**Содержание**

Оглавление

Введение3

Постанова задачи 14

Ход решения задачи 16

Постановка задачи 28

Ход решения задачи 29

Постановка задачи 311

Ход решения задачи 313

Постановка задачи 415

Ход решения задачи 417

Заключение19

Список литературы20

**Введение**

В данном проекте были разработаны программы на языке Python, направленные на решение конкретных задач. В процессе работы использовались различные библиотеки и инструменты, что способствовало эффективной реализации поставленных целей. Основные технологии, примененные в проекте, включают NumPy для выполнения численных расчетов, что обеспечило высокую производительность и точность вычислений. Для обработки и анализа данных была использована библиотека Pandas, позволяющая удобно манипулировать данными и извлекать из них полезную информацию. Кроме того, для визуализации результатов применялся Matplotlib, что сделало представление данных более наглядным и понятным.

**Задача 1.**

1. Крестьянин и черт

*Максимальное время: 0,1 с.*

*Максимальная память: 4 MB*

Идет крестьянин и плачется: "Эхма! Жизнь моя горькая! Заела нужда совсем! Вот в кармане только несколько монет, да и те сейчас нужно отдать. И как это у других бывает, что на всякие свои деньги они еще деньги получают? Хоть бы кто помочь мне захотел".

Только успел это сказать, как глядь, а перед ним черт стоит и говорит: "Вот видишь этот мост через реку. Стоит тебе перейти через мост, и у тебя будет вдвое больше денег, чем есть. Перейдешь опять, и снова станет вдвое больше. Но за то, что я у тебя деньги удваиваю, после каждого перехода ты мне должен отдавать по K монет".

"Ой ли," - сказал крестьянин -"ну-ка, попробуем". Перешел мост, и деньги у него удвоились. Отдал он черту K монет, перешел мост еще раз, и опять деньги удвоились. Снова отдал крестьянин черту K монет.

Однако после Z переходов и отдач черту по K монет оказалось, что у крестьянина не осталось ни одной монеты.

Требуется определить, сколько комбинаций условий перехода через мост может быть, если известно, что у крестьянина изначально было не более MaxN монет. Комбинацией условий перехода является тройка чисел N, K, Z, где N - начальное количество монет у крестьянина, K - количество монет, отдаваемых черту после каждого перехода, Z - количество переходов. Естественно, что для этой тройки должно выполняться условие, что после Z циклов у крестьянина не должно остаться монет.

Входной файл содержит целое число MaxN - максимальное количество, которое может быть изначально у крестьянина (1 £ MaxN £ 2000000000).

Выходной файл должен содержать одно целое число - количество комбинаций условий перехода через мост.

Пример 1:

Input.txt

10

Output.txt

14

Пример 2:

Input.txt

50

Output.txt

77

Пример 3:

Input.txt

100

Output.txt

157

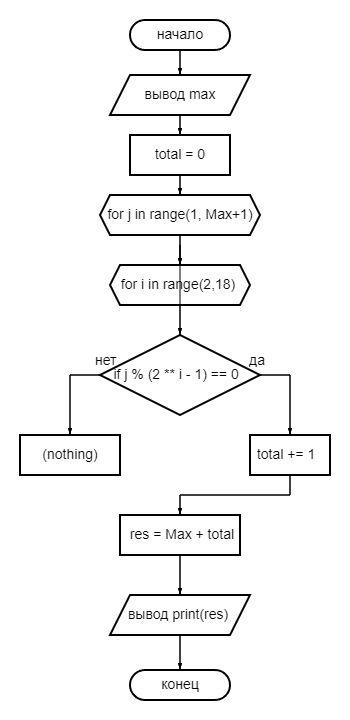
****

Рисунок 1 – Блок-схема алгоритма решения задачи 1

Ход программы представлен на рисунках ниже:

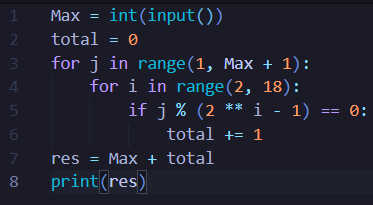


Рисунок 2 – Код программы

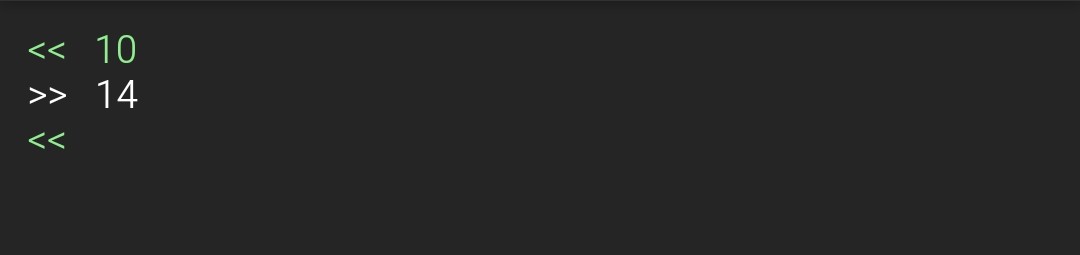


Рисунок 3 – Исходные данные и результат для теста 1

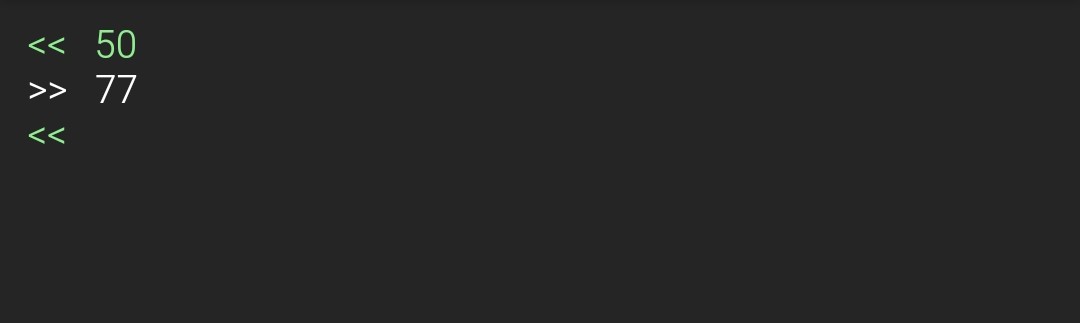


Рисунок 4 – Исходные данные и результат для теста 2



Рисунок 5 – Исходные данные и результат для теста 3

**Задача 2**

2. Отбор в разведку

Из N солдат, выстроенных в шеренгу, требуется отобрать троих в разведку. Для того чтобы сделать это, выполняется следующая операция: если солдат в шеренге больше 3, то шеренга разбивается на две, одна из которых состоит из солдат, стоящие на четных позициях, а вторая – стоящих на нечетных позициях. Эта процедура повторяется для всех полученных шеренг до тех пор, пока в каждой из них не останется 3 или менее солдат. Если солдат осталось трое, то данную группу можно послать в разведку.

Требуется определить, сколько групп по 3 человека может быть сформировано из исходной шеренги.

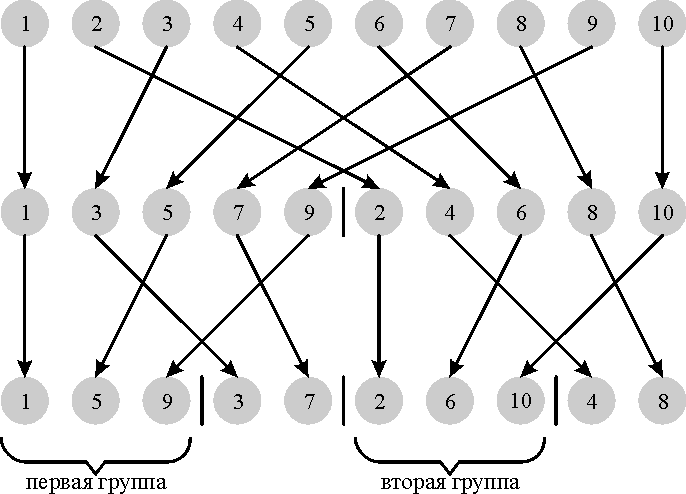


Рисунок 6 - Рисунок к задаче

Входной файл содержит число N – количество солдат в исходной шеренге. (0 < N ≤ 10000000).

