pi.$$

**Ответ:**  $6\pi$ .

#### **Задача 4.2.3.4. (20 баллов)**

*Темы: стереометрия, площадь проекции, построение сечения.*

##### **Условие**

В правильной четырехугольной призме  $ABCD A_1 B_1 C_1 D_1$  боковые ребра равны 8, а стороны основания — 4. Точка  $M$  делит ребро  $DD_1$  в отношении 3:5, считая от вершины  $D$ . Найдите площадь сечения призмы плоскостью, проходящей через точку  $M$  перпендикулярно прямой  $DB_1$ .

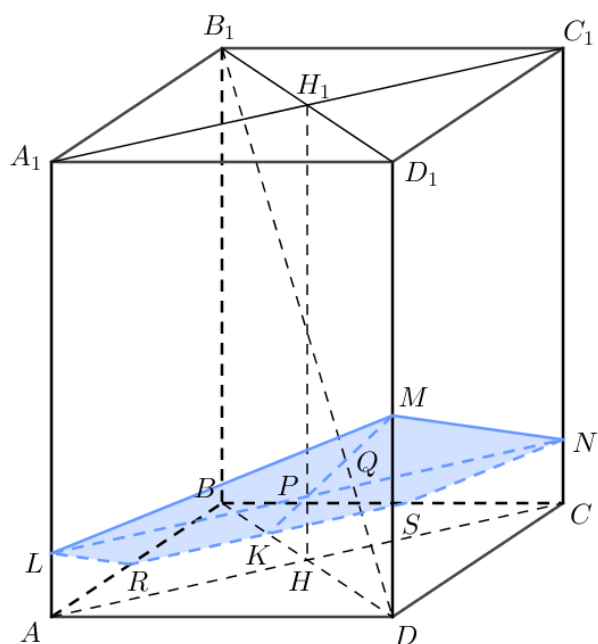
**Решение**

Рис. 4.2.4

**Построение сечения:**

1. В плоскости  $BB_1D_1$ :  $MK \perp DB_1$ ,  $K = MK \cap BD$ ,  $P = MK \cap HH_1$ , где  $H$  и  $H_1$  — центры оснований призмы.
2. В плоскости  $ABC$ :  $RS \parallel AC$ ,  $K \in RS$ ,  $R = RS \cap AB$ ,  $S = RS \cap BC$ .
3. В плоскости  $AA_1C_1$ :  $LN \parallel AC$ ,  $P \in LN$ ,  $L = LN \cap AA_1$ ,  $N = LN \cap CC_1$ .
4.  $RLMNS$  — искомое сечение:  $MK \subset RLMNS$ ,  $MK \perp DB_1$  по построению,  $LN \subset RLMNS$ ,  $LN \parallel AC$ ,  $AC \perp DB_1$  по теореме о трех перпендикулярах.

**Вычисление площади:**

1. Найдем косинус угла между построенной плоскостью и плоскостью основания. Имеем:  $RS$  — ребро двугранного угла,  $BD \perp AC$  (диагонали квадрата),  $RS \parallel AC$ , тогда  $BD \perp RS$ , а  $MK \perp RS$  (теорема о трех перпендикулярах). Следовательно,  $\angle(RLM, ABC) = \angle(MK, BD) = \angle MKD = \alpha$ .

Треугольник  $KMD$  — прямоугольный,  $DQ$  — его высота.

Значит,  $\triangle KMD \sim \triangle DMQ$  и  $\alpha = \angle MKD = \angle QDM$ . Далее

$$B_1D = \sqrt{8^2 + (4\sqrt{2})^2} = 4\sqrt{6},$$

$$\cos \alpha = \frac{DD_1}{B_1D} = \frac{8}{4\sqrt{6}} = \frac{2}{\sqrt{6}},$$

$$\operatorname{tg} \alpha = \frac{1}{\sqrt{2}}.$$

2.  $RSCDA$  — ортогональная проекция сечения  $RLMNS$  на плоскость основа-

ния. Имеем

$$\begin{aligned}BD &= AC = 4\sqrt{2}, \\KD &= \frac{MD}{\operatorname{tg} \alpha} = 3\sqrt{2}, \\KH &= KD - DH = (3 - 2)\sqrt{2} = \sqrt{2}.\end{aligned}$$

