Федеральное государственное автономное

образовательное учреждение высшего образования

«Омский государственный технический университет»

Кафедра "Информатика и вычислительная техника"

**ОТЧЕТ**

**ЛАБОРАТОРНАЯ РАБОТА №1**

|  |  |
| --- | --- |
|  | Выполнил: |
|  | студент группы ПИН-222 Сорокина С.Д. |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (дата, подпись) |
|  | Проверил: |
|  | ассистент Елисеенко Ю.А. |
|  |  |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (дата, подпись) |

Омск 2024

**ОПИСАНИЕ ЗАДАЧИ**

Задача 1.4 (Австралийское голосование)

Австралийские бюллетени требуют, чтобы избиратели расположили всех кандидатов в порядке предпочтения. Первоначально учитывается только первый кандидат из получившегося списка, и если один из кандидатов набрал более 50% голосов, то он считается избранным. Тем не менее, если ни один из кандидатов не набрал более 50% голосов, все кандидаты с наименьшим числом голосов выбывают. Бюллетени, засчитанные в пользу этих кандидатов, засчитываются в пользу не выбывшего кандидата, который следующим идет по порядку предпочтения. Этот процесс исключения самых слабых кандидатов и пересчет их бюллетеней в пользу следующего по порядку предпочтения, не выбывшего кандидата продолжается до тех пор, пока один из кандидатов не наберет более 50% голосов или пока у всех кандидатов не окажется одинаковое число голосов.

Задача 2.4 (По порядку)

В Большом городе мною казино. В одном из них дилер жульничает. Она довела до совершенства несколько перетасовочных трюков; каждый трюк меняет порядок карт одним и тем же образом, когда бы он ни был использован. Простой пример - это трюк «нижней карты», при котором нижняя карта переносится на верх колоды. Используя различные комбинации известных ей трюков, нечестный дилер может расположить карты практически в любом желаемом порядке.

Вы были наняты менеджером службы безопасности, чтобы поймать этого дилера. Вам дали список всех трюков, используемых дилером, а также список визуальных подсказок, благодаря которым вы можете определить, какой трюк она использует в любой данный момент времени. Ваша задача состоит в том, чтобы предсказать порядок карт после последовательности трюков.

Стандартная игральная колода состоит из 52 карт, четыре масти по 13 значений каждая. Значения карт: 2, 3. 4, 5, б, 7, 8, 9, 10, Валет (Jack), Дама (Queen), Король (King), Туз (Асе). Названия мастей: Трефы (Clubs), Бубны (Diamonds), Червы (Hearts), Пики (Spades). Определенная карта в колоде может быть однозначно определена по ее значению и масти и обычно обозначается <значение> of <масть>. Например, «9 of Hearts» или «King of Spades». По традиции карты в новой колоде идут по мастям в алфавитном порядке, а потом по значению в порядке, описанном выше.

