Лабораторная работа № 2

Вариант 13

В отчете указать: что такое ДП, принципы ДП и т.п.

**Динамическое программирование** – это способ решения сложных задач путем сведения их к более простым задачам того же типа.

Задание №1

Постановка задачи:

Кузнечик. На числовой прямой в точке 0 находится кузнечик. Он умеет прыгать только вперед, делая прыжки от 1 до L делений, т.е. из точки K он может переместиться в точку K+1, K+2, ... , K+L за один прыжок. Определите, сколько существует различных путей кузнечика, чтобы попасть из точки 0 в точку N.

Тесты: N=6 L=9; 32

N=27 L=13; 67047424

N=14, L=10; 8172

Код:

def Kyz(A, N, L):

A[0] = 1

A[1] = 1

for i in range(2, N + 1):

A[i] = A[i - 1]

j = 2

while j <= i and j <= L:

A[i] += A[i - j]

j += 1

return A

Fin = open('input.txt', "r+")

s = Fin.readline().split()

if not s:

print('Файл пустой')

elif int(s[0]) == 0:

print('0 элементов Матрицы')

else:

N = int(s[0]) # Элементы

L = int(s[1]) # Деления

A = [0]\*(N+1)

A = Kyz(A, N, L)

print(A)

Fin.write("\n {}".format(A[N]))

Fin.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 | 6 9 | 32 | 32 |
| 2 | 27 13 | 67047424 | 67047424 |
| 3 | 14 10 | 8172 | 8172 |

Задание №2

Постановка задачи:

Черепашка. На квадратной доске расставлены целые неотрицательные числа, каждое из которых не превосходит 100. Черепашка, находящаяся в левом верхнем углу, мечтает попасть в правый нижний. При этом она может переползать только в клетку справа или снизу и хочет, чтобы сумма всех чисел, оказавшихся у нее на пути, была бы минимальной. Определить эту сумму. Ввод организовать при помощи текстового файла. Формат входных данных: в первой строке входного файла записано число N - размер доски (1<N<30). Далее следует N строк, каждая из которых содержит N целых чисел, представляющих доску.

7

5 9 6 9 9 8 5

9 9 4 0 8 1 5

1 7 2 7 8 8 4

Ответ: 57

Код:

def Matr(Mat, N, M):

for i in range(N):

Mat.append([0] \* M)

s = Fin.readline().split()

for j in range(M):

Mat[i][j] = float(s[j])

print(A[i])

return Mat

def CHER(A, N):

for i in range(1, N):

A[0][i] += A[0][i-1]

for i in range(1, N):

A[i][0] += A[i-1][0]

for i in range(1, N):

for j in range(1, N):

if A[i-1][j] >= A[i][j-1]:

A[i][j] += A[i][j-1]

else:

A[i][j] += A[i-1][j]

return A

Fin = open('input.txt', "r+")

s = Fin.readline().split()

if not s:

print('Файл пустой')

elif int(s[0]) == 0:

print('0 элементов Матрицы')

else:

N = int(s[0])

A = []

A = Matr(A, N, N)

A = CHER(A, N)

for i in range(N):

Fin.write("\n {}".format(A[i]))

Fin.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 | [Файл](https://drive.google.com/file/d/1aSL8x4waXROiQkRb2k5TMTZcQQ2MT02r/view?usp=sharing) | [5.0, 14.0, 20.0]  [14.0, 23.0, 24.0]  [15.0, 22.0, 24.0] | [5.0, 14.0, 20.0]  [14.0, 23.0, 24.0]  [15.0, 22.0, 24.0] |
| 2 | [Файл](https://drive.google.com/file/d/1une68AKkGPNvDhzmYTIFmpoc5_Qiuedx/view?usp=sharing) | [5.0, 14.0, 20.0, 25.0]  [14.0, 23.0, 24.0, 27.0]  [15.0, 22.0, 24.0, 26.0]  [17.0, 20.0, 26.0, 33.0] | [5.0, 14.0, 20.0, 25.0]  [14.0, 23.0, 24.0, 27.0]  [15.0, 22.0, 24.0, 26.0]  [17.0, 20.0, 26.0, 33.0] |
| 3 | [Файл](https://drive.google.com/file/d/1Os19p3fGiSFQqJAfgKpZZzyubIMPgDMd/view?usp=sharing) | [5.0, 14.0, 20.0, 25.0, 32.0]  [14.0, 23.0, 24.0, 27.0, 35.0]  [15.0, 22.0, 24.0, 26.0, 28.0]  [17.0, 20.0, 26.0, 33.0, 29.0]  [18.0, 20.0, 27.0, 36.0, 37.0] | [5.0, 14.0, 20.0, 25.0, 32.0]  [14.0, 23.0, 24.0, 27.0, 35.0]  [15.0, 22.0, 24.0, 26.0, 28.0]  [17.0, 20.0, 26.0, 33.0, 29.0]  [18.0, 20.0, 27.0, 36.0, 37.0] |

Задание №3

Постановка задачи:

К-ичные числа. Среди чисел в системе счисления с основанием K (2≤K≤10) определить сколько имеется чисел из N (1<N<10) разрядов таких, что в их записи содержится более двух подряд идущих нулей.

