МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И

МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

**«МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»**

**(МТУСИ)**

Кафедра Математическая кибернетика и информационные технологии

Курсовая работа

по дисциплине «Структуры и алгоритмы обработки данных»

Выполнила: студентка группы БФИ2001

Щеголькова Д.Г.

Вариант №6

Преподаватель: Симонов С.Е.

Москва, 2022

Содержание

Задание3

Задача 13

Задача 25

Задача 37

Задача 49

Задача 511

Задача 615

Задача 717

Задача 818

Задача 921

Задача 1024

Вывод26

Приложения27

**Задание**: решить 10 задач согласно варианту.

**Ход выполнения:**

**Задача 1**

Скоро состоится большой футбольный чемпионат! 𝑛 команд будут участвовать в нем, и каждая пара различных команд сыграет ровно один матч друг с другом.

У матча может быть два различных исхода: матч может закончиться вничью, тогда обе команды получат по 1 очку; одна из команд может победить, тогда победившая команда получит 3 очка, а проигравшая — 0.

Счет команды — суммарное количество очков, полученное командой за все матчи, в которых она играла.

Вас интересует гипотетическая ситуация, в которой все команды будут в конце чемпионата иметь одинаковый счет. Простой пример такой ситуации — если все матчи закончатся вничью, но вас интересует подобная ситуация с минимально возможным количеством ничьих.

Ваша задача — найти ситуацию (выбрать результат каждой игры), в которой у всех команд одинаковый итоговый счет, а количество ничьих минимально возможно.

Входные данные:

В первой строке задано одно целое число 𝑡 (1≤𝑡≤100) — количество наборов входных данных.

Затем следуют сами наборы входных данных. Каждый набор описывается одной строкой, содержащей целое число 𝑛 (2≤𝑛≤100) — количество команд.

Выходные данные:

Для каждого набора входных данных выведите 𝑛(𝑛−1)2 целых чисел, описывающих результаты игр в следующем порядке: первое число соответствует результату матча между командой 1 и командой 2, второе — между командой 1 и командой 3, затем 1 и 4, ..., 1 и 𝑛, 2 и 3, 2 и 4, ..., 2 и 𝑛, и так далее, до результата матча между командой 𝑛−1 и командой 𝑛.

Число, соответствующее результату матча между командой 𝑥 и командой 𝑦, должно быть равно 1, если побеждает команда 𝑥, −1, если побеждает команда 𝑦, или 0, если матч заканчивается вничью.

Все команды должны иметь одинаковый итоговый счет, и количество ничьих должно быть минимально возможным. Если оптимальных ответов несколько — выведите любой из них. Можно показать, что всегда есть способ составить результаты игр так, что все команды имеют одинаковый итоговый счет.

Пример:

Входные данные:

2

2

3

Выходные данные:

0

1 -1 1

Примечание:

В первом наборе входных данных из примера обе команды получают по 1 очку, так как их матч заканчивается вничью.

Во втором наборе входных данных из примера команда 1 побеждает команду 2 (команда 1 получает 3 очка), команда 1 проигрывает команде 3 (команда 3 получает 3 очка), и команда 2 выигрывает у команды 3 (команда 2 получает 3 очка).

Код для решения задачи 1 представлен в Приложении 1. Результаты выполнения программы представлены на рисунке 1.

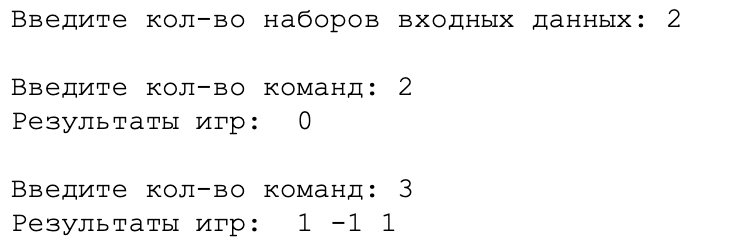


Рисунок 1 – Решение задачи 1

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 2-3.

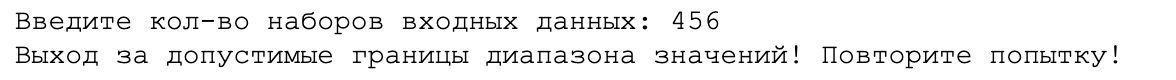


Рисунок 2 – Сообщение при выходе за допустимые границы диапазона значений

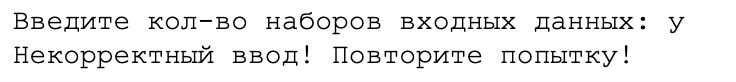


Рисунок 3 – Сообщение при вводе некорректных данных

**Задача 2**

У вас есть последовательность 𝑎 из 𝑛 элементов 1, 2, 3, …, 𝑘−1, 𝑘, 𝑘−1, 𝑘−2, …, 𝑘−(𝑛−𝑘) (𝑘≤𝑛<2𝑘).

Назовем инверсией в 𝑎 пару индексов 𝑖<𝑗 таких, что 𝑎[𝑖]>𝑎[𝑗].

Предположим, что у вас есть некоторая перестановка 𝑝 размера 𝑘 и вы строите последовательность 𝑏 размера 𝑛 следующим образом: 𝑏[𝑖]=𝑝[𝑎[𝑖]].

Ваша задача — найти такую перестановку 𝑝, что суммарное количество инверсий в 𝑏 не превосходит суммарного количества инверсий в 𝑎, и 𝑏 лексикографически максимальна.

Напоминание: последовательность из 𝑘 целых чисел называется перестановкой, если она содержит все числа от 1 по 𝑘 ровно по одному разу.

Еще одно напоминание: последовательность 𝑠 лексикографически меньше последовательности 𝑡, если либо 𝑠 — префикс 𝑡, или для первого 𝑖, для которого 𝑠𝑖≠𝑡𝑖, выполняется 𝑠𝑖<𝑡𝑖 (в первой позиции, в которой эти последовательности различаются, элемент в 𝑠 меньше элемента в 𝑡).

Входные данные:

В первой строке задано одно целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных.

В первой и единственной строке каждого набора заданы два целых числа 𝑛 и 𝑘 (𝑘≤𝑛<2𝑘; 1≤𝑘≤105) — длина последовательности 𝑎 и ее максимум.

Гарантируется, что сумма 𝑘 по всем наборам входных данных не превосходит 105.

Выходные данные:

Для каждого набора входных данных, выведите 𝑘 целых чисел — перестановку 𝑝, которая максимизирует 𝑏 лексикографически, не увеличивая суммарное количество инверсий.

Можно доказать, что 𝑝 существует и единственная.

Пример:

Входные данные:

4

1 1

2 2

3 2

4 3

Выходные данные:

1

1 2

2 1

1 3 2

Примечание:

В первом наборе входных данных, последовательность 𝑎 = [1], поэтому существует только одна перестановка 𝑝 = [1].

Во втором наборе, последовательность 𝑎 = [1,2]. В 𝑎 нет инверсий, а потому только одна перестановка 𝑝 = [1,2] не увеличивает количество инверсий.

В третьем наборе, 𝑎 = [1, 2, 1] и имеет 1 инверсию. Если мы используем 𝑝 = [2,1], то 𝑏 = [𝑝[𝑎[1]], 𝑝[𝑎[2]], 𝑝[𝑎[3]]] = [2, 1, 2] и также имеет 1 инверсию.

В четвертом наборе, 𝑎 = [1, 2, 3, 2], и так как 𝑝 = [1, 3, 2] то 𝑏 = [1, 3, 2, 3]. И 𝑎, и 𝑏 имеют по 1 инверсии и 𝑏 — лексикографически максимальна.

Код для решения задачи 2 представлен в Приложении 2. Результаты выполнения программы представлены на рисунке 4.

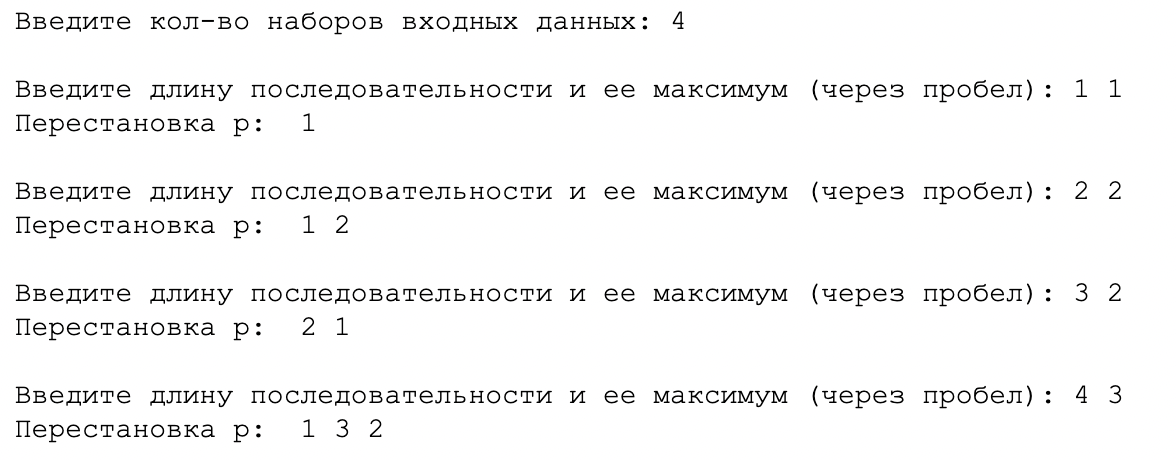


Рисунок 4 – Решение задачи 2

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 5-6.