**БЛОК-СХЕМЫ**

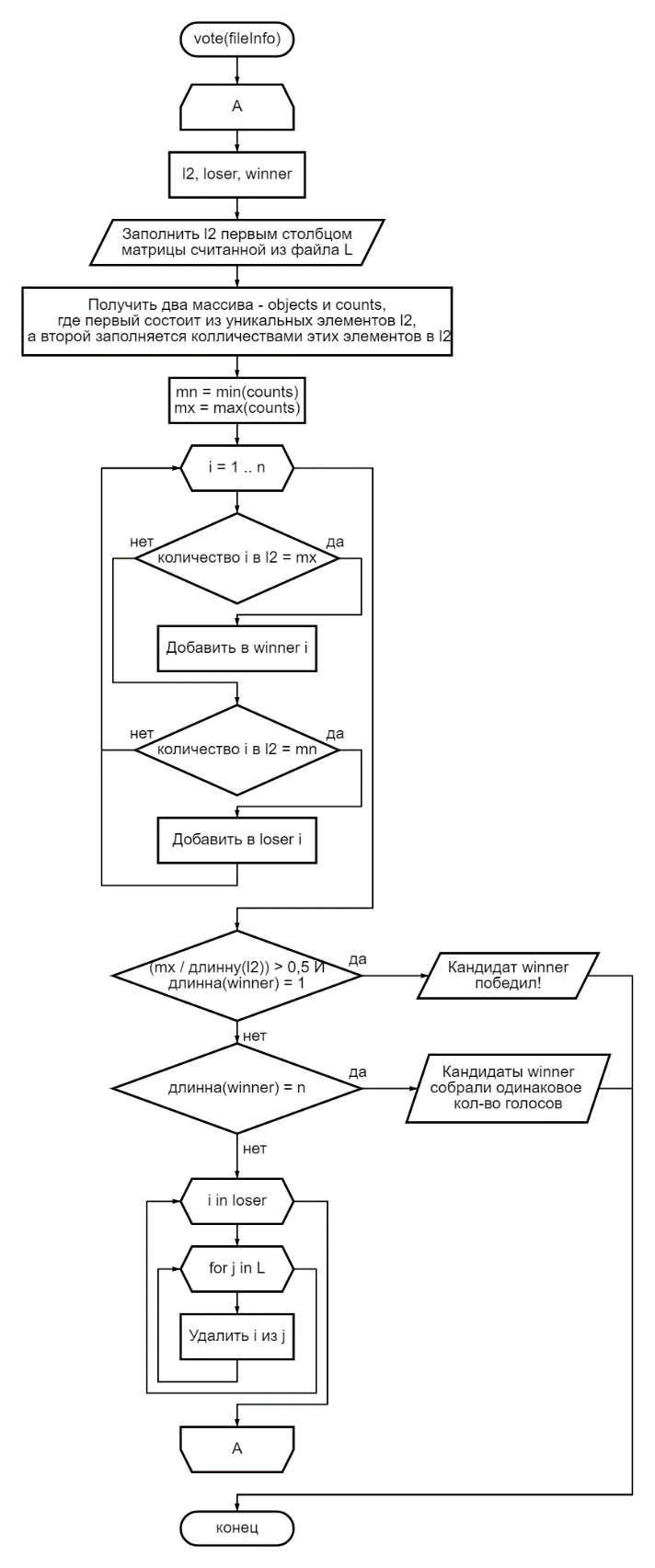
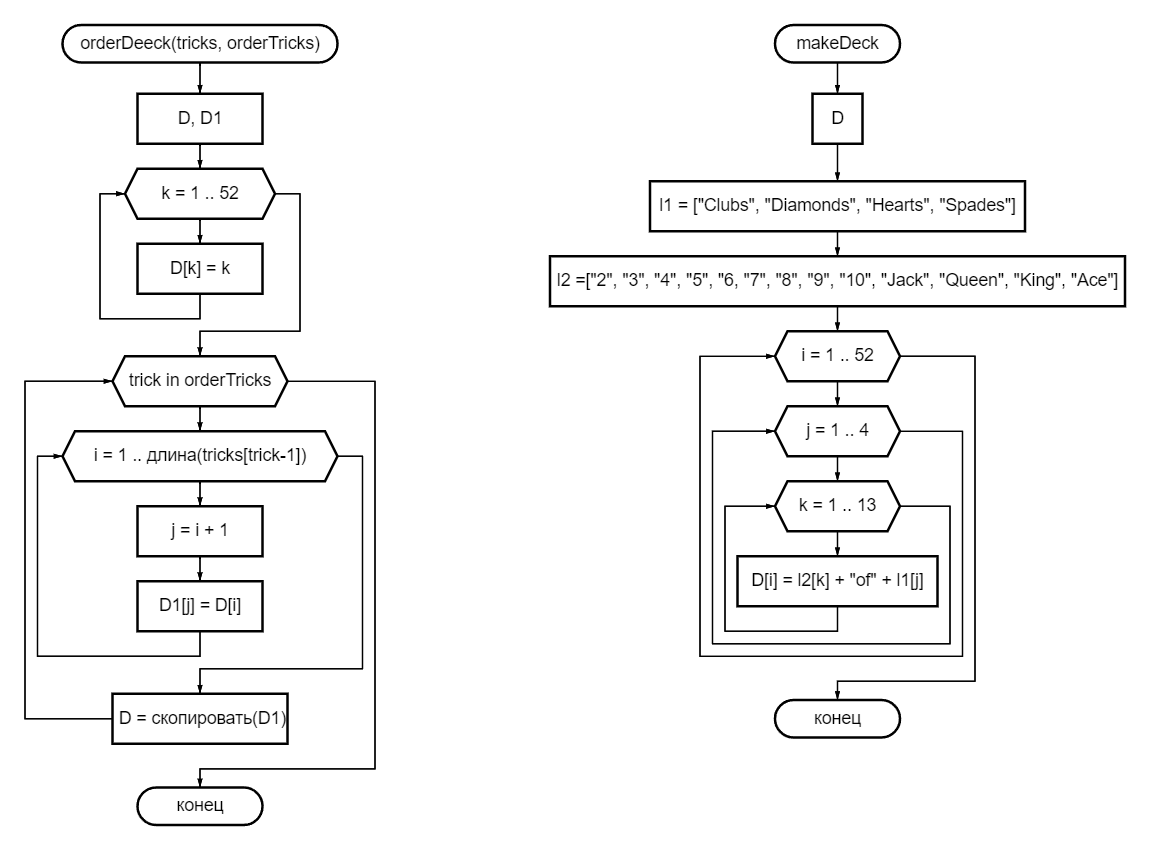
**** Блок-схема решения задачи 1.4 представлена на рисунке 1. Функция vote принимает на вход список fileInfo содержащий в себе количество участников n, количество записей m, словарь с номером и именем кандидата D и, наконец, массив размерностью n на m содержащий в себе запись бюллетеней.