Имеем

$$\frac{RS}{AC} = \frac{BK}{HB} = \frac{2\sqrt{2} - \sqrt{2}}{2\sqrt{2}} = \frac{1}{2}.$$

Тогда  $S_{RBS} = \frac{1}{4}S_{ABC} = 2$ . Отсюда

$$S_{RSCDA} = S_{ABCD} - S_{RBS} = 16 - 2 = 14.$$

Окончательно находим

$$S_{RLMNS} = \frac{S_{RSCDA}}{\cos \alpha} = \frac{14}{2/\sqrt{6}} = 7\sqrt{6}.$$

**Ответ:**  $7\sqrt{6}$ .

### **Задача 4.2.3.5. (20 баллов)**

*Темы: сумма степеней, разложение на дроби, перестановка слагаемых.*

#### **Условие**

Вычислите

$$\sum_{n=1}^{2025} \sqrt{\left(\sum_{j=1}^n \sum_{i=j}^n i^2\right)^{-1}}.$$

#### **Решение**

Рассмотрим выражение в скобках:

$$\begin{aligned}\sum_{j=1}^n \sum_{i=j}^n i^2 &= \sum_{j=1}^n (j^2 + (j+1)^2 + \dots + n^2) = \\&= 1^2 + 2^2 + 3^2 + \dots + n^2 + \\&\quad 2^2 + 3^2 + \dots + n^2 + \\&\quad 3^2 + \dots + n^2 + \\&\quad \dots \\&\quad + n^2 = \\&= 1 \cdot 1^2 + 2 \cdot 2^2 + 3 \cdot 3^2 + \dots + n \cdot n^2 = \sum_{k=1}^n k^3.\end{aligned}$$

Воспользуемся формулой для суммы кубов:

$$\sum_{k=1}^n k^3 = \frac{n^2(n+1)^2}{4}.$$

Тогда

$$\begin{aligned} \sum_{n=1}^{2025} \sqrt{\left(\sum_{j=1}^n \sum_{i=j}^n i^2\right)^{-1}} &= \sum_{n=1}^{2025} \sqrt{\frac{4}{n^2(n+1)^2}} = \sum_{n=1}^{2025} \frac{2}{n(n+1)} = \\ &= 2 \sum_{n=1}^{2025} \left(\frac{1}{n} - \frac{1}{n+1}\right) = 2 \left(1 - \frac{1}{2} + \frac{1}{2} - \frac{1}{3} + \dots + \frac{1}{2025} - \frac{1}{2026}\right) = \\ &= 2 \left(1 - \frac{1}{2026}\right) = \frac{2025}{1013}. \end{aligned}$$

**Ответ:**  $\frac{2025}{1013}$ .

## 4.3. Инженерный тур

### 4.3.1. Общая информация

Участникам предстоит разработать алгоритм оценки популярности публикаций в социальной сети.

Социальные сети ежедневно генерируют огромные объемы контента, но определить, какие публикации становятся популярными и почему — сложная аналитическая задача. Для ее решения команды должны создать алгоритм оценки популярности публикаций, анализируя их взаимодействие с пользователями и выявляя ключевые факторы, влияющие на распространение информации.

### 4.3.2. Легенда задачи

При просмотре публикаций разные пользователи по-разному взаимодействуют с ними: кто-то активно делится, кто-то комментирует, кто-то просто ставит реакцию. Предпочитаемые взаимодействия могут изменяться в зависимости от того, какое устройство у пользователя и в какое время он просматривает публикацию. Поэтому требуется предсказывать сразу несколько характеристик взаимодействия пользователей с публикациями.

### 4.3.3. Требования к команде и компетенциям участников

Число участников в команде: 3 человека.

**Компетенции**, которыми должны обладать члены команды:

- **Аналитик** придумывает идеи решения.
- **Программист** реализует идеи в виде программы.
- **Тестировщик** тестирует программы и настраивает гиперпараметры.