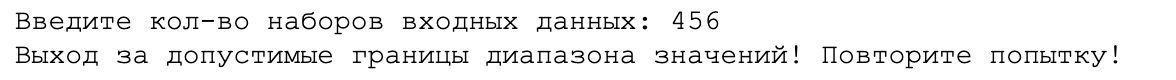


Рисунок 5 – Сообщение при выходе за допустимые границы диапазона значений

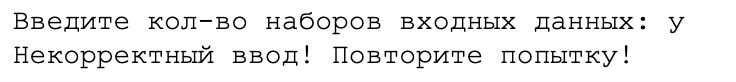


Рисунок 6 – Сообщение при вводе некорректных данных

При достижении максимального значения суммы, заданного в условии задачи, программа выдает пользователю сообщение и прекращает свою работу, что продемонстрировано на рисунке 7.

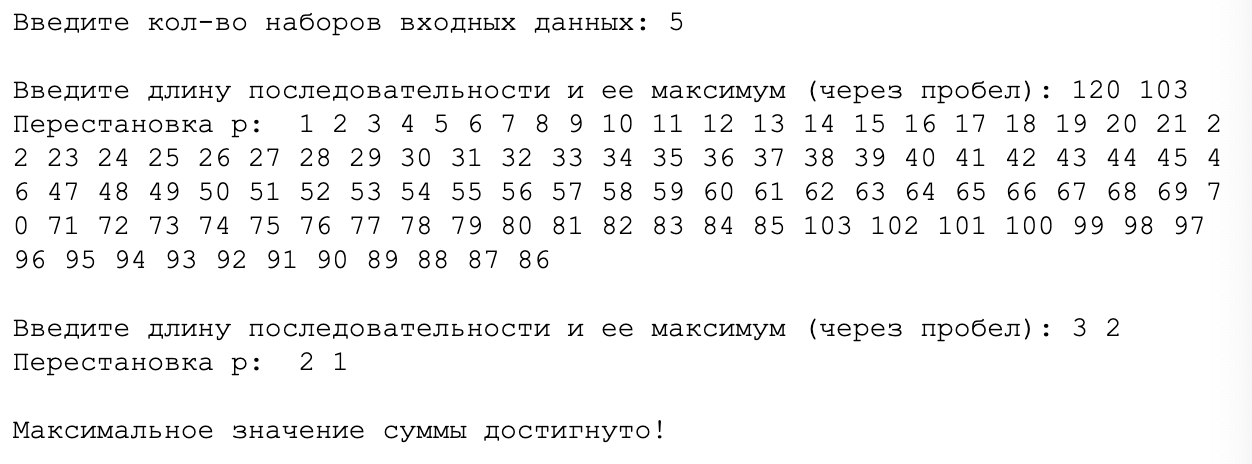


Рисунок 7 – Достижение максимального значения суммы

**Задача 3**

Дан набор из 𝑛 (𝑛 всегда равно степени 2) элементов, содержащих все целые числа 0, 1, 2, …, 𝑛−1 единожды.

Найдите 𝑛/2 пар элементов таких, что:

Каждый элемент набора принадлежит ровно одной паре. Сумма по всем парам побитового И элементов пары должна быть в точности равна 𝑘. Формально, если 𝑎𝑖 и 𝑏𝑖 — элементы 𝑖-й пары, то должно выполняться ∑𝑖=1𝑛/2𝑎𝑖&𝑏𝑖=𝑘, где & обозначает операцию побитового И. Если существует несколько решений, найдите любое из них. Если решений не существует, выведите −1.

Входные данные:

Каждый тест состоит из нескольких наборов входных данных. Первая строка ввода содержит одно целое число 𝑡 (1≤𝑡≤400) — количество наборов входных данных. Далее следует их описание.

Каждый набор входных данных состоит из одной строки, содержащей два целых числа 𝑛 и 𝑘 (4≤𝑛≤216, 𝑛 — степень 2, 0≤𝑘≤𝑛−1).

Сумма 𝑛 по всем наборам входных данных не превосходит 216. Все наборы в каждом тесте будут различными.

Выходные данные:

Для каждого набора входных данных, если решения не существует, выведите одну строку, содержащую −1.

В противном случае выведите 𝑛/2 строк, 𝑖-я их них должна содержать 𝑎𝑖 и 𝑏𝑖 — элементы 𝑖-й пары.

Если существует несколько решений, выведите любое. Порядок пар и элементов в парах не имеет значения.

Пример:

Входные данные:

4

4 0

4 1

4 2

4 3

Выходные данные:

0 3

1 2

0 2

1 3

0 1

2 3

-1

Примечание:

В первом наборе входных данных (0&3) + (1&2) = 0.

Во втором наборе (0&2) + (1&3) = 1.

В третьем наборе (0&1) + (2&3) = 2.

В четвертом наборе решения не существует.

Код для решения задачи 3 представлен в Приложении 3. Результаты выполнения программы представлены на рисунке 8.

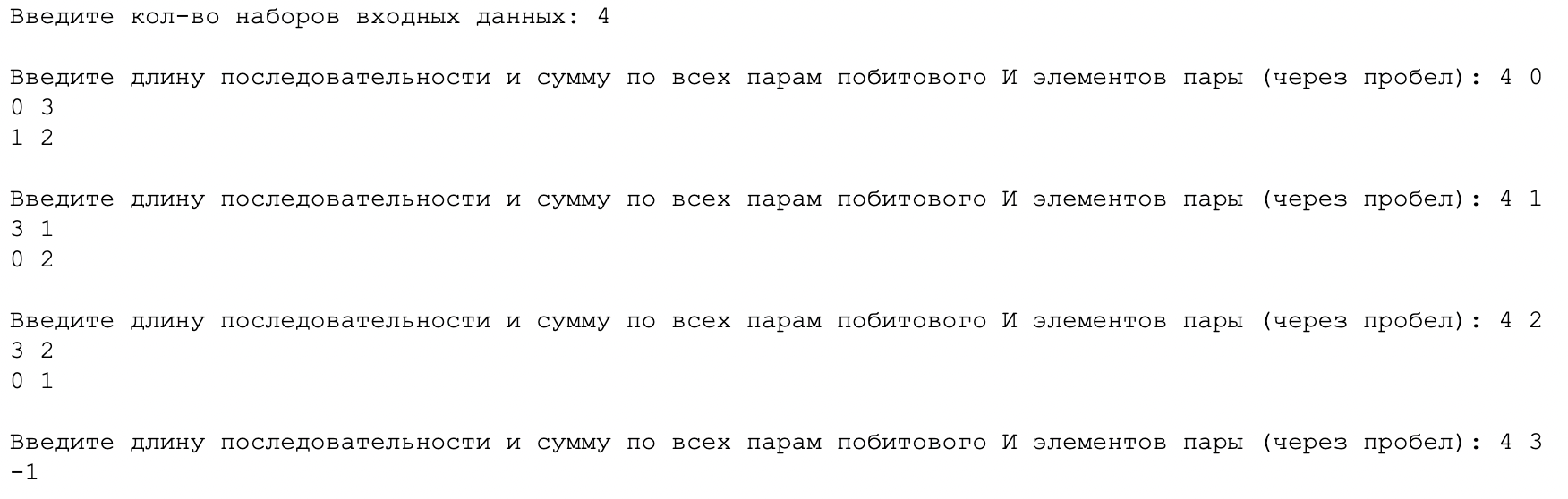


Рисунок 8 – Решение задачи 3

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 9-11.

