САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе №7
по курсу «Алгоритмы и структуры данных»
Тема: Dynamic

Выполнил:
Лазарев Марк Олегович
К3241

Санкт-Петербург
2025 г.

Задачи по варианту

Задача №1. Обмен монет

Задача №2. Наибольшая общая последовательность двух последовательностей

Задачи по варианту

1 задача. Обмен монет

Как мы уже поняли из лекции, не всегда "жадное"решение задачи на обмен монет работает корректно для разных наборов номиналов монет. Например, если доступны номиналы 1, 3 и 4, жадный алгоритм поменяет 6 центов, используя три монеты (4 + 1 + 1), в то время как его можно изменить, используя всего две монеты (3 + 3). Теперь ваша цель - применить динамическое программирование для решения задачи про обмен монет для разных номиналов.

- **Формат ввода / входного файла (input.txt).** Целое число $money$ ($1 \leq money \leq 10^3$). Набор монет: количество возможных монет k и сам набор $coins = \{coin_1, ..., coin_k\}$. $1 \leq k \leq 100$, $1 \leq coin_i \leq 10^3$. Проверку можно сделать на наборе $\{1, 3, 4\}$. Формат ввода: первая строка содержит через пробел $money$ и k; вторая - $coin_1 coin_2 ... coin_k$.
  - ***Вариация 2:*** Количество монет в кассе ограничено. Для каждой монеты из набора $coins = \{coin_1, ..., coin_k\}$ есть соответствующее целое число - количество монет в кассе данного номинала $c = \{c_1, ..., c_k\}$. Если они закончились, то выдать данную монету невозможно.
- **Формат вывода / выходного файла (output.txt).** Вывести одно число – минимальное количество необходимых монет для размена $money$ доступным набором монет $coins$.
- Ограничение по времени. 1 сек.
- Примеры:

