

Университет ИТМО

Факультет инфокоммуникационных технологий

1 курс

Лабораторная работа №2-1

Выполнила:

Чагина Вероника Александровна

группа К3144

Преподаватель:

Харьковская Татьяна Александровна

Дата выполнения: 21.03.2022

Санкт-Петербург

Описание заданий

Вариант 10

Задания: 3,7,9,15,21(вместо 9 выполнила 11)(доп. 2, 5, 10)

Задание 2

Описание задания :

Вы собираетесь поехать в другой город, расположенный в d км от вашего родного города. Ваш автомобиль может проехать не более m км на полном баке, и вы начинаете с полным баком. По пути есть заправочные станции на расстояниях $stop_1, stop_2, \dots, stop_n$ из вашего родного города. Какое минимальное количество заправок необходимо?

Задание 3

Описание задания :

У вас есть n объявлений для размещения на популярной интернет-странице. Для каждого объявления вы знаете, сколько рекламодатель готов платить за один клик по этому объявлению. Вы настроили n слотов на своей странице и оценили ожидаемое количество кликов в день для каждого слота. Теперь ваша цель - распределить рекламу по слотам, чтобы максимизировать общий доход.

Задание 5

Описание задания :

Вы организуете веселый конкурс для детей. В качестве призового фонда у вас есть n конфет. Вы хотели бы использовать эти конфеты для раздачи k лучшим местам в конкурсе с естественным ограничением, заключающимся в том, что чем выше место, тем больше конфет. Чтобы осчастливить как можно больше детей, вам нужно найти наибольшее значение k , для которого это возможно.

Задание 7

Описание задания :

В некоей воинской части есть сапожник. Рабочий день сапожника длится K минут. Заведующий складом оценивает работу сапожника по количеству починенной обуви, независимо от того, насколько сложный ремонт требовался в каждом случае. Дано n сапог, нуждающихся в починке. Определите, какое максимальное количество из них сапожник сможет починить за один рабочий день.

Задание 9

Описание задания :

Алисе в стране чудес попались n волшебных яблок. Про каждое яблоко известно, что после того, как его съешь, твой рост сначала уменьшится на a_i сантиметров, а потом увеличится на b_i сантиметров.

Алиса очень голодная и хочет съесть все n яблок, но боится, что в какой-то момент ее рост s станет равным нулю или еще меньше, и она пропадет совсем. Помогите ей узнать, можно ли съесть яблоки в таком порядке, чтобы в любой момент времени рост Алисы был больше нуля.

Задание 11

Описание задания :

Вам дается набор золотых слитков, и ваша цель - набрать как можно больше золота в свою сумку. Существует только одна копия каждого слитка, и для каждого слитка вы можете либо взять его, либо нет (т.е. вы не можете взять часть слитка).

Задание 15

Описание задания :

Дана строка, составленная из круглых, квадратных и фигурных скобок. Определите, какое наименьшее количество символов необходимо удалить из этой строки, чтобы оставшиеся символы образовывали правильную скобочную последовательность.

Задание 21

Описание задания :

Пете необходимо решить следующую задачу: сможет ли игрок, обладая набором из N карт, отбить M карт, которыми под него сделан ход?

Как известно, в «Дурака» играют колодой из 36 карт. В Петиней программе каждая карта представляется в виде строки из двух символов, где первый символ означает ранг ('6', '7', '8', '9', 'T', 'J', 'Q', 'K', 'A') карты, а второй символ означает масть ('S', 'C', 'D', 'H').

Ранги перечислены в порядке возрастания старшинства.

Решения и исходный код

Задание 2

```
file = open('input.txt')
need = int(file.readline())
tank = int(file.readline())
n = int(file.readline())
stations = list(map(int, file.readline().split()))
file.close()
```

```
def getFillsCount(need, tank, stations):
    stations.append(need)
    curr = 0
    count = 0
    for i in range(len(stations) - 1):
        if stations[i] - curr > tank:
            return -1
        if (stations[i + 1] - curr) > tank:
            curr = stations[i]
            count += 1
    if stations[len(stations) - 1] - curr < tank:
        return count
    else:
        return -1
```

```
count = getFillsCount(need, tank, stations)
out = open('output.txt', 'w')
out.write(str(count))
print(str(count))
out.close()
```

Задание 3

```
with open('input.txt') as file:
    n = int(file.readline())
    a = list(map(int, file.readline().split()))
    b = list(map(int, file.readline().split()))
```

```
a = list(sorted(a))
b = list(sorted(b))
```

```
summ = 0
for i in range(n):
    summ += a[i] * b[i]
```

```
out = open("output.txt", "w")
out.write(str(summ))
out.close()
```

Задание 5

```
with open('input.txt') as file:
    n = int(file.readline())

ans = []
cur_sum = 0
i = 1

while cur_sum + i <= n:
    ans.append(i)
    cur_sum += i
    i += 1
if cur_sum != n:
    print(ans[-1])
    ans[-1] += n - cur_sum

print(len(ans))
print(*ans)
```