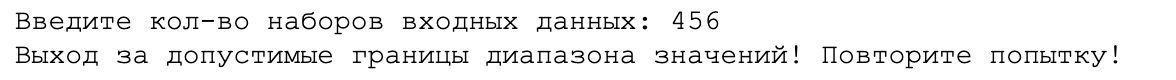


Рисунок 9 – Сообщение при выходе за допустимые границы диапазона значений

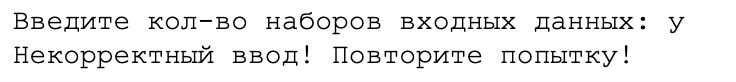


Рисунок 10 – Сообщение при вводе некорректных данных

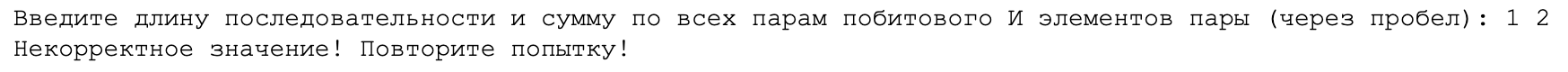


Рисунок 11 – Сообщение при вводе некорректных данных

### Задача 4

Вася и Петя играют в одну простую игру. Вася загадал число x от 1 до n, а Петя пытается угадать это число.

Петя может задавать вопросы вида: «Делится ли загаданное число на число y?».

Игра происходит по следующим правилам: вначале Петя спрашивает все вопросы, которые его интересуют (в том числе, он может не задать ни одного вопроса), затем Вася отвечает на каждый из вопросов «да» или «нет». После получения всех ответов Петя должен назвать число, которое загадал Вася.

К сожалению, Петя не слишком хорошо разбирается в теории чисел. Помогите ему найти минимальное количество вопросов, которые он должен задать, чтобы гарантированно угадать число Васи, а также сами числа yi, про которые он должен задать вопросы.

Входные данные:

В единственной строке записано число n (1 ≤ n ≤ 103).

Выходные данные:

Выведите длину искомой последовательности вопросов k (0 ≤ k ≤ n), а затем k чисел — саму последовательность вопросов yi (1 ≤ yi ≤ n).

Если существует несколько корректных последовательностей вопросов минимальной длины, то разрешается вывести любую.

Примеры:

Входные данные:

4

Выходные данные:

3

2 4 3

Входные данные:

6

Выходные данные:

2 4 3 5

Примечание:

Последовательность из ответа на первый тест из условия действительно корректна.

Если загаданное число не делится ни на одно из чисел последовательности, то оно равно 1.

Если же загаданное число делится на 4, то оно равно 4.

Если загаданное число делится на 3, то загаданное число равно 3.

Иначе, оно равно 2. Стало быть, эта последовательность вопросов действительно угадывает загаданное число. Можно показать, что не существует последовательности вопросов, состоящей из менее, чем трёх вопросов, удовлетворяющей условию.

Код для решения задачи 4 представлен в Приложении 4. Результаты выполнения программы представлены на рисунке 12.

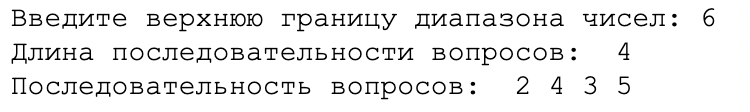


Рисунок 12 – Решение задачи 4

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 13-14.

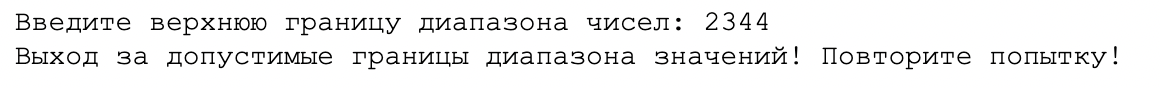


Рисунок 13 – Сообщение при выходе за допустимые границы диапазона значений

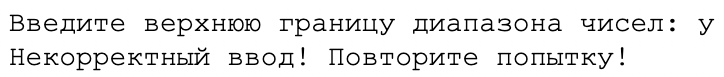


Рисунок 14 – Сообщение при вводе некорректных данных

### Задача 5

Где-то в параллельном Средиземье, когда Саруман поймал и обыскал Фродо, он обнаружил лишь 𝑛 колец. Причем 𝑖-е кольцо было либо золотым, либо серебряным. Для удобства Саруман записал бинарную строку 𝑠 из 𝑛 символов, где 𝑖-м символом был 0, если 𝑖-е кольцо золотое, и 1, если серебряное.

У Сарумана есть волшебная функция 𝑓, которая принимает бинарную строку, а в ответ выдает число, полученное путем перевода строки в двоичное число, а затем перевода двоичного числа в десятичное. Например, 𝑓(001010) = 10, 𝑓(111) = 7, 𝑓(11011101) = 221.

Саруман, однако, считает, что порядок колец играет какую-то важную роль. Он хочет найти 2 пары целых чисел (𝑙1, 𝑟1), (𝑙2, 𝑟2), для которых:

1≤𝑙1≤𝑛, 1≤𝑟1≤𝑛, 𝑟1−𝑙1+1≥⌊𝑛2⌋

1≤𝑙2≤𝑛, 1≤𝑟2≤𝑛, 𝑟2−𝑙2+1≥⌊𝑛2⌋

Пары (𝑙1, 𝑟1) и (𝑙2, 𝑟2) различны. То есть, должно выполняться хотя бы одно из условий 𝑙1≠𝑙2 и 𝑟1≠𝑟2.

Пусть 𝑡 — подстрока 𝑠[𝑙1:𝑟1] строки 𝑠, а 𝑤 — подстрока 𝑠[𝑙2:𝑟2] строки 𝑠. Тогда существует неотрицательное целое число 𝑘, такое, что 𝑓(𝑡)=𝑓(𝑤)⋅𝑘.  
Здесь подстрока 𝑠[𝑙:𝑟] обозначает 𝑠𝑙𝑠𝑙+1…𝑠𝑟−1𝑠𝑟, а ⌊𝑥⌋ обозначает округление числа вниз до ближайшего целого.

Помогите Саруману решить эту задачу! Гарантируется, что при ограничениях задачи существует хотя бы одно решение.

Входные данные:

Каждый тест содержит несколько наборов входных данных.

В первой строке находится одно целое положительное число 𝑡 (1≤𝑡≤103) — количество наборов входных данных. Описание наборов входных данных приведено ниже.

В первой строке каждого набора входных данных находится одно целое положительное число 𝑛 (2≤𝑛≤2⋅104) — количество колец.

Во второй строке каждого набора входных данных находится непустая бинарная строка длины 𝑛.

Гарантируется, что сумма 𝑛 по всем наборам входных данных не превосходит 105.

Выходные данные:

Для каждого набора входных данных выведите в отдельной строке четыре числа 𝑙1, 𝑟1, 𝑙2, 𝑟2, которые обозначают начало первой подстроки, конец первой подстроки, начало второй подстроки и конец второй подстроки соответственно.

Если существует несколько правильных вариантов ответа, вы можете вывести любой.

Пример:

Входные данные:

7  
6  
101111  
9  
111000111  
8  
10000000  
5  
11011  
6  
001111  
3  
101  
30  
100000000000000100000000000000

Выходные данные:

3 6 1 3

1 9 4 9

5 8 1 4

1 5 3 5

1 6 2 4

1 2 2 3

1 15 16 30

Примечание:

В первом наборе входных данных 𝑓(𝑡) = 𝑓(1111) = 15, 𝑓(𝑤) = 𝑓(101) = 5.

Во втором наборе входных данных 𝑓(𝑡) = 𝑓(111000111) = 455, 𝑓(𝑤) = 𝑓(000111) = 7.

В третьем наборе входных данных 𝑓(𝑡) = 𝑓(0000) = 0, 𝑓(𝑤) = 𝑓(1000) = 8.

В четвертом наборе входных данных 𝑓(𝑡) = 𝑓(11011) = 27, 𝑓(𝑤) = 𝑓(011) = 3.

В пятом наборе входных данных 𝑓(𝑡) = 𝑓(001111) = 15, 𝑓(𝑤) = 𝑓(011) = 3.

Код для решения задачи 5 представлен в Приложении 5. Результаты выполнения программы представлены на рисунке 15.