### 4.3.4. Оборудование и программное обеспечение

Таблица 4.3.1

Наименование	Описание
All Cups <a href="https://cups.online/">https://cups.online/</a>	Платформа для проведения соревнований по машинному обучению.

Таблица 4.3.1

Наименование	Описание
Yandex DataSphere <a href="https://datasphere.yandex.cloud/">https://datasphere.yandex.cloud/</a>	Сервис для ML-разработки. Используется для анализа данных и обучения моделей.
Docker	ПО для виртуализации. Используется для отправки решения в тестирующую систему.

### 4.3.5. Описание задачи

При просмотре публикаций разные пользователи по-разному взаимодействуют с ними: кто-то активно делится, кто-то комментирует, кто-то просто ставит реакцию. Предпочитаемые взаимодействия могут изменяться в зависимости от того, какое устройство у пользователя и в какое время он просматривает публикацию. Поэтому требуется предсказывать сразу несколько характеристик взаимодействия пользователей с публикациями.

#### *Набор данных*

Каждая строка в наборе данных кодирует какую-то одну публикацию. Набор данных состоит из следующих столбцов:

- `view` — число просмотров публикации;
- `like` — число отметок «мне нравится»;
- `comment` — число комментариев;
- `hide` — число скрытий публикации;
- `expand` — число раскрытий публикации;
- `open_photo` — число открытий фотографии;
- `open` — число открытий публикации;
- `share_to_message` — число пересылок публикации;
- `text` — текст публикации;
- `photo` — прикрепленная к публикации фотография в кодировке `base64`.

Пример содержит способ преобразования изображения.

Тренировочный набор данных, который находится в файле `train.csv`, включает все эти характеристики.

При запуске решения в контейнере тестовый набор данных будет доступен в файловой системе по пути `/tmp/data/test.csv`. Тестовый набор данных состоит из колонок: `view`, `text` и `photo`. Файл `sample_test.csv` содержит пример тестового набора данных с идентичной структурой.

В результате работы в контейнере решение должно создать CSV-файл `result` в папке `/var/log/`. Он должен содержать колонки: `like`, `comment`, `hide`,

`expand`, `open_photo`, `open` и `share_to_message` с оценками соответствующих характеристик.

### 4.3.6. Система оценивания

Максимальный балл — 100.

Для оценки решения используется умноженный на 100 коэффициент детерминации, усредненный по всем предсказанным характеристикам. Во время соревнования доступна оценка решения только на 30% тестового набора данных, а после — на оставшихся 70%. Для тестирования на заключительном этапе участникам следует выбрать три решения.

#### *Формат входных данных*

При запуске решения в контейнере тестовый набор данных будет доступен в файловой системе по пути `/tmp/data/test.csv`. Обратите внимание, что в примере он сначала копируется в рабочую папку, перед запуском решения.

Тестовый набор данных состоит из колонок: `view`, `text` и `photo`. Файл `sample_test.csv` содержит пример тестового набора данных с идентичной структурой.

#### *Формат выходных данных*

В результате работы в контейнере решение должно создать CSV-файл `result` в папке `/var/log/`. Он должен содержать колонки: `like`, `comment`, `hide`, `expand`, `open_photo`, `open` и `share_to_message` с оценками соответствующих характеристик.

### 4.3.7. Решение задачи

Для решения необходимо выделить признаки из текста и изображений. Для выделения признаков из текста можно воспользоваться `CountVectorizer`, а для изображений — посчитать различные статистики по пикселям, либо воспользоваться какой-нибудь предобученной сверточной сетью.

После извлечения признаков следует определить для каждой целевой характеристики, какой алгоритм восстановления регрессии будет лучше ее предсказывать. Для некоторых характеристик эффективнее всего работает наивный алгоритм регрессии.

Ниже приведен пример авторского решения.

Файл для `fit.py` обучения модели.

