**ФГБОУ ВО “МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”**

**Лабораторная работа № 5**

Разветвляющиеся вычислительные процессы

**Задание 1 Вариант № 10**

по дисциплине:

Основы программирования

Выполнил:

студент 1 курса

группы 201-322

Кириллов Д.П

**МОСКВА 2020**

**Постановка задачи:**

В двумерном массиве вычислить количество локальных минимумов и сумму модулей элементов которые находятся выше главной диагонали

**Теоретическая часть:**

Для решения задачи были использованы вложенные списки, встроенная функция abs(), циклы, условные операторы. Для организации массива – конечной последовательности однотипных величин был использован список, заполненный с помощью цикла for. Также была использована функция randint из модуля Math

**Описание программы:**

Программа написана на алгоритмическом языке Python 3.8.5, реализована в среде ОС Windows 10

**Описание алгоритма:**

1. Создание двумерного массива с помощью функции makeRandomArr()
2. Вывод исходного массива
3. Вывод количества локальных минимумов с помощью функции findLocalMin()
4. Вывод суммы модулей элементов которые находятся выше главной диагонали с помощью функции sumAbs()

**Листинг программы:**

from random import (randint)

# Возвращает двумерный массив Row x Col со случайными значениями

# в диапазоне от min до max

def makeRandomArr(row, col, min, max):

arr = [[0] \* col for i in range(row)]

for Row in range(row):

for Col in range(col):

arr[Row][Col] = randint(min, max)

return arr

# Выводит количество локальных минимумов

def findLocalMin(arr):

rowLength = len(arr)

colLength = len(arr[0])

count = 0

for Row in range(rowLength):

for Col in range(colLength):

# Текущий элемент который нужно проверить

el = arr[Row][Col]

# Проверка первой строки

if (Row == 0):

if (Col == 0):

# Проверка элемента в левом верехнем углу

if (el < arr[Row][Col + 1])\

and (el < arr[Row + 1][Col + 1])\

and (el < arr[Row + 1][Col]):

count += 1

elif (Col == colLength - 1):

# Проверка элемента в правом верхнем углу

if (el < arr[Row][Col - 1])\

and (el < arr[Row + 1][Col - 1])\

and (el < arr[Row + 1][Col]):

count += 1

else:

# Проверка остальных элементов в первой строке

if (el < arr[Row][Col - 1])\

and (el < arr[Row + 1][Col - 1])\

and (el < arr[Row + 1][Col])\

and (el < arr[Row + 1][Col + 1])\

and (el < arr[Row][Col + 1]):

count += 1

# Проверка последней строки

if (Row == rowLength - 1):

if (Col == 0):

# Проверка элемента в левом нижнем углу

if (el < arr[Row - 1][Col])\

and (el < arr[Row - 1][Col + 1])\

and (el < arr[Row][Col + 1]):

count += 1

elif (Col == colLength - 1):

# Проверка элемента в правом верхнем углу

if (el < arr[Row - 1][Col])\

and (el < arr[Row - 1][Col - 1])\

and (el < arr[Row][Col - 1]):

count += 1

else:

# Проверка остальных элементов в поселедней строке

if (el < arr[Row][Col - 1])\

and (el < arr[Row - 1][Col - 1])\

and (el < arr[Row - 1][Col])\

and (el < arr[Row - 1][Col + 1])\

and (el < arr[Row][Col + 1]):

count += 1

# Проверка остальных строк (промежуточных)

if (Row != rowLength - 1) and (Row != 0):

# Проверка левого крайнего элемента

if (Col == 0):

if (el < arr[Row - 1][Col])\

and (el < arr[Row + 1][Col])\

and (el < arr[Row][Col + 1])\

and (el < arr[Row + 1][Col + 1])\

and (el < arr[Row - 1][Col + 1]):

count += 1

# Проверка правого крайнего элемента

elif (Col == colLength - 1):

if (el < arr[Row - 1][Col])\

and (el < arr[Row + 1][Col])\

and (el < arr[Row][Col - 1])\

and (el < arr[Row + 1][Col - 1])\

and (el < arr[Row - 1][Col - 1]):

count += 1

# Проверка любых не крайних элементов

else:

if (el < arr[Row - 1][Col - 1])\

and (el < arr[Row - 1][Col])\

and (el < arr[Row - 1][Col + 1])\

and (el < arr[Row + 1][Col - 1])\

and (el < arr[Row + 1][Col])\

and (el < arr[Row + 1][Col + 1])\

and (el < arr[Row][Col - 1])\

and (el < arr[Row][Col + 1]):

count += 1

return count

# Находит сумму элементов выше главной диагонали

def sumAbs(arr):

rowLength = len(arr)

colLength = len(arr[0])

sum = 0

startIndex = 1

for Row in range(rowLength - 1):

for Col in range(startIndex, colLength):

sum += abs(arr[Row][Col])

startIndex += 1

return sum

arr = makeRandomArr(10, 10, 0, 9)

print('\_\_\_\_\_\_\_ Исходный массив \_\_\_\_\_\_\_')

for Row in range(len(arr)):

print(arr[Row])

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

print('Количество локальных минимумов: {0}'.format(findLocalMin(arr)))

print('Сумма модулей элементов выше главной диагонали: {0}'.format(sumAbs(arr)))

**Результат тестирования программы**

\_\_\_\_\_\_\_ Исходный массив \_\_\_\_\_\_\_

[2, 5, 0, 3, 4, 0, 5, 1, 2, 1]

[7, 5, 4, 4, 1, 9, 5, 5, 1, 6]

[9, 0, 1, 4, 3, 1, 3, 8, 2, 7]

[1, 6, 5, 1, 4, 8, 1, 7, 2, 9]

[5, 8, 9, 1, 4, 3, 2, 5, 1, 5]

[5, 2, 3, 6, 5, 1, 6, 2, 6, 7]

[3, 8, 8, 6, 5, 2, 2, 9, 5, 2]

[1, 4, 1, 0, 5, 0, 8, 5, 3, 9]

[1, 1, 4, 9, 9, 0, 8, 3, 9, 9]

[8, 1, 4, 4, 8, 4, 9, 6, 7, 4]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Количество локальных минимумов: 10

Сумма модулей элементов выше главной диагонали: 189

**Использованная литература**

pythonworld.ru Справочник для начинающих. Python 3