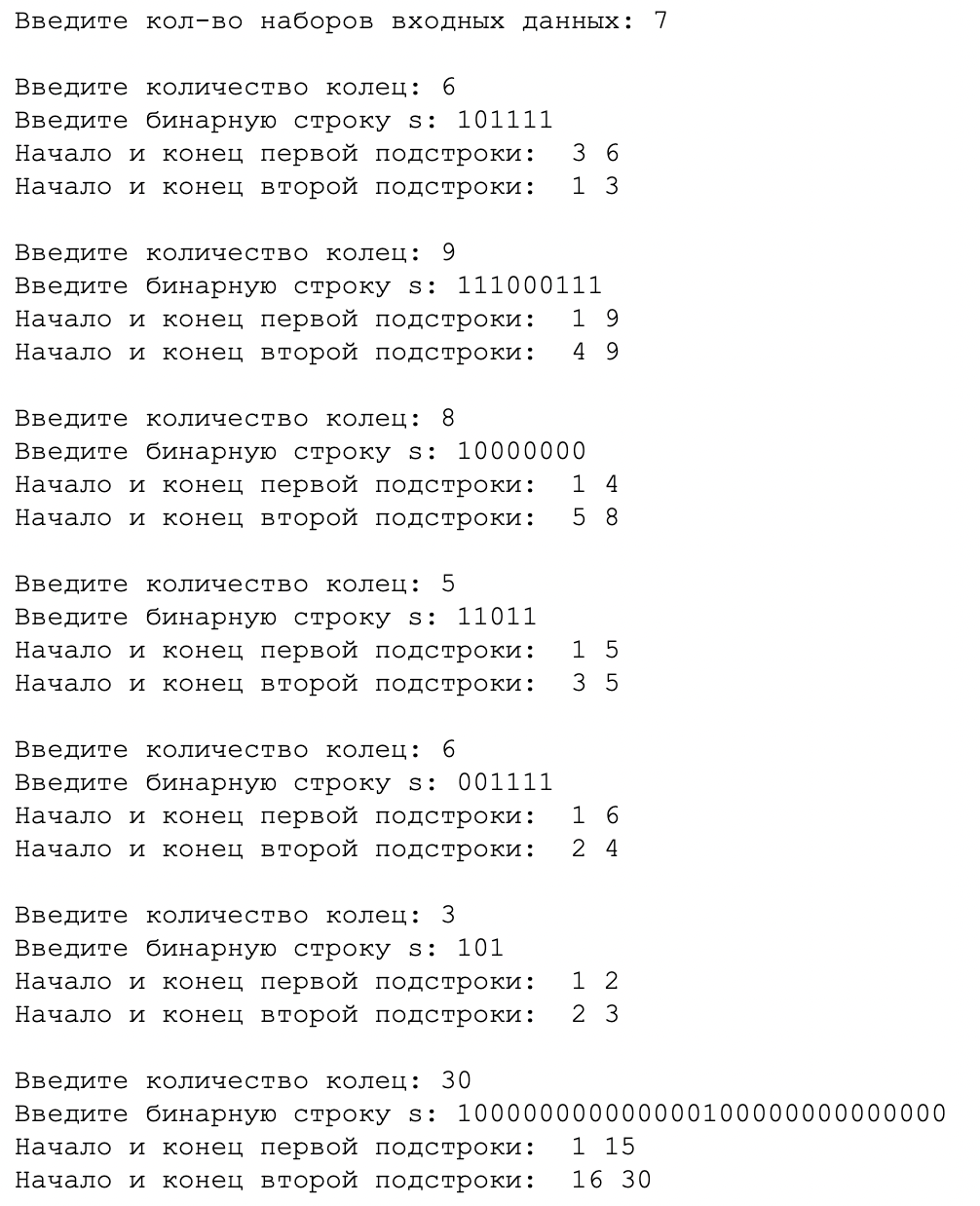


Рисунок 15 – Решение задачи 5

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 16-17.

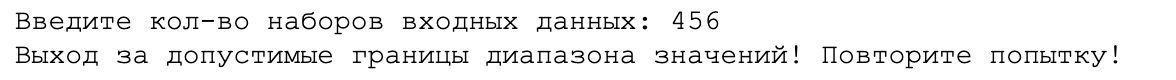


Рисунок 16 – Сообщение при выходе за допустимые границы диапазона значений

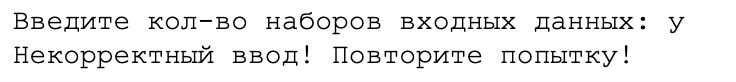


Рисунок 17 – Сообщение при вводе некорректных данных

При достижении максимального значения суммы, заданного в условии задачи, программа выдает пользователю сообщение, показанное на рисунке 18, и прекращает свою работу.



Рисунок 18 – Достижение максимального значения суммы

### Задача 6

Вам задана последовательность 𝑎 длины 𝑛, состоящая из целых чисел от 1 до 𝑛. Среди чисел могут быть одинаковые.

Найдите количество наборов из 𝑚=3 элементов, таких что максимальное число в наборе отличается от минимального не больше, чем на 𝑘=2. Формально, в этой задаче вам нужно найти количество троек индексов 𝑖<𝑗<𝑧, таких что max(𝑎𝑖, 𝑎𝑗, 𝑎𝑧) − min(𝑎𝑖, 𝑎𝑗, 𝑎𝑧) ≤ 2.

Например, если 𝑛 = 4 и 𝑎 = [1, 2, 4, 3], то существуют две такие тройки (𝑖=1, 𝑗=2, 𝑧=4 и 𝑖=2, 𝑗=3, 𝑧=4). Если же 𝑛 = 4 и 𝑎 = [1, 1, 1, 1], то все четыре возможные тройки подходят.

Входные данные:

В первой строке находится одно целое число 𝑡 (1≤𝑡≤2⋅105) — количество наборов входных данных. Далее следуют 𝑡 наборов входных данных.

В первой строке каждого набора входных данных находится целое число 𝑛 (1≤𝑛≤2⋅105) — длина последовательности 𝑎.

В следующей строке находятся 𝑛 целых чисел 𝑎1, 𝑎2, …, 𝑎𝑛 (1≤𝑎𝑖≤𝑛) — последовательность 𝑎.

Гарантируется, что сумма 𝑛 по всем наборам входных данных не превышает 2⋅105.

Выходные данные:

Выведите 𝑡 ответов на наборы входных данных. Каждый ответ — это количество упорядоченных троек элементов, таких что максимальное число в тройке отличается от минимального не больше, чем на 2. Обратите внимание, что в отличии от сложной версии задачи, здесь не нужно выводить ответ по какому-либо модулю. Необходимо вывести точное значение ответа.

Пример:

Входные данные:

4

4

1 2 4 3

4

1 1 1 1

1

1

10

5 6 1 3 2 9 8 1 2 4

Выходные данные:

2  
4  
0  
15

Код для решения задачи 6 представлен в Приложении 6. Результаты выполнения программы представлены на рисунке 19.

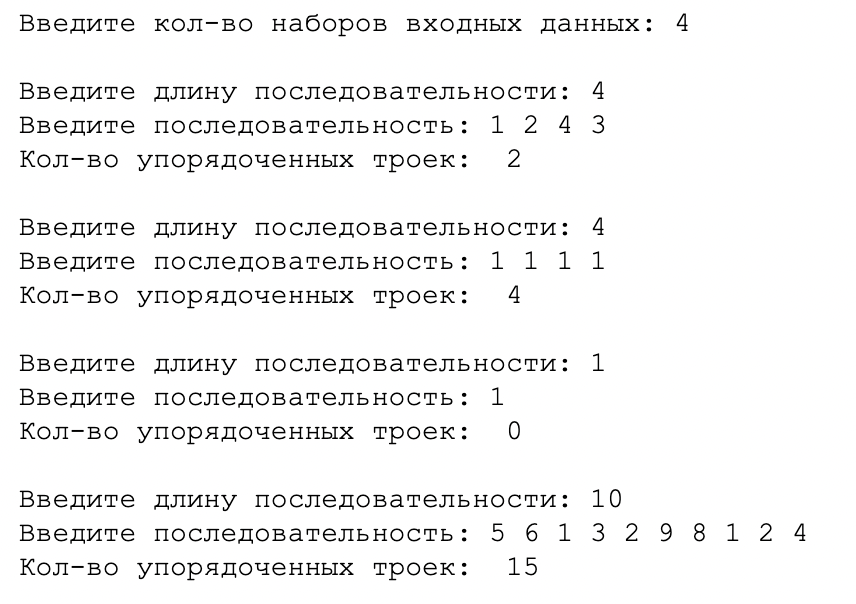


Рисунок 19 – Решение задачи 6

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 20-21.

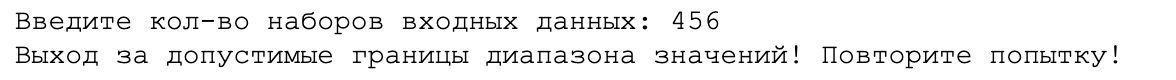


Рисунок 20 – Сообщение при выходе за допустимые границы диапазона значений

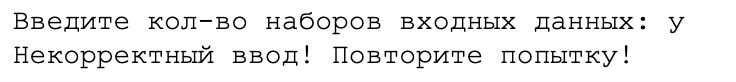


Рисунок 21 – Сообщение при вводе некорректных данных

**Задача 7**

Ану придумала свою собственную функцию 𝑓: 𝑓(𝑥, 𝑦) = (𝑥|𝑦) − 𝑦, где | обозначает операцию побитового ИЛИ. К примеру, 𝑓(11, 6) = (11|6) – 6 = 15 – 6 = 9. Можно показать, что для любых неотрицательных чисел 𝑥 и 𝑦 значение 𝑓(𝑥, 𝑦) также неотрицательное.