(В начале файла пишется числа K и N)

Код:

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 | Файл |  |  |
| 2 | Файл |  |  |
| 3 | Файл |  |  |
| 4 | Файл |  |  |
| 5 | Файл |  |  |

Задание №4

Постановка задачи:

Кузнечик. На числовой прямой в точке 0 находится кузнечик. Он умеет прыгать только вперед на 1, на 2, на 3 или на 4 деления, т.е. из точки K он может переместиться в точку K+1, K+2, K+3 или K+4 за один прыжок. На некоторых делениях числовой оси находятся лягушки, которые поедают кузнечиков. Считая, что лягушки никуда не перемещаются, определите, сколько существует различных безопасных путей кузнечика, чтобы попасть из точки 0 в точку N.

(В файле первой строкой написано N, во второй строке записаны деления, где находятся лягушки в порядке возрастания)

Код:

def Kyz(A, N):

A[0] = 1

A[1] = 1

L = 4

s = Fin.readline().split()

z = 0

for i in range(2, N + 1):

print(s[z], i)

if i == int(s[z]):

A[i] = 0

z += 1

else:

A[i] = A[i - 1]

j = 2

while j <= i and j <= L:

A[i] += A[i - j]

j += 1

return A

Fin = open('input.txt', "r+")

s = Fin.readline().split()

if not s:

print('Файл пустой')

elif int(s[0]) == 0:

print('0 элементов Матрицы')

else:

N = int(s[0]) # Элементы

A = [0]\*(N+1)

A = Kyz(A, N)

print(A)

Fin.write("\n {}".format(max(A)))

Fin.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 | [Файл](https://drive.google.com/file/d/1OnDI9mt7raaK_Uy3hAdwbTSOpOXHraWs/view?usp=sharingview?usp=sharingsp=sharing) | 15 | 15 |
| 2 | [Файл](https://drive.google.com/file/d/1z7ChlhSEgGjYAUzZdHHT9I6odbbM2mad/view?usp=sharing) | 3 | 3 |
| 3 | [Файл](https://drive.google.com/file/d/16gVs1MVYUgKLGDdwGhq2Gcz8or5TFm_Q/view?usp=sharing) | 73 | 73 |

Задание №5

Постановка задачи:

Хромой король. На квадратной доске в каждой клетке короля ожидают неприятности в количестве от 0 до 30. Хромой король, находящийся в левом верхнем углу, мечтает попасть в правый нижний угол. При этом он может передвигаться только в клетку справа или снизу и хочет, чтобы сумма всех неприятностей, оказавшихся у него на пути, была бы минимальной. Определить эту сумму и путь, каким образом должен двигаться король. Ввод и вывод организовать при помощи текстовых файлов. Формат входных данных: в первой строке входного файла записано число N - размер доски (1<N<30). Далее следует N строк, каждая из которых содержит N чисел - количество неприятностей в клетках доски. В первую строку выходного файл нужно вывести единственное число: минимальную сумму, а во второй строке вывести путь в виде строки символов, обозначив символом R движение вправо, а символом D - движение вниз.

Код:

def Matr(Mat, N, M):

for i in range(N):

Mat.append([0] \* M)

s = Fin.readline().split()

for j in range(M):

Mat[i][j] = float(s[j])

return Mat

def CHER(A, N):

for i in range(1, N):

A[0][i] += A[0][i-1]

for i in range(1, N):

A[i][0] += A[i-1][0]

for i in range(1, N):

for j in range(1, N):

if A[i-1][j] >= A[i][j-1]:

A[i][j] += A[i][j-1]

else:

A[i][j] += A[i-1][j]

return A

def PUT(A, N, B):

a = N-1

b = N-1

for i in range(2\*(N-1)-1, -1, -1):

if A[a-1][b] < A[a][b-1]:

a -= 1

B[i] = 'D'

elif A[a-1][b] > A[a][b-1]:

b -= 1

B[i] = 'R'

else:

if A[a - 1][b - 1] < A[a - 2][b] and A[a - 1][b - 1] < A[a][b - 2]:

a -= 1

B[i] = 'D'

elif A[a - 2][b] < A[a - 1][b - 1] and A[a - 2][b] < A[a][b - 2]:

a -= 1

B[i] = 'D'

else:

b -= 1

B[i] = 'R'

return B

Fin = open('input.txt', "r+")

s = Fin.readline().split()

if not s:

print('Файл пустой')

elif int(s[0]) == 0:

print('0 элементов Матрицы')

else:

N = int(s[0])

A = []

A = Matr(A, N, N)

A = CHER(A, N)

B = [0]\*2\*(N-1)

B = PUT(A, N, B)

for i in range(N):

Fin.write("\n {}".format(A[i]))

Fin.write("\n {}".format(A[N-1][N-1]))

Fin.write("\n {}".format(B))

Fin.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 | [Файл](https://drive.google.com/file/d/1g_8tluUmiqiLCzBQwYyoUK6_qivtzO6I/view?usp=sharing) | 34.0  ['R', 'R', 'R', 'D', 'D', 'D', 'D', 'R'] | 34.0  ['R', 'R', 'R', 'D', 'D', 'D', 'D', 'R'] |
| 2 | Файл |  |  |