Рисунок 1 – Блох-схема функции vote

Блок-схема решения задачи 2.4 представлена на рисунке 2. Функция makeDeck() возвращает колоду. Функция orderDeeck принимает переменные tricks и orderTricks. Первый список содержит в себе последовательность карт после трюка, второй же, последовательность проведения трюков над колодой.

Рисунок 2 – Блох-схема функций orderDeeck и makeDeck

**НАБОРЫ ТЕСТОВЫХ ДАННЫХ**

Содержимое файлов для тестирования программ, решающих задачи 1.4 и 2.4 представлены на рисунках 3 и 4 соответственно.

Рисунок 3.1 – Файл для тестирования программы, решающий задачу 1.4

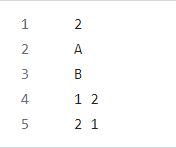
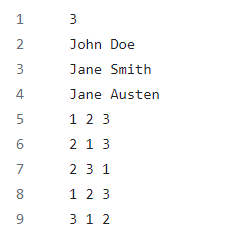


Рисунок 3.2 – Файл для тестирования программы, решающий задачу 1.4



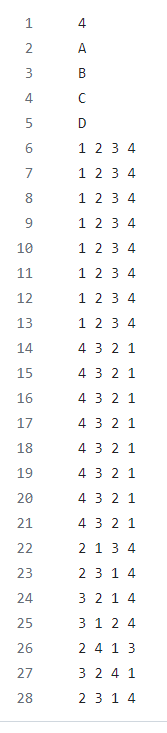


Рисунок 3.3 – Файл для тестирования программы, решающий задачу 1.4

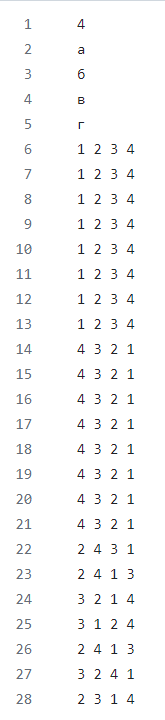


Рисунок 3.4 – Файл для тестирования программы, решающий задачу 1.4

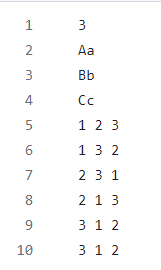


Рисунок 3.5 – Файл для тестирования программы, решающий задачу 1.4

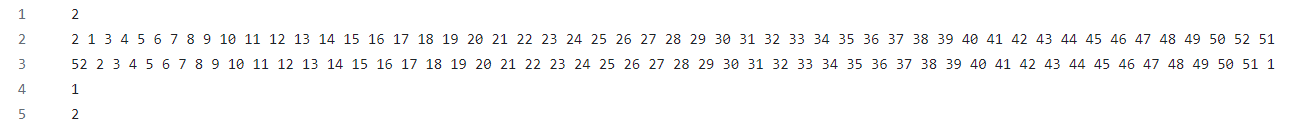


Рисунок 4.1 – Файл для тестирования программы, решающий задачу 2.4

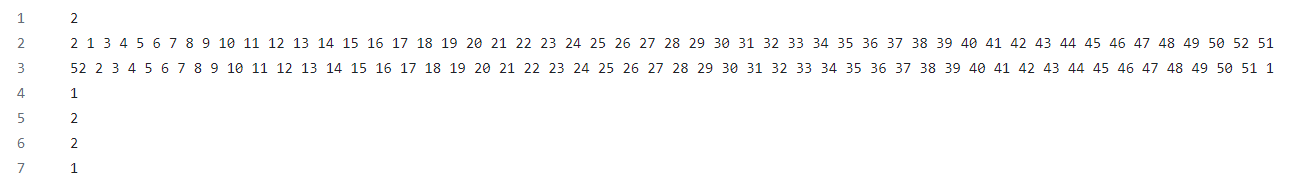


Рисунок 4.2 – Файл для тестирования программы, решающий задачу 2.4

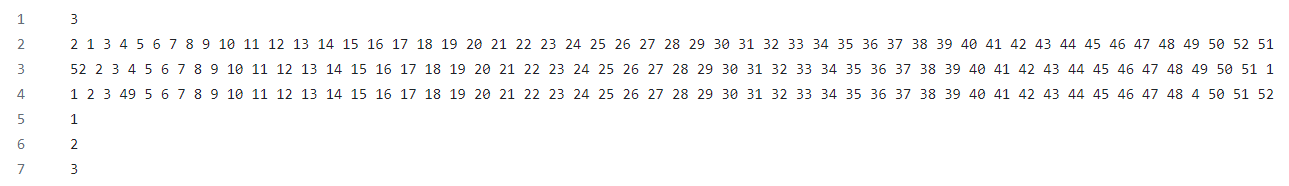


Рисунок 4.3 – Файл для тестирования программы, решающий задачу 2.4

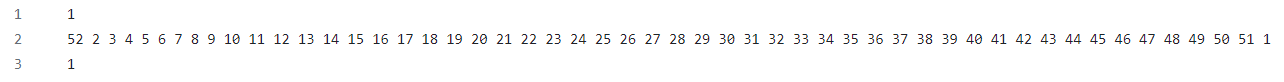


Рисунок 4.4 – Файл для тестирования программы, решающий задачу 2.4

Рисунок 4.5 – Файл для тестирования программы, решающий задачу 2.4

**КОД НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ PYTHON**

Код, решающий задачи 1.4 и 2.4 представлен в листингах 1 и 2.

import numpy as np

def vote(fileInfo):

n = int(fileInfo[0])

m = int(fileInfo[1])

D = fileInfo[2]