Она хотела бы исследовать данную функцию, и придумала несколько задач для себя. К сожалению, она не может решить все из них, и ей нужна ваша помощь. Вот одна из этих задач.

Значение массива [𝑎1, 𝑎2, …, 𝑎𝑛] определяется как 𝑓(𝑓(…𝑓(𝑓(𝑎1, 𝑎2), 𝑎3), …𝑎𝑛−1), 𝑎𝑛) (для примеров обратитесь к примечаниям). Вам дан массив из не обязательно различных чисел. Как нужно переставить местами его элементы, чтобы его значение стало максимальным возможным?

Входные данные:

Первая строка содержит одно целое число 𝑛 (1≤𝑛≤105) — длина массива.

Вторая строка содержит 𝑛 целых чисел 𝑎1, 𝑎2, …, 𝑎𝑛 (0≤𝑎𝑖≤109) — элементы массива. Не гарантируется, что элементы массива различны.

Выходные данные:

Выведите 𝑛 чисел — перестановку массива с максимальным значением. Если существует несколько решений, выведите любое из них.

Примеры:

Входные данные:

4

4 0 11 6

Выходные данные:

11 6 4 0

Примечание:

В первом тестовом случае, значение массива [11, 6, 4, 0] равно 𝑓(𝑓(𝑓(11, 6), 4), 0) = 𝑓(𝑓(9, 4), 0) = 𝑓(9, 0) = 9.

[11, 4, 0, 6] также является верным ответом.

Код для решения задачи 7 представлен в Приложении 7. Результаты выполнения программы представлены на рисунке 22.

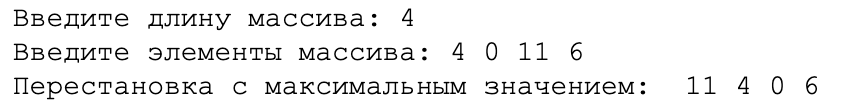


Рисунок 22 – Решение задачи 7

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 23-24.

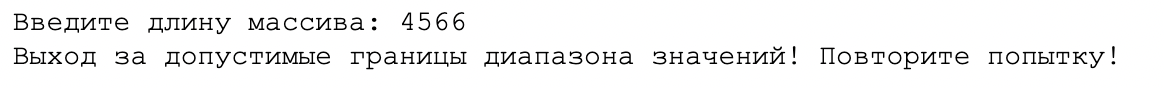


Рисунок 23 – Сообщение при выходе за допустимые границы диапазона значений

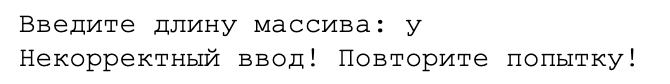


Рисунок 24 – Сообщение при вводе некорректных данных

### Задача 8

Магазин целых чисел продаёт 𝑛 отрезков, 𝑖-й из которых состоит из чисел от 𝑙𝑖 до 𝑟𝑖 и стоит 𝑐𝑖 монет.

Завтра Вася отравится в этот магазин и купит несколько отрезков. Он получит все числа, лежащие хотя бы в одном из выбранных отрезков. Количество монет, которое он потратит на покупку равно сумме стоимостей отрезков, которые он купит.

В качестве подарка за покупку, Вася дополнительно получит ещё несколько целых чисел. Число 𝑥, не купленное Васей ранее, он получит тогда и только тогда, когда для 𝑥 выполняются два следующих условия:

Найдётся купленное Васей число 𝑙, меньшее 𝑥.

Найдётся купленное Васей число 𝑟, большее 𝑥.

Число 𝑥 Вася может получить в подарок не более одного раза, таким образом все числа, которые получит Вася будут различными.

Например, если Вася купит отрезок [2, 4] за 20 монет и отрезок [7, 8] за 22 монеты, он потратит 42 монеты и получит числа 2, 3, 4, 7, 8 как содержимое отрезков. В качестве подарка он получит числа 5 и 6.

По техническим причинам, завтра в магазине будут доступны к покупке только 𝑠 первых отрезков (то есть отрезки [𝑙1, 𝑟1], [𝑙2, 𝑟2], …, [𝑙𝑠, 𝑟𝑠]). Вася не сможет купить отрезки, не доступные к покупке.

Вася хочет получить (купить или получить в подарок) как можно больше целых чисел. Если это можно сделать разными способами, то он выберет самый дешёвый из них.

Для всех значений 𝑠 от 1 до 𝑛 определите, сколько монет потратит Вася, если к покупке будут доступны только 𝑠 первых отрезков.

Входные данные:

В первой строке задано одно целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных. Далее следуют описания этих наборов.

В первой строке дано одно число 𝑛 (1≤𝑛≤105) — количество отрезков в магазине.

В следующих 𝑛 строках дано по три числа 𝑙𝑖, 𝑟𝑖, 𝑐𝑖 (1≤𝑙𝑖≤𝑟𝑖≤109,1≤𝑐𝑖≤109) — концы 𝑖-го отрезка и его стоимость.

Гарантируется, что сумма 𝑛 по всем наборам входных данных не превосходит 2⋅105.

Выходные данные:

Для каждого набора входных данных выведите 𝑛 чисел: 𝑠-е (1≤𝑠≤𝑛) из них должно быть равно количеству монет, которое потратит Вася, если к покупке будут доступны только первые 𝑠 отрезков.

Пример:

Входные данные:

3

2

2 4 20

7 8 22

2

5 11 42

5 11 42

6

1 4 4

6 8 9

7 8 7

2 10 252

1 11 271

1 10 1

Выходные данные:

20

42

42

42

4

13

11

256

271

271

Примечания:

В первом наборе входных данных при 𝑠 = 1 Вася может купить только один отрезок [2, 4] за 20 монет и получить 3 числа.

Способ получить 7 чисел за 42 монеты при 𝑠 = 2 описан в условии.

Во втором наборе входных данных обратите внимание на то, что в магазине могут быть одинаковые отрезки.

Код для решения задачи 8 представлен в Приложении 8. Результаты выполнения программы представлены на рисунке 25.