Выходной файл должен содержать количество вариантов формирования групп разведки.

Пример 1:

Input.txt

10

Output.txt

2

Пример 2:

Input.txt

10

Output.txt

2

Пример 3:

Input.txt

15

Outputлл.txt

1

Ход программы представлен на рисунках ниже:

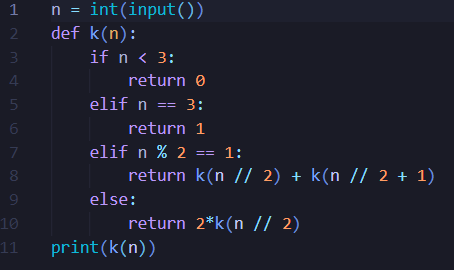
****

Рисунок 7 – Код программы

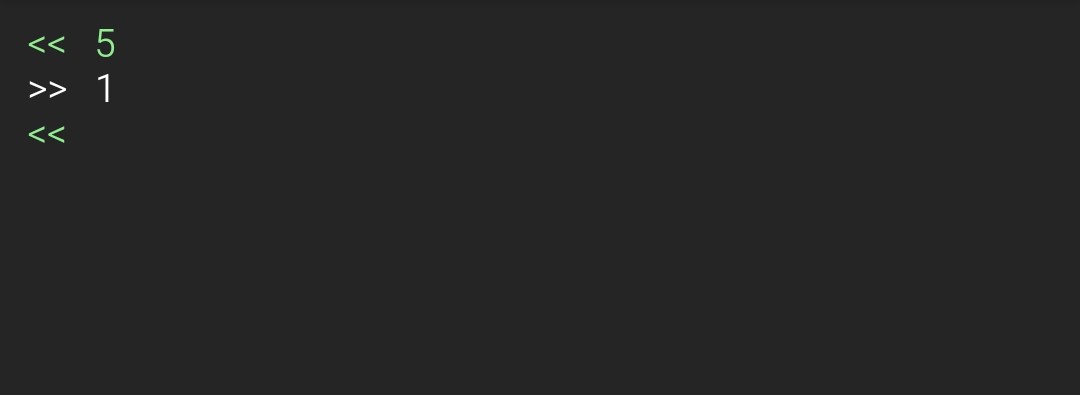


Рисунок 8 – Исходные данные и результат для теста 1



Рисунок 9 – Исходные данные и результат для теста 2



Рисунок 10 – Исходные данные и результат для теста 3

**Задача 3**

3. Упаковки молока

*Максимальное время: 0,2 с.*

*Максимальная память: 16 MB*

На оптовой базе имеется молоко, выпущенное несколькими фирмами.

Молоко каждой фирмы расфасовано в два вида упаковок, представляющих собой параллелепипеды. Для каждого вида упаковки каждой из фирм известна стоимость, которая включает как стоимость материала тары, так и стоимость собственно молока.

Требуется определить фирму, у которой стоимость одного литра собственно молока минимальна, а также эту стоимость.

Примечание

Считать, что материал тары абсолютно тонкий и все плоскости параллелепипеда состоят из одного слоя материала.

Считать, что у двух упаковок одной фирмы стоимость единицы площади материала одинакова.

Считать, что у двух упаковок одной фирмы стоимость одного литра собственно молока одинакова.

Входной файл

Первая строка содержит целое число N - количество фирм (1 £ N £ 100).

Следующие N строк содержат шесть целых чисел Xi1, Yi1, Zi1, Xi2, Yi2, Zi2 - размеры двух видов упаковок i-ой фирмы в сантиметрах (0 < Xi1, Yi1, Zi1, Xi2, Yi2, Zi2 £ 100; 1 £ i £ N), а также два вещественных числа Ci1 и Ci2 - стоимости первой и второй упаковок соответственно у i-ой фирмы в рублях (0 < Ci1, Ci2 £ 1000.0). В стоимости упаковок включаются как стоимость материала тары, так и стоимость собственно молока.