L = fileInfo[3]

while (True):

l2 = []

winner = []

loser = []

for i in range(m):

l2.append(L[i][0])

objects, counts = np.unique(l2, return\_counts = True) # ([1, 2, 3, 4], [2, 4, 5, 1])

mn, mx = min(counts), max(counts)

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

if l2.count(i) == mx:

winner.append(i)

if l2.count(i) == mn:

loser.append(i)

if (mx / len(l2) > 0.5) and (len(winner) == 1):

print("Кандидат", \*[D[i] for i in winner], "побеждает!")

break

elif len(winner) == n:

print("Кандидаты", \*[D[i] for i in winner], "собрали одинаковое кол-во голосов!")

break

else:

for i in loser:

for j in L:

j.remove(i)

def fileRead(text):

f = open(text, 'r', encoding = "UTF-8")

n = f.readline()

D = {}

for i in range(int(n)):

s = f.readline()

D[i + 1] = s.strip()

L = []

m = 0

while True:

line = f.readline()

if not line:

break

l = []

l = [int(i.strip()) for i in line.split(' ')]

m = m + 1

L.append(l)

fileInfo = [n, m, D, L]

return(fileInfo)

print("Тест 1")

text = 'text.txt'

fileInfo = fileRead(text)

vote(fileInfo)

print("Тест 2")

text = 'text2.txt'

fileInfo = fileRead(text)

vote(fileInfo)

print("Тест 3")

text = 'text3.txt'

fileInfo = fileRead(text)

vote(fileInfo)

print("Тест 4")

text = 'text4.txt'

fileInfo = fileRead(text)

vote(fileInfo)

print("Тест 5")

text = 'text5.txt'

fileInfo = fileRead(text)

Листинг 1 – Код программы, решающий задачу 1.4

vote(fileInfo)

def readFile(text):

f = open(text, 'r', encoding="UTF-8")

n = int(f.readline()) #считываем кол-во трюков

tricks = list()

orderTricks = list()

for i in range(n): #проходим n раз по файлу и записываем в каком порядке распологаются карты после n-го трюка (перед каждым n-ым трюков колода карт находится в изначальном положении)

tricks.append(f.readline())

while True:

line = f.readline() #записываем порядок выполнения трюков пока не кончится файл

if not line:

break

orderTricks.append(int(line))

return tricks, orderTricks

def orderDeeck(tricks, orderTricks):

from copy import copy

D, D1 = dict(), dict()

D = {k: k for k in range(1, 52 + 1)} #заполняем ключи и значения цифрами по порядку 1:1 2:2 3:3 и тд

for trick in orderTricks: #проходимся по последовательности трюков

for j, i in enumerate(tricks[int(trick)-1].split(), start = 1): #производим перестановку, j это цифры по порядку, i индекс карты

i = int(i)

D1[j] = D.get(i) #заполняем D1 в формате j номер по порядку: индекс карты

D = copy(D1)

return D

def makeDeck() -> dict: #создаем словарь порядковый номер: название карты

from itertools import product

return {k: f"{v[1]} of {v[0]}" for k, v in enumerate(product(["Clubs", "Diamonds", "Hearts", "Spades"], list(range(2, 10 + 1)) + ["Jack", "Queen", "King", "Ace"]), start = 1)}

print("Начальная колода")

print(makeDeck().items())

print("Тест 1")

print(\*[makeDeck()[i[1]] for i in orderDeeck(\*readFile("text.txt")).items()], sep = "\n")

print("Тест 2")

print(\*[makeDeck()[i[1]] for i in orderDeeck(\*readFile("text2.txt")).items()], sep = "\n")

print("Тест 3")

print(\*[makeDeck()[i[1]] for i in orderDeeck(\*readFile("text3.txt")).items()], sep = "\n")

print("Тест 4")

print(\*[makeDeck()[i[1]] for i in orderDeeck(\*readFile("text4.txt")).items()], sep = "\n")

print("Тест 5")

print(\*[makeDeck()[i[1]] for i in orderDeeck(\*readFile("text5.txt")).items()], sep = "\n")

Листинг 2 – Код программы, решающий задачу 2.4

**РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММ**

Скриншоты с результатами работы программ представлены в рисунках 5 и 6.