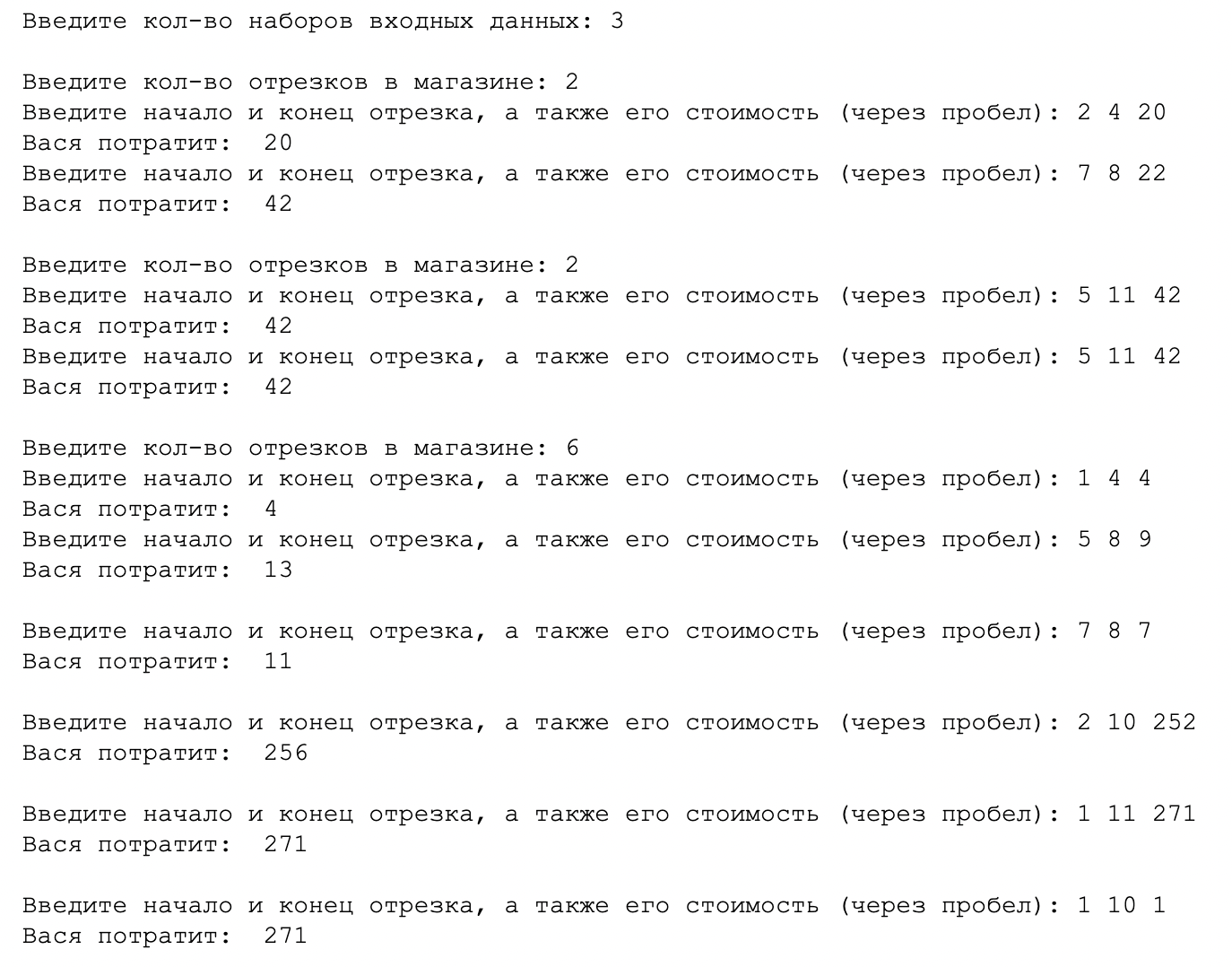


Рисунок 25 – Решение задачи 8

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 26-27.

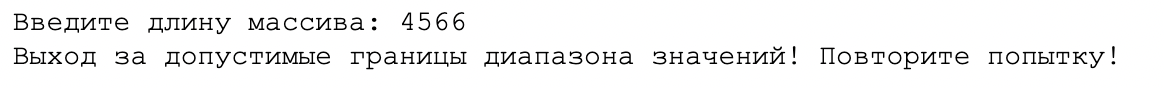


Рисунок 26 – Сообщение при выходе за допустимые границы диапазона значений

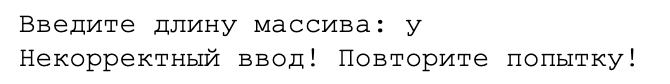


Рисунок 27 – Сообщение при вводе некорректных данных

### Задача 9

Предположим, вы стоите на плоскости 𝑋𝑌 в точке (0, 0) и хотите попасть в точку (𝑛, 𝑛).

Вы можете двигать только в двух направлениях: вправо, т. е. горизонтально и в направлении увеличения 𝑥 координаты, или вверх, т. е. вертикально и в направлении увеличения 𝑦 координаты. Другими словами, ваш путь имеет следующую структуру: первоначально, вы выбираете: пойти вправо или вверх; далее вы проходите некоторое положительное целое расстояние в выбранном направлении (расстояния можно выбирать независимо); далее вы меняете направление движения (от «вправо» к «вверх», либо от «вверх» к «вправо») и повторяете процесс. Вам не нравится менять свое направление слишком много раз, а потому вы сделаете не более 𝑛−1 изменений направления.

В результате ваш путь будет представлять ломаную от (0, 0) к (𝑛, 𝑛), состоящую из не более чем 𝑛 отрезков, каждый из которых имеет положительную целую длину, а также вертикальные и горизонтальные отрезки идут по очереди.

Не все пути равны. У вас есть 𝑛 целых чисел 𝑐1, 𝑐2, …, 𝑐𝑛, где 𝑐𝑖 — это цена 𝑖-го отрезка.

Используя эти цены, можно определить цену пути как сумму длин отрезков этого пути, умноженных на их стоимости, т. е. если путь состоит из 𝑘 отрезков (𝑘≤𝑛), то цена пути равна ∑𝑖=1𝑘𝑐𝑖⋅𝑙𝑒𝑛𝑔𝑡ℎ𝑖 (отрезки нумеруются от 1 по 𝑘 в порядке их обхода).

Определите путь минимальной стоимости и выведите его цену.

Входные данные:

В первой строке задано одно целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных.

В первой строке каждого набора задано одно целое число 𝑛 (2≤𝑛≤105).

Во второй строке каждого набора заданы 𝑛 целых чисел 𝑐1, 𝑐2, …, 𝑐𝑛 (1≤𝑐𝑖≤109) — цены каждого отрезка.

Гарантируется, что сумма 𝑛 не превосходит 105.

Выходные данные:

Для каждого набора входных данных вы должны вывести минимальную цену пути из (0, 0) в (𝑛, 𝑛), состоящего из не более 𝑛 чередующихся отрезков.

Пример:

Входные данные:

3

2

13 88

3

2 3 1

5

4 3 2 1 4

Выходные данные:

202

13

19

Примечание:

В первом примере из условия, чтобы достигнуть (2,2), нужно сделать хотя бы один поворот, и путь может состоять из 2 отрезков: горизонтального длины 2 и вертикального длины 2. Стоимость равна 2⋅𝑐1+2⋅𝑐2=26+176=202.

Во втором примере можно построить путь из 3 отрезков: длины 1, длины 3 и длины 2. Стоимость равна 1⋅2+3⋅3+2⋅1=13.

В третьем примере можно построить путь из 4 отрезков: длины 1, длины 1, длины 4 и длины 4. Стоимость равна 1⋅4+1⋅3+4⋅2+4⋅1=19.

Код для решения задачи 9 представлен в Приложении 9. Результаты выполнения программы представлены на рисунке 28.

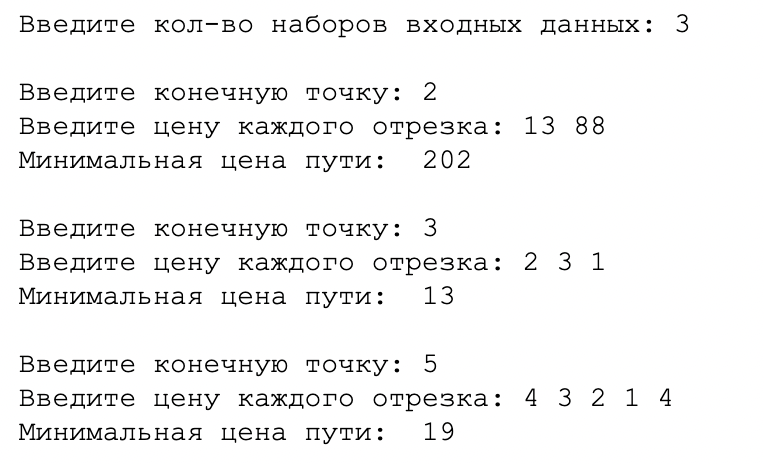


Рисунок 28 – Решение задачи 9

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 29-30.

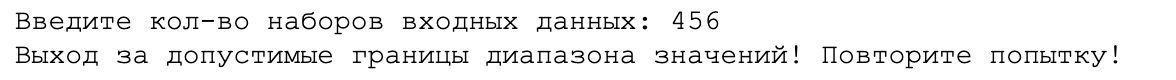


Рисунок 29 – Сообщение при выходе за допустимые границы диапазона значений

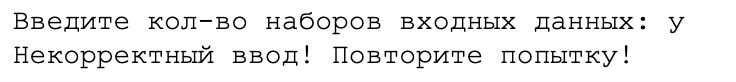


Рисунок 30 – Сообщение при вводе некорректных данных

При достижении максимального значения суммы, заданного в условии задачи, программа выдает пользователю сообщение, показанное на рисунке 31, и прекращает свою работу.



Рисунок 31 – Достижение максимального значения суммы

### Задача 10

Необъяснимый народ эти битландцы. Какие же у них странные обычаи!

Как обычно, дядя J. хочет раскрасить n яиц для известного Битландского фестиваля, который называется Битруз. Дядя J. попросил выполнить эту работу G. и A. Дети очень обрадовались. Ведь они знают, что им заплатят за работу.

Всего у дяди J. есть n яиц. Для каждого яйца G. назвал количество денег, которое он требует за покраску этого яйца. Аналогично, для каждого яйца A. назвал количество денег, которое он требует за покраску этого яйца. Оказалось, что для каждого яйца сумма количеств денег, которые требует за покраску A. и G., равна 1000.

Дядя J. хочет поделить яйца между детьми так, чтобы каждое яйцо было отдано на покраску ровно одному ребенку. Также дядя хочет, чтобы сумма денег, которая достанется A. отличалась от суммы денег, которая достанется G. не более чем на 500.

Помогите дяде J. Найдите требуемое распределение яиц или скажите, что разделить яйца требуемым образом не получится.

Входные данные:

В первой строке записано целое число n (1 ≤ n ≤ 106) — количество яиц. В каждой из следующих n строк записано два целых числа: ai, gi (0 ≤ ai, gi ≤ 1000; ai + gi = 1000) — количество денег, которое требует A. за покраску i-того яйца, и количество денег, которое требует G. за покраску i-того яйца.