Задание 7

```
with open('input.txt') as file:
    K, n = map(int, file.readline().split())
    t = list(map(int, file.readline().split()))

t = sorted(t)
i = 0
cur_sum = 0

while i < len(t) and cur_sum + t[i] <= K:
    print(cur_sum, t[i], i)
    cur_sum += t[i]
    i += 1
print(i)
```

Задание 11

```
with open('input.txt') as file:
    W, n = map(int, file.readline().split())
    A = list(map(int, file.readline().split()))

F = [1] + [0] * W
F_new = F[:]

for j in range(len(A)):
    for i in range(A[j], W + 1):
        print(i, A[j])
        if F[i - A[j]] == 1:
            F_new[i] = 1
    F = F_new[:]

i = W
while F[i] == 0:
    i -= 1
print(i)
```

Задание 15

```
import math

with open('input.txt') as file:
    s = file.readline()
n = len(s)

dp = [[0 for l in range(n)] for k in range(n)]
ep = [[0 for l in range(n)] for k in range(n)]

for i in range(n):
    for j in range(n):
        if i == j:
            dp[i][j] = 1

for right in range(n):
    for left in range(right, -1, -1):
        if left == right:
            dp[left][right] = 1
        else:
            minimal = math.inf
            mink = -1
            if s[left] == '(' and s[right] == ')' \
                or s[left] == '[' and s[right] == ']' \
                or s[left] == '{' and s[right] == '}':
                minimal = dp[left + 1][right - 1]

            for k in range(left, right):
                if minimal > dp[left][k] + dp[k + 1][right]:
                    minimal = dp[left][k] + dp[k + 1][right]
                    mink = k
            dp[left][right] = minimal
            ep[left][right] = mink

def restoring_response(left1, right1):
    temp = right1 - left1 + 1
    if dp[left1][right1] == temp:
        return

    if dp[left1][right1] == 0:
        print(s[left1:right1 + 1], end="")
        return

    if ep[left1][right1] == -1:
        print(s[left1], end="")
        restoring_response(left1 + 1, right1 - 1)
        print(s[right1], end="")
        return

    restoring_response(left1, ep[left1][right1])
    restoring_response(ep[left1][right1] + 1, right1)

restoring_response(0, n - 1)
```

Задание 21

```
def CardValues(card):
    arr = ["6", "7", "8", "9", "T", "J", "Q", "K", "A"]
    return arr.index(card)

file = open("input.txt")

myCount, enemyCount, trump = file.readline().split()
myCount = int(myCount)
enemyCount = int(enemyCount)
myCards = file.readline().split()
enemyCards = file.readline().split()

myCardsSuit = {"S": [], "C": [], "D": [], "H": []}
enemyCardsSuit = {"S": [], "C": [], "D": [], "H": []}
#создаем массивы, в которых для каждой масти будут записаны карты, которые есть в наборах
for card in myCards:
    myCardsSuit[card[1]].append(card[0])
for card in enemyCards:
    enemyCardsSuit[card[1]].append(card[0])
#сортируем карты по их старшинству
cardSuit = ["S", "C", "D", "H"]
for suit in cardSuit:
    myCardsSuit[suit] = sorted(myCardsSuit[suit], key=lambda x: CardValues(x))
    enemyCardsSuit[suit] = sorted(enemyCardsSuit[suit], key=lambda x: CardValues(x),
reverse=True)
cardSuit.remove(trump)

def Winner():
    for enemyCard in enemyCardsSuit[trump]:
        beaten = False
        for myCard in myCardsSuit[trump]:
            if CardValues(myCard) > CardValues(enemyCard):
                myCardsSuit[trump].remove(myCard)
                beaten = True
                break
        if not beaten:
            return False

    for suit in cardSuit:
        for enemyCard in enemyCardsSuit[suit]:
            beaten = False
            for myCard in myCardsSuit[suit]:
                if CardValues(myCard) > CardValues(enemyCard):
                    myCardsSuit[suit].remove(myCard)
                    beaten = True
                    break
            if not beaten:
                for myTrump in myCardsSuit[trump]:
                    beaten = True
                    myCardsSuit[trump].remove(myTrump)
                    break
                if not beaten:
                    return False
```

```
    return True
```

```
result = Winner()
```

```
out = open("output.txt", "w")
```

```
if result:
```

```
    out.write("YES")
```

```
else:
```

```
    out.write("NO")
```

```
out.close()
```


Вывод

В данной лабораторной работе изучается динамическое программирование.

Динамическое программирование — это подход к решению задач, который основывается на том, что исходная задача разбивается на более мелкие подзадачи, которые проще решить. И потом решения этих подзадач можно использовать для решения исходной задачи.