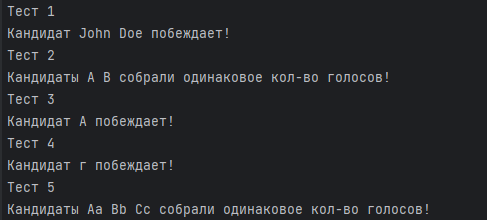


Рисунок 5 – Результат работы программы, решающей задачу 1.4

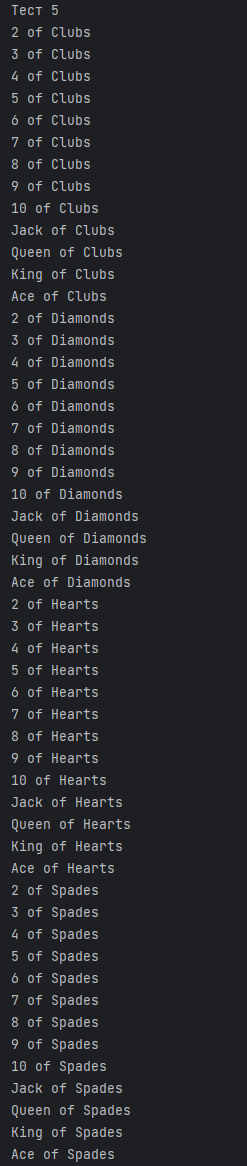
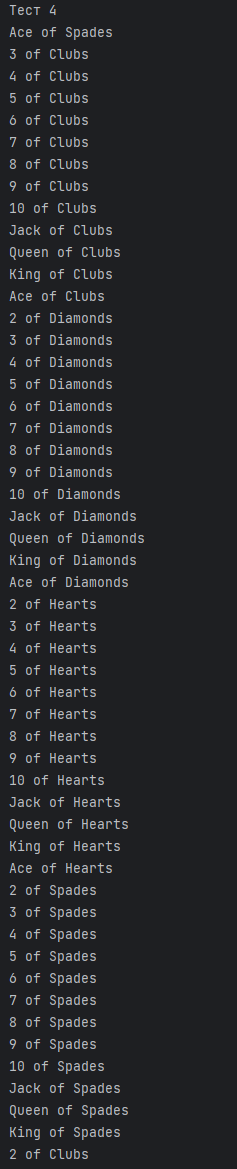
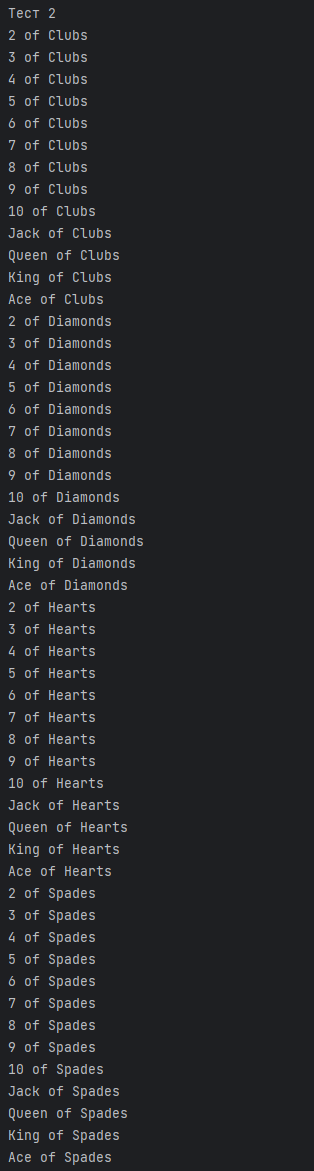
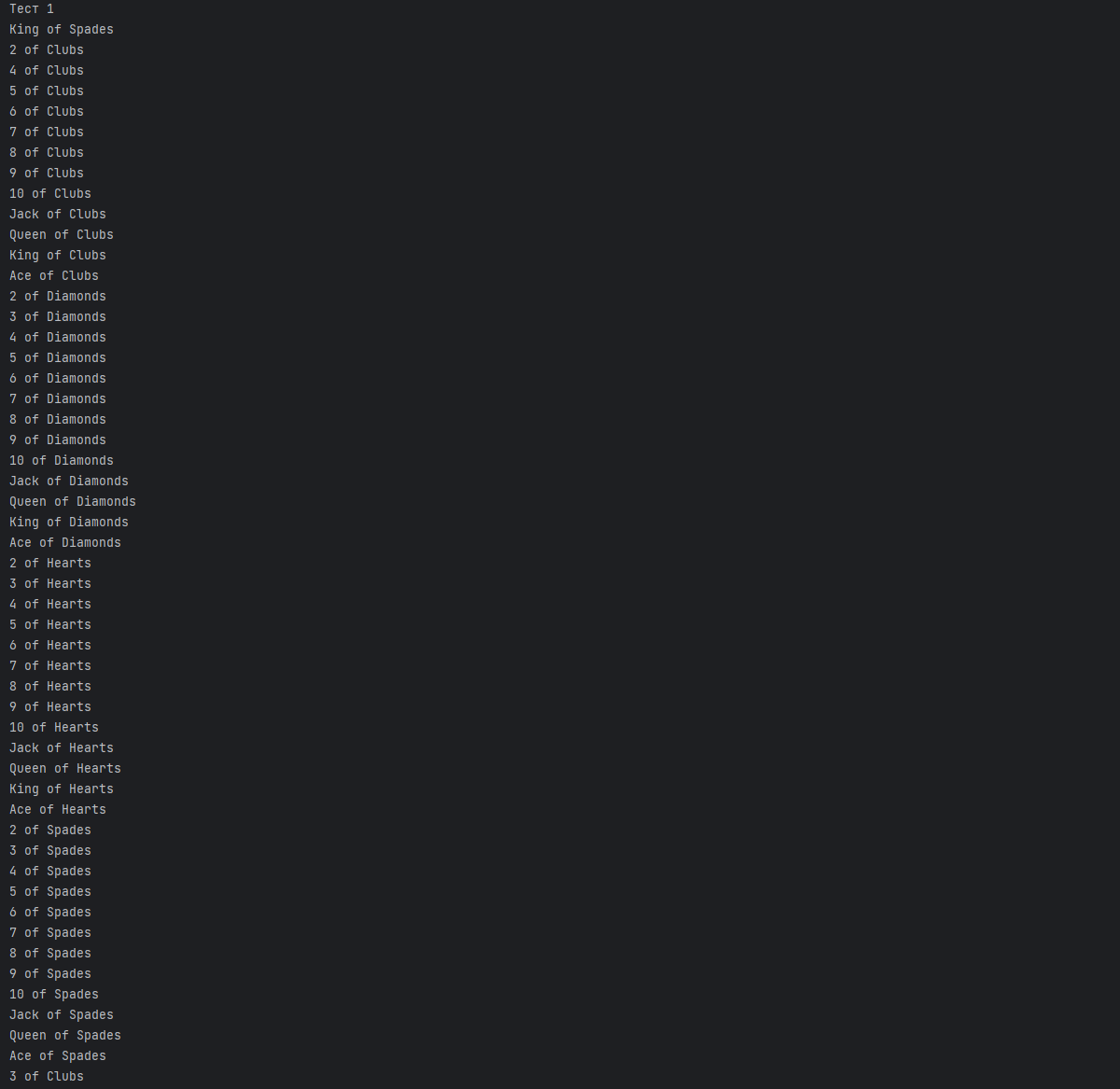


Рисунок 6 – Результат работы программы, решающей задачу 2