Выходные данные:

Если разделить яйца между детьми требуемым образом нельзя, выведите «-1» (без кавычек).

Иначе выведите строку, состоящую из n символов «G» и «A»: i-тый символ должен обозначать того, кому достанется i-тое яйцо в требуемом распределении (символ «A» обозначает A., символ «G» обозначает G.). Если обозначить сумму денег, которую дядя должен будет отдать A. за покраску, через Sa, а сумму денег, которую дядя должен будет отдать G. за покраску, через Sg, должно выполняться неравенство |Sa - Sg| ≤ 500.

Если существует несколько решений, разрешается вывести любое.

Примеры:

Входные данные:

2

1 999

999 1

Выходные данные:

AG

Входные данные:

400 600

400 600

400 600

Выходные данные:

AGA

Код для решения задачи 10 представлен в Приложении 10. Результаты выполнения программы представлены на рисунке 32.

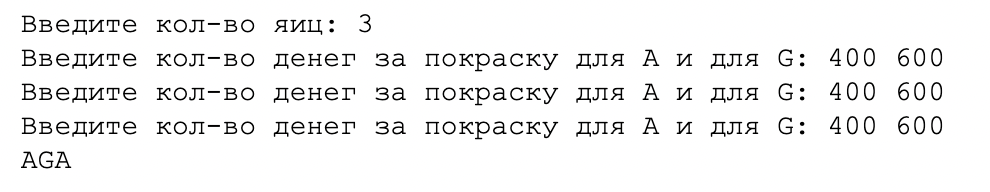


Рисунок 32 – Решение задачи 10

Также в ходе работы была реализована защита от неверно введенных данных. Сообщения об ошибках при вводе представлены на рисунках 29-30.

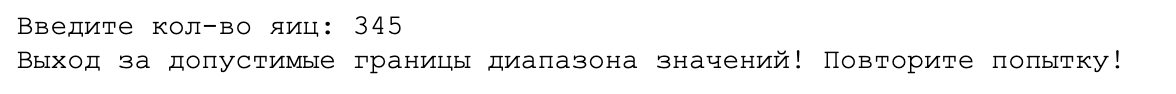


Рисунок 29 – Сообщение при выходе за допустимые границы диапазона значений

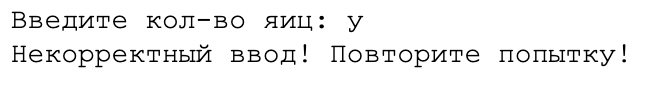


Рисунок 30 – Сообщение при вводе некорректных данных

**Вывод**: в ходе данной работы было решено 10 задач на основе варианта.

**Приложения**

Приложение 1. Листинг кода для решения задачи 1

def num1(n):

#получаем количество матчей

kol = 0

for i in range(1, n):

kol += n - i

ans = []

#для нечетного кол-ва команд

if n % 2 == 1:

for i in range(kol):

if i % 2 == 0:

ans.append("1")

else:

ans.append("-1")

#для четного кол-ва команд

else:

r = n - 1

ans = [0] \* r

for i in range(r):

temp = [0] \* (r - i)

for j in range(r - i):

if i % 4 == 0:

if j == 0:

temp[j] = 0

elif j % 2 == 1:

temp[j] = 1

else:

temp[j] = -1

elif i % 4 == 1:

if j % 2 == 0:

temp[j] = -1

else:

temp[j] = 1

elif i % 4 == 2:

if j == 0:

temp[j] = 0

elif j % 2 == 1:

temp[j] = -1

else:

temp[j] = 1

else:

if j % 2 == 0:

temp[j] = 1

else:

temp[j] = -1

ans[i] = temp.copy()

ans = sum(ans, [])

return ans

try:

t = int(input("Введите кол-во наборов входных данных: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= t and t <= 100:

print()

while t > 0:

try:

n = int(input("Введите кол-во команд: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 2 <= n and n <= 100:

ans = num1(n)

print("Результаты игр: ", \*ans, '\n')

t -= 1

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 2. Листинг кода для решения задачи 2

d ef num2(n, k):

#х - длина последовательности от 1 до к и снова до 1

x = k \* 2 - 1

ans = []

#добавляем в ответ числа от 1 до x-n

for i in range(x - n):

ans.append(i + 1)

y = k

#добавляем в ответ числа от k до 1 (если необходимо)

for i in range(k - len(ans)):

ans.append(y)

y -= 1

return ans

#сумма k для проверки входных данных

sum\_k = 0

try:

t = int(input("Введите кол-во наборов входных данных: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= t and t <= 1000:

print()

while t > 0:

try:

n, k = map(int, input("Введите длину последовательности и ее максимум (через пробел): ").split())

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

sum\_k += k

if k <= n and n <= 2\*k and 1 <= k and k <= 105 and sum\_k <= 105:

ans = num2(n, k)

print("Перестановка p: ", \*ans, '\n')

if sum\_k == 105:

print("Максимальное значение суммы достигнуто!")

break

t -= 1

else:

sum\_k -= k

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 3. Листинг кода для решения задачи 3

def num3(n, k):

p = [] #список, куда мы будем записывать пары

s = [] #список, куда мы будем записывать числа, которые уже взяли в р

#случай когда n = 4, k = 3

if k == n - 1 and k == 3:

return -1

if k == n - 1:

num = n - 1

i = 1

#вычисляем пары и добавляем их в p и s

while i < n:

opp = n - i - 1

p.append((i, num))

s.append(i)

s.append(num)

num = opp

i \*= 2

p.append((0, opp))

s.append(0)

s.append(opp)

if k > 0 and k < n - 1:

#добавляем пары в список p

p.append((n - 1, k))

p.append((0, n - k - 1))

#добавляем выбранные в пары числа в список s

s.append(0)

s.append(k)

s.append(n - 1)

s.append(n - k - 1)

#добавляем в ответ недостающие пары (если нужно)

for i in range(0, n):

if i in s:

continue

j = n - 1 - i

if j < i:

break

p.append((i, j))

return p

#проверка на степень 2

def check\_2(n):

fl = False

i = 1

while i < n:

i = i \* 2

if i == n:

fl = True

return fl

try:

t = int(input("Введите кол-во наборов входных данных: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= t and t <= 400:

print()

while t > 0:

try:

n, k = map(int, input("Введите длину последовательности и сумму по всех парам побитового И элементов пары (через пробел): ").split())

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 4 <= n and n <= 216 and 0 <= k and k <= n-1 and check\_2(n):

ans = num3(n, k)

if ans == -1:

print(ans)

else:

for a, b in ans:

print(a, b)

print()

t -= 1

else:

print("Некорректное значение! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 4. Листинг кода для решения задачи 4

def num4(n):

k = [] #последовательность вопросов

#простые числа

primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67,

71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139,

149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223,

227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293,

307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383,

389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463,

467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569,

571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647,

653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743,

751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839,

853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941,

947, 953, 967, 971, 977, 983, 991, 997]

for p in primes:

if (p <= n):

i = 1

y = p

while y <= n:

#добавляем число в список k

k.append(y)

i += 1

#возводим число в степень i

y = p\*\*i

else:

break

return k

try:

n = int(input("Введите верхнюю границу диапазона чисел: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= n and n <= 103:

ans = num4(n)

print("Длина последовательности вопросов: ", len(ans))

print("Последовательность вопросов: ", \*ans, '\n')

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 5. Листинг кода для решения задачи 5

#функция f - переводит число из 2-ичной в 10-ичную сс

def f(s, l, r):

num = 0

leng = r-l+1

for i in range(leng):

num += int(s[i]) \* (2\*\*(leng - i - 1))

return num

def num5(n, s):

k1 = 0

k0 = 0

#ищем кол-во 0 и 1 в строке s

for i in range(n):

if s[i] == '0':

k0 += 1

else:

k1 += 1

if (k0 > k1):

l1, r1, l2, r2 = 1, n//2, n//2+1, n

#если f от первой последовательности больше чем от второй - меняем их местами

if f(s, l1, r1) <= f(s, l2, r2):

return l1, r1, l2, r2

else:

return l2, r2, l1, r1

elif s[0] == s[1]:

for i in range(n):

if s[i] == '0' and s[0] == '1':

l1, r1, l2, r2 = 1, n, i+1, n

break

elif s[i] == '0' and s[0] == '0':

l1, r1, l2, r2 = 1, n, i+1, n-2

return l1, r1, l2, r2

elif s[0] != s[1]:

if (n % 2 == 0):

l1, r1, l2, r2 = n//2, n, 1, n//2

else:

l1, r1, l2, r2 = n//2 + 1, n, 1, n//2 + 1

if f(s, l1, r1) > f(s, l2, r2):

return l1, r1, l2, r2

else:

return l2, r2, l1, r1

else:

l1, r1, l2, r2 = 1, n-1, 2, n

return l1, r1, l2, r2

#проверка строки, чтобы она состояла только из 0 и 1

def check\_bin(s, n):

fl = True

for i in range(n):

if s[i] != "0" and s[i] != "1":

fl = False

break

return fl

sum\_n = 0

try:

t = int(input("Введите кол-во наборов входных данных: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= t and t <= 103:

print()

while t > 0:

try:

n = int(input("Введите количество колец: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 2 <= n and n <= 2\*104:

try:

s = input("Введите бинарную строку s: ")