input.txt	output.txt
2 3 1 3 4	2

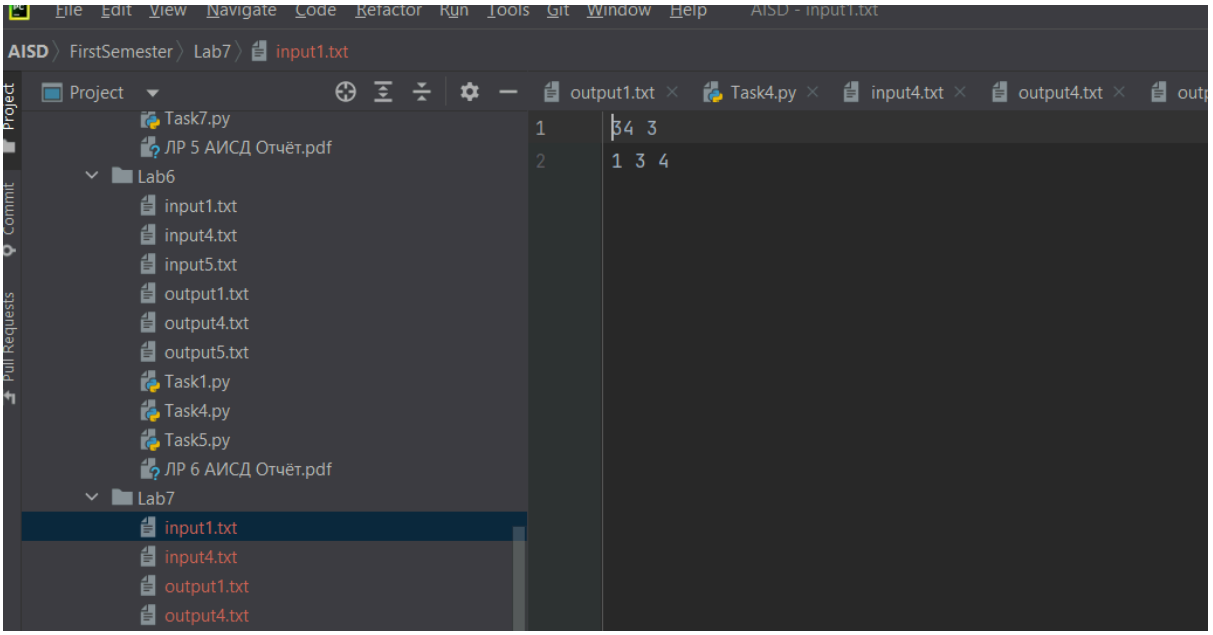
input.txt	output.txt
34 3 1 3 4	9

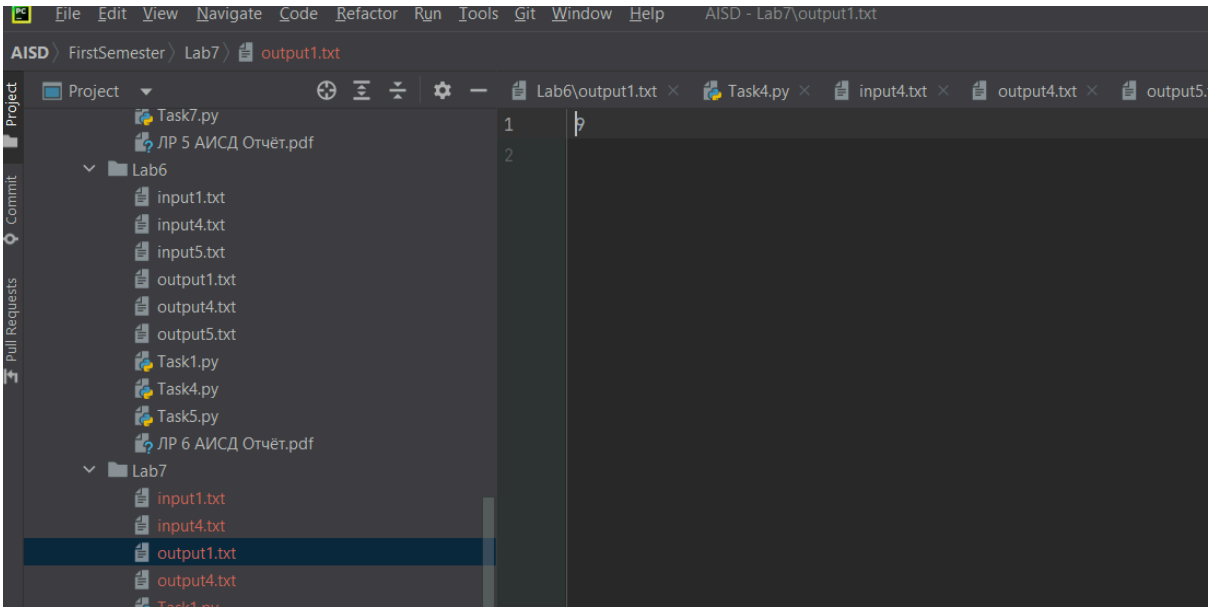
Код программы:

```
import time

def min_coins(money, coins):
    # Инициализация массива для хранения минимального
количества монет
    dp = [float('inf')] * (money + 1)
    dp[0] = 0

    # Заполнение массива dp
    for coin in coins:
        for x in range(coin, money + 1):
            dp[x] = min(dp[x], dp[x - coin] + 1)

    return dp[money] if dp[money] != float('inf') else
-1

def process_input(input_file, output_file):
    with open(input_file, 'r') as infile:
                            money,    k    =    map(int,
infile.readline().strip().split())
                             coins    =    list(map(int,
infile.readline().strip().split()))

    result = min_coins(money, coins)

    with open(output_file, 'w') as outfile:
        outfile.write(f"{result}\n")

def main():
    start_time = time.perf_counter()

    process_input('input1.txt', 'output1.txt')

    end_time = time.perf_counter()
          print(f"Время    выполнения:    {end_time    -
start_time:.9f} секунд")
```

```
if __name__ == "__main__":
    main()
```

Результат работы кода на примерах:





4 задача. Наибольшая общая подпоследовательность двух последовательностей

Вычислить длину самой длинной общей подпоследовательности из двух последовательностей.