Выходной файл

Должен содержать одну строку, состоящую из целого и вещественного чисел, разделенных пробелом - номер фирмы, у которой стоимость одного литра собственно молока минимальна, а также эту стоимость в рублях (стоимость выводить с двумя знаками после запятой).

Если имеется несколько фирм с одинаковой минимальной стоимостью собственно молока, то вывести ту из них, номер которой минимален.

Пример 1:

Input.txt

2

10 10 5 10 10 10 12.23 20.12

5 15 20 7 8 9 43.28 16.99

Output.txt

2 4.17

Пример 2:

Input.txt

2

1 2 3 4 5 6 7 8

2 3 4 5 6 7 8 9

Output.txt

1 -490.87

Пример 2:

Input.txt

1

2 4 7 1 3 5 6 7

Output.txt

1 -382.88

Ход программы представлен на рисунках ниже:

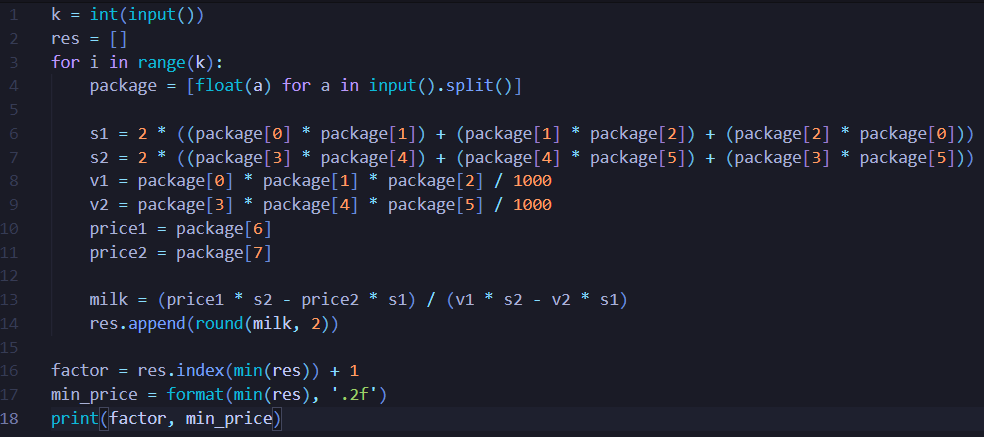


Рисунок 11 – Код программы



Рисунок 12 – Исходные данные и результат для теста 1



Рисунок 13 – Исходные данные и результат для теста 2



Рисунок 14 – Исходные данные и результат для теста 3

**Задача 4**

4. Обмен денег

Известно, что в разных странах могут использоваться разные денежные системы. В каждой денежной системе имеется несколько денежных единиц (например, в России их две: рубль и копейка), между которыми введены некоторые соотношения (например, 1 рубль = 100 копеек).

В разных странах могут быть разные традиции, предрассудки, суеверия. Будем считать, что в определенных странах считаются несчастливыми некоторые числа, поэтому они пропускаются при счете (например, если число 13 считать несчастливым, то значение 15 копеек в реальности соответствует 14 денежным единицам типа "копейка").

Требуется для заданной суммы денег в одной из денежных систем определить ее эквивалент в другой денежной системе.

Примечание: считать, что соотношение между младшими (заданными последними) денежными единицами любых денежных систем 1 : 1.

Входной файл

Первая строка содержит целое число Nисх - количество денежных единиц в исходной денежной системе (1  Nисх  4), а также Nисх-1 целых чисел Ci - соотношения между денежными единицами в исходной денежной системе, причем i-е число обозначает реальное количество (i+1)-ых денежных единиц в i-ой денежной единице (1  i  Nисх-1; 2  Сi  100). Числа в строке разделены пробелами.

Во второй строке дано целое число Kисх - количество несчастливых чисел в стране с исходной денежной системой, а также Kисх целых чисел Aj - сами несчастливые числа (0  Kисх  100; 1  j  Kисх;1  Aj  1000). Все числа Aj уникальные. Числа в строке разделены пробелами.