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

sum\_n += n

if len(s) == n and check\_bin(s, n) and sum\_n <= 105:

l1, r1, l2, r2 = num5(n, s)

print("Начало и конец первой подстроки: ", l1, r1)

print("Начало и конец второй подстроки: ", l2, r2, '\n')

if sum\_n == 105:

print("Максимальное значение суммы достигнуто!")

break

t -= 1

else:

sum\_n -= n

print("Некорректное значение! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 6. Листинг кода для решения задачи 6

def num6(n, a):

#если длина последовательности < 3, тройку сформировать невозможно

if (n < 3):

ans = 0

else:

ans = 0

#проверяем пару из i-, j- и z-ого значения, если удовлетворяет условию - увеличиваем ответ

for i in range(n-2):

for j in range(i+1, n-1):

for z in range(j+1, n):

if (max(a[i], a[j], a[z]) - min(a[i], a[j], a[z]) <= 2):

ans += 1

return ans

#сумма n для проверки входных данных

sum\_n = 0

try:

t = int(input("Введите кол-во наборов входных данных: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= t and t <= 2\*105:

print()

while t > 0:

try:

n = int(input("Введите длину последовательности: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 1 <= n and n <= 2\*105:

try:

a = list(map(int,input("Введите последовательность: ").split()))

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if len(a) == n:

fl = True

for i in range(n):

if 1 > a[i] or a[i] > n:

fl = False

sum\_n += n

if fl and sum\_n <= 2\*105:

ans = num6(n, a)

print("Кол-во упорядоченных троек: ", ans, '\n')

if sum\_n == 2\*105:

print("Максимальное значение суммы максимумов достигнуто!")

break

t -= 1

else:

sum\_n -= n

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Некорректный ввод! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 7. Листинг кода для решения задачи 7

def num7(n, a):

idx = 0

bit = 7 #максимальное введенное число(109) занимает 7 бит памяти

while bit >= 0:

count = 0

for i, num in enumerate(a):

#если при побитовом сдвиге числа вправо результат будет равен 1

if (num >> bit) & 1 == 1:

count += 1 #прибавляем count

idx = i #запоминаем текущий индекс

#если count больше 1, значит ранее уже встречалось число с 1 в текущей позиции

if count > 1:

break #выходим и проверяем следующее значение

#если count равен 1, значит мы нашли нужное число

if count == 1:

break #выходим

bit -= 1 #уменьшаем значение bit для поиска на следующей позиции

return idx

try:

n = int(input("Введите длину массива: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= n and n <= 105:

try:

a = list(map(int, input("Введите элементы массива: ").split()))

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if len(a) == n:

fl = True

for i in range(n):

if 0 > a[i] or a[i] > 109:

fl = False

if fl:

ans = num7(n, a)

print("Перестановка с максимальным значением: ", f"{a[ans]} ", end = "")

del a[ans]

print(" ".join(map(str, a)))

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Некорректный ввод! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 8. Листинг кода для решения задачи 8

def num8(n, low, high, cost, l, r, c, lowcost, highcost):

if low > l and high < r:

lowcost = c

highcost = c

cost = c

low = l

high = r

elif low > l:

lowcost = c

low = l

if r == high:

cost = c

if highcost > c:

highcost = c

else:

cost = lowcost + highcost

elif high < r:

highcost = c

high = r

if l == low:

cost = c

if lowcost > c:

lowcost = c

else:

cost = lowcost + highcost

elif low == l and high == r:

if cost > c:

cost = c

if highcost > c:

highcost = c

if lowcost > c:

lowcost = c

elif low == l:

if lowcost > c:

lowcost = c

if lowcost + highcost < cost:

cost = lowcost + highcost

elif high == r:

if highcost > c:

highcost = c

if lowcost + highcost < cost:

cost = lowcost + highcost

return cost, lowcost, highcost, low, high

try:

t = int(input("Введите кол-во наборов входных данных: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= t and t <= 1000:

print()

while t > 0:

try:

n = int(input("Введите кол-во отрезков в магазине: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 1 <= n and n <= 105:

try:

low, high, cost = map(int, input("Введите начало и конец отрезка, а также его стоимость (через пробел): ").split())

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 1 <= low <= high <= 10\*\*9 and 1 <= cost <= 10\*\*9:

lowcost = cost

highcost = cost

print("Вася потратит: ", cost)

for j in range(n-1):

try:

l, r, c = map(int, input("Введите начало и конец отрезка, а также его стоимость (через пробел): ").split())

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 1 <= l <= r <= 10\*\*9 and 1 <= c <= 10\*\*9:

for i in range(n-1):

cost, lowcost, highcost, low, high = num8(n, low, high, cost, l, r, c, lowcost, highcost)

print("Вася потратит: ", cost)

print()

t -= 1

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 9. Листинг кода для решения задачи 9

def num9(n, otr):

#задаем искомые минимумы (для х, у и ответ) бесконечностями

xmin = float('inf')

ymin = float('inf')

ans = float('inf')

temp = 0

x = n

y = n

for i in range(n):

#так как ходы должны чередоваться, на четных итерациях работаем с у, а на нечетных с х

if i%2 == 0:

#значение текущего пути

val = temp + y \* otr[i] + x \* xmin

#выбираем минимальное значение в ответ

ans = min(ans, val)

#уменьшаем у

y -= 1

#выбираем минимальное значение для у

ymin = min(ymin, otr[i])

#добавляем текущую цену отрезка к переменной temp

temp += otr[i]

else:

#значение текущего пути

val = temp + x \* otr[i] + y \* ymin

#выбираем минимальное значение в ответ

ans = min(ans, val)

#уменьшаем х

x -= 1

#выбираем минимальное значение для х

xmin = min(xmin, otr[i])

#добавляем текущую цену отрезка к переменной temp

temp += otr[i]

return ans

#сумма n для проверки введенных данных

sum\_n = 0

try:

t = int(input("Введите кол-во наборов входных данных: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= t and t <= 1000:

print()

while t > 0:

try:

n = int(input("Введите конечную точку: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if 2 <= n and n <= 105:

try:

otr = [int(i) for i in input("Введите цену каждого отрезка: ").split()]

except ValueError:

print("Некорректный ввод! Повторите попытку!\n")

else:

if len(otr) == n:

fl = True

for i in range(n):

if 1 > otr[i] or otr[i] > 109:

fl = False

sum\_n += n

if fl and sum\_n <= 105:

ans = num9(n, otr)

print("Минимальная цена пути: ", ans, '\n')

if sum\_n == 105:

print("Максимальное значение суммы достигнуто!")

break

t -= 1

else:

sum\_n -= n

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Некорректный ввод! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!\n")

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")

Приложение 10. Листинг кода для решения задачи 10

def num10(prices):

sA = 0 #сумма для А

sG = 0 #сумма для G

sol = []

for i, (A, G) in enumerate(prices):

#при рассмотрении первой(нулевой) строки выбираем меньшее число и добавляем в ответ

if i == 0:

if A <= G:

sol.append("A")

sA += A

else:

sol.append("G")

sG += G

#при рассмотрении следующих строк сравниваем суммы для А и G

else:

#если сумма A больше или равна сумме G

if sA >= sG:

#если после прибавления числа в А сумма А и сумма G будут отличаться не больше чем на 500, берем А

if sA + A - sG <= 500:

sol.append('A')

sA += A

#иначе берем G

else:

sol.append('G')

sG += G

#то же самое для случая, когда сумма А меньше суммы G

else:

if sG + G - sA <= 500:

sol.append('G')

sG += G

else:

sol.append('A')

sA += A

return ''.join(sol)

try:

n = int(input("Введите кол-во яиц: "))

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 1 <= n and n <= 106:

prices = []

while n > 0:

try:

A, G = map(int, input("Введите кол-во денег за покраску для А и для G: ").split())

except ValueError:

print("Некорректный ввод! Повторите попытку!")

else:

if 0 <= A and A <= 1000 and 0 <= G and G <= 1000 and A+G == 1000:

prices.append((A, G))

n -= 1

else:

print("Некорректные данные! Повторите попытку!")

ans = num10(prices)

print(ans)

else:

print("Выход за допустимые границы диапазона значений! Повторите попытку!")