#### Python

```
1 import pandas as pd
2 import numpy as np
3 import base64
```

```

4 from io import BytesIO
5 from PIL import Image
6 from sklearn.feature_extraction.text import CountVectorizer
7 from sklearn.dummy import DummyRegressor
8 from sklearn.ensemble import GradientBoostingRegressor
9 import pickle
10 text_vectorizer = CountVectorizer(max_features=777)
11 def img_vectorizer(photo_base64):
12     img =
13     ↪ np.array(Image.open(BytesIO(base64.b64decode(photo_base64))))
14
15     s = img.shape
16     if (len(s) == 2):
17         img = np.repeat(img, 3)
18
19     h, w = s[0], s[1]
20     img.resize((h * w, 3))
21
22     stats = []
23     stats.append(np.array([h,w]))
24     stats.append(img.min(axis=0))
25     stats.append(img.max(axis=0))
26     stats.append(img.mean(axis=0))
27     stats.append(img.std(axis=0))
28     stats.append(np.median(img, axis=0))
29     cm = np.corrcoef(img.T)
30     stats.append(cm[np.triu_indices(len(cm), k = 1)])
31     return np.concatenate(stats)
32 train = pd.read_csv("train.csv")
33 X_train_img = np.vstack(train['photo'].map(img_vectorizer))
34 X_train_text =
35     ↪ text_vectorizer.fit_transform(train["text"].fillna("")).toarray()
36 X_train = np.hstack([X_train_img, X_train_text])
37 model = {
38     'like': GradientBoostingRegressor(n_estimators=77),
39     'comment': DummyRegressor(),
40     'hide': DummyRegressor(),
41     'expand': GradientBoostingRegressor(n_estimators=77),
42     'open_photo': GradientBoostingRegressor(n_estimators=77),
43     'open': GradientBoostingRegressor(n_estimators=77),
44     'share_to_message': GradientBoostingRegressor(n_estimators=77)
45 }
46 for target, reg in model.items():
47     y_train = train[target] / train['view']
48     reg.fit(X_train, y_train)
49 with open('model.pickle', 'wb') as f:
50     pickle.dump(text_vectorizer, f)
51     pickle.dump(model, f)

```

Файл predict.py для вывода модели.

## Python

```

1 with open('text_vectorizer.pickle', 'rb') as f:
2     text_vectorizer = pickle.load(f)
3     model = pickle.load(f)
4 test = pd.read_csv("test.csv")
5 X_test_img = np.vstack(test['photo'].map(img_vectorizer))
6 X_test_text =
7     ↪ text_vectorizer.fit_transform(test["text"].fillna("")).toarray()

```

```
7 X_test = np.hstack([X_test_img, X_test_text])
8 prediction = pd.DataFrame()
9 for target, reg in model.items():
10     prediction[target] = reg.predict(X_test) * test['view']
11 prediction.to_csv("submission.csv", index=False)
```

Файл requirements.txt с используемыми версиями библиотек.

#### text

```
1 pandas==2.2.3
2 scikit-learn==1.6.1
3 pillow==10.4.0
4 Dockerfile:
5 FROM python:3
6 WORKDIR /bdl
7 COPY run.sh .
8 COPY requirements.txt .
9 RUN pip install -r requirements.txt
10 COPY model.pickle .
11 COPY predict.py .
12 CMD ["bash", "run.sh"]
```

Материалы доступны по ссылке: <https://disk.yandex.ru/d/ng3EGZLMueC79Q>.

### 4.3.8. Материалы для подготовки

1. Учебник по машинному обучению: <https://education.yandex.ru/handbook/ml>.
2. Документация к Python библиотеке для машинного обучения: <https://scikit-learn.ru/>.
3. Документация к Python библиотеке для анализа данных: <http://pandas.greekwriter.ru/>.

## 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 100,0 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 205,0 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 9 победителей и 21 призер (первые 25% участников рейтинга становятся победителями или призерами, из них первые 8% становятся победителями, оставшиеся — призерами).

### ***Критерий определения победителей и призеров (независимо от уровня)***

<b>Категория</b>	<b>Количество баллов</b>
Победители	66,00 и выше
Призеры	От 45,84 до 65,79

## 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 (Р2Р). 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».