Даны две последовательности $A = (a_1, a_2, ..., a_n)$ и $B = (b_1, b_2, ..., b_m)$, найти длину их самой длинной общей подпоследовательности, т.е. наибольшее неотрицательное целое число p такое, что существуют индексы $1 \leq i_1 < i_2 < ... < i_p \leq n$ и $1 \leq j_1 < j_2 < ... < j_p \leq m$ такие, что $a_{i_1} = b_{j_1}, ..., a_{i_p} = b_{j_p}$.

- **Формат ввода / входного файла (input.txt).**
  - Первая строка: n - длина первой последовательности.
  - Вторая строка: $a_1, a_2, ..., a_n$ через пробел.
  - Третья строка: m - длина второй последовательности.

  - Четвертая строка: $b_1, b_2, ..., b_m$ через пробел.
- Ограничения: $1 \leq n, m \leq 100$; $-10^9 < a_i, b_i < 10^9$.
- **Формат вывода / выходного файла (output.txt).** Выведите число p.
- Ограничение по времени. 1 сек.
- Примеры:

input.txt	output.txt
3 2 7 5 2 2 5	2

input.txt	output.txt
1 7 4 1 2 3 4	0

input.txt	output.txt
4 2 7 8 3 4 5 2 8 7	2

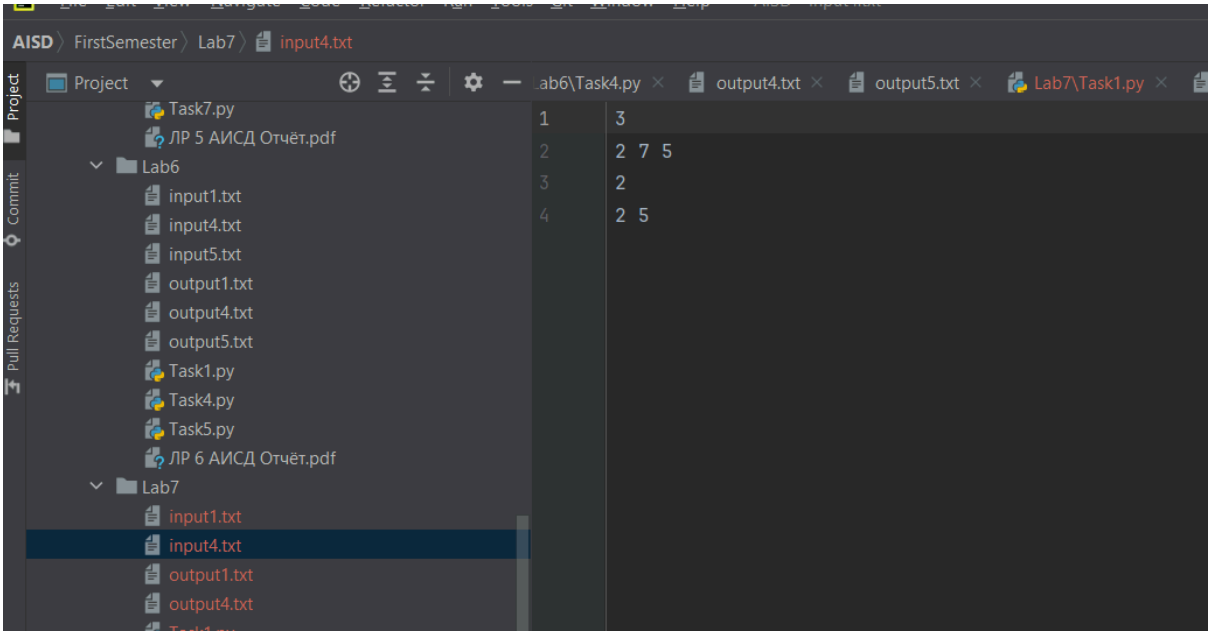
- В первом примере одна общая подпоследовательность – $(2, 5)$ длиной 2, во втором примере две последовательности не имеют одинаковых элементов. В третьем примере - длина 2, последовательности – $(2, 7)$ или $(2, 8)$.