Третья строка содержит целое число Nкон - количество денежных единиц в конечной денежной системе (1  Nкон  4), а также Nкон-1 целых чисел Dp - соотношения между денежными единицами в конечной денежной системе, причем p-е число обозначает реальное количество (p+1)-ых денежных единиц в p-ой денежной единице (1  p  Nкон-1; 2  Dp  100). Числа в строке разделены пробелами.

В четвертой строке дано целое число Kкон - количество несчастливых чисел в стране с конечной денежной системой, а также Kкон целых чисел Bm - сами несчастливые числа (0  Kкон  100; 1  m  Kкон;1  Bm  1000). Все числа Bm уникальные. Числа в строке разделены пробелами.

В пятой строке дано Nисх целых чисел Ei, разделенных пробелами - представление денежной суммы в исходной денежной системе, причем 0  E1  200; 0  Fi < Ci-1, где Fi - реальное количество денежных единиц для Ei (2  i  Nисх).

Выходной файл

Должен содержать одну строку, состоящую из Nкон целых чисел Sp, разделенных пробелами - представление денежной суммы в конечной денежной системе, причем 0  Tp < Dp-1, где Tp - реальное количество денежных единиц для Sp (2  p  Nкон).

Пример 1:

Input.txt

2 100

1 13

2 10

3 5 22 7

1 42

Output.txt

14 2

Пример 2:

Input.txt

5 3 1 4 1 5

1 2 3 4 5

3 2 1 2 3

1 2 3

10 20 30 40 50

Output.txt

285 0 0

Пример 3:

Input.txt

3 100 10

2 4 13

2 20

2 5 19

2 1 43 2

Output.txt

102 13

Ход программы представлен на рисунках ниже:

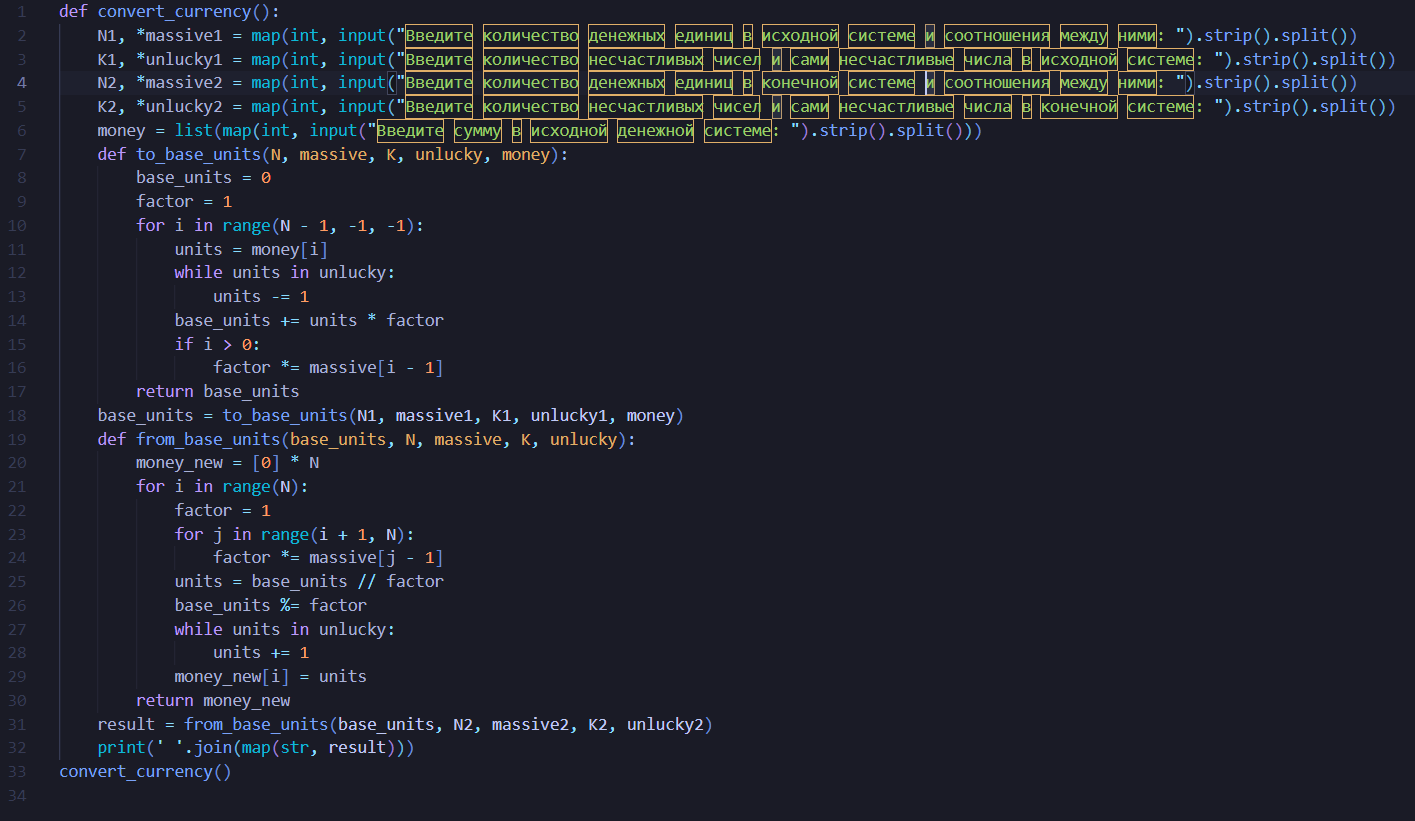


Рисунок 15 – Код программы

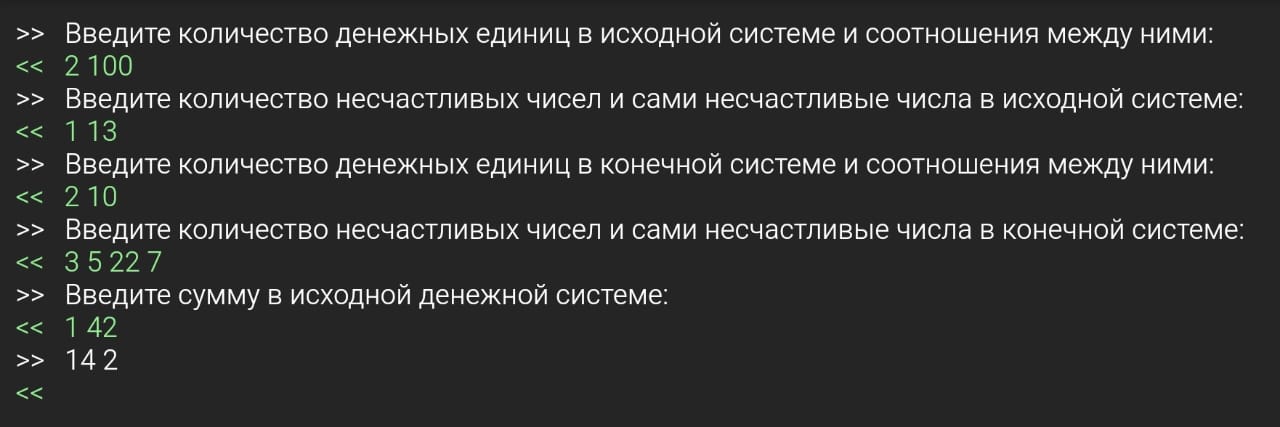


Рисунок 16 – Исходные данные и результат для теста 1

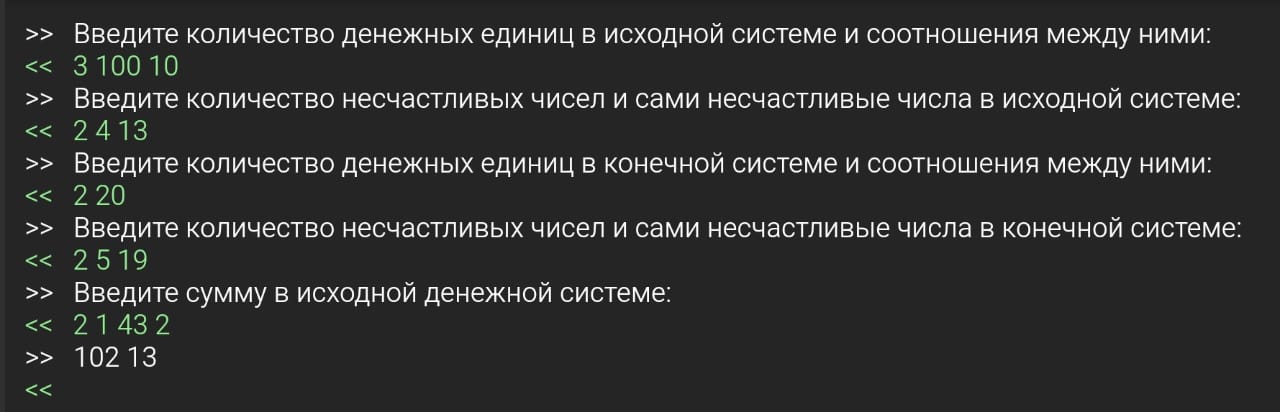


Рисунок 17 – Исходные данные и результат для теста 2

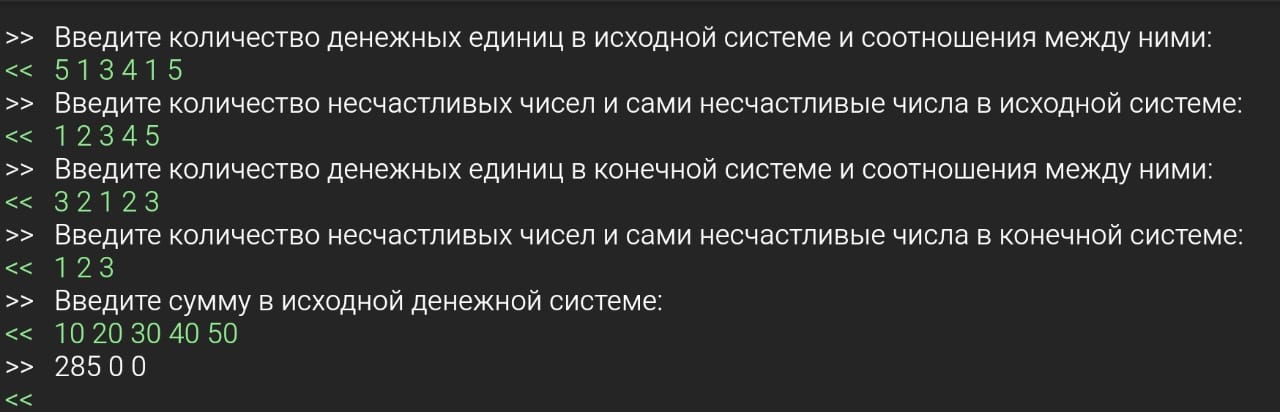


Рисунок 18 – Исходные данные и результат для теста 3

**Заключение**

Разработанные программы на языке Python продемонстрировали свою эффективность в решении поставленных задач. Использование библиотек NumPy, Pandas и Matplotlib обеспечило не только высокую производительность при численных расчетах, но и удобство в обработке и анализе данных. NumPy предоставил мощные инструменты для работы с многомерными массивами, что значительно ускорило вычисления. Pandas, в свою очередь, упростил манипуляции с данными, позволяя легко выполнять фильтрацию, агрегацию и преобразование данных. Визуализация результатов с помощью Matplotlib сделала информацию более доступной и понятной для восприятия, что особенно важно для представления результатов анализа.

Список литературы:

1. МакКинни, Уэс. "Python для анализа данных: работа с библиотеками NumPy, Pandas и Matplotlib". Издательство: ДМК Пресс, 2017.
2. Ханна, Э. "Визуализация данных с помощью Matplotlib". Издательство: Addison-Wesley, 2018.
3. Скиена С. Алгоритмы. Руководство по вычислительным процессам. – СПб.: Питер, 2020.
4. Образовательный сайт «[Школа программирования](https://acmp.ru/asp/do/index.asp?main=task&id_course=2&id_section=14&id_topic=12&id_problem=68)»