Студент группы ИС-26 Осадчий Даниил

Практическое занятие №6

Тема: составление программ с функциями в IDE PyCharm Community.

Цели практического занятия: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ со списками в IDE PyCharm Community.

Постановка задачи:

1. Дан список размера N и целые числа K и L ($1 < K \leq L \leq N$). Найти среднее арифметическое всех элементов списка, кроме элементов с номерами от K до L включительно.

Тип алгоритма: циклический

Текст программы:

```
#Дан список размера N и целые числа K и L (1 < K ≤ L ≤ N). Найти среднее
#арифметическое всех элементов списка, кроме элементов с номерами от K до L
#включительно.
import random
#вводим размер списка
N = input("Введите размер списка N: ")

while type(N) != int: #проверка для исключения ошибок
    try:
        N = int(N)
    except:
        print('Число должно быть целым!')
        N = input("Введите размер списка N: ")

# Генерация случайных значений K и L
K = random.randint(2, N - 1)#K