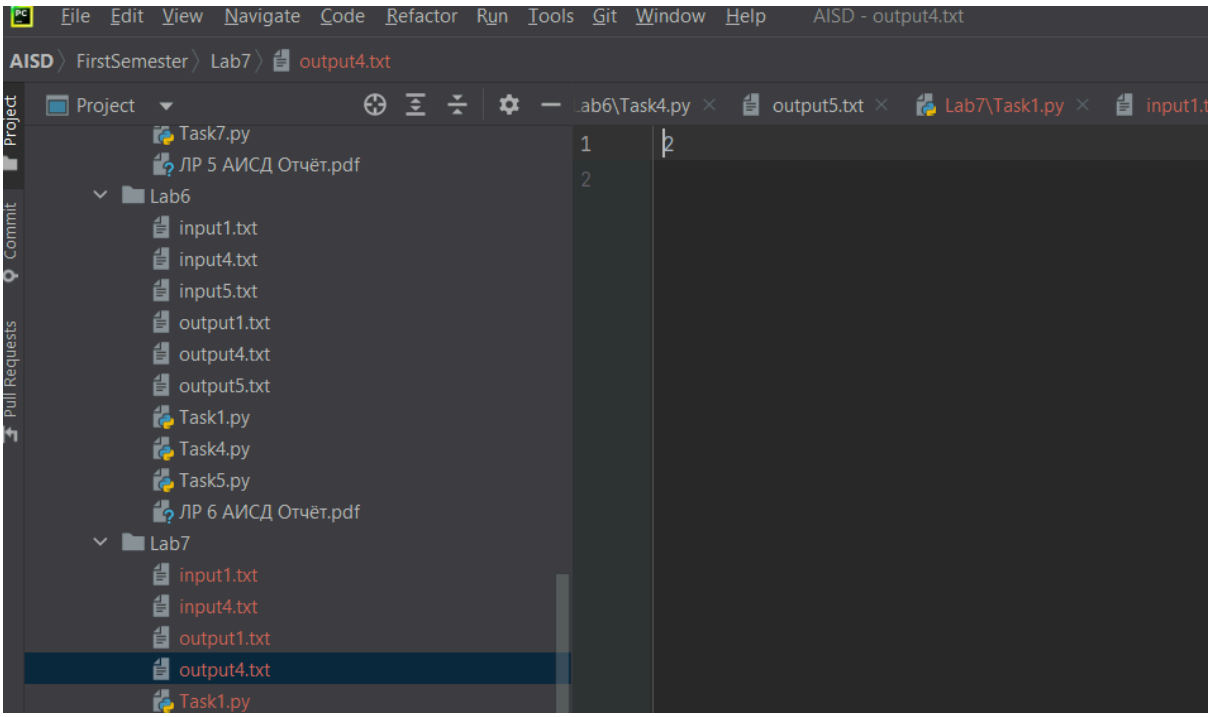
Код программы:

```
def lcs_length(a, b, n, m):
    dp = [[0] * (m + 1) for _ in range(n + 1)]

    for i in range(1, n + 1):
```

```
        for j in range(1, m + 1):
            if a[i - 1] == b[j - 1]:
                dp[i][j] = dp[i - 1][j - 1] + 1
            else:
                dp[i][j] = max(dp[i - 1][j], dp[i][j -
1])
    return dp[n][m]

with open("input4.txt", "r") as file:
    n = int(file.readline().strip())
                        a       =       list(map(int,
file.readline().strip().split()))
    m = int(file.readline().strip())
                        b       =       list(map(int,
file.readline().strip().split()))

result = lcs_length(a, b, n, m)

with open("output4.txt", "w") as file:
    file.write(f"{result}\n")
```

Результат работы кода на примерах:




File Edit View Navigate Code Refactor Run Tools Git Window Help
AISD - output4.txt
AISD
FirstSemester
Lab7
output4.txt
Project
Commit
Pull Requests
Task7.py
ЛР 5 АИСД Отчёт.pdf
Lab6
input1.txt
input4.txt
input5.txt
output1.txt
output4.txt
output5.txt
Task1.py
Task4.py
Task5.py
ЛР 6 АИСД Отчёт.pdf
Lab7
input1.txt
input4.txt
output1.txt
output4.txt
Task1.py
output5.txt
Lab7\Task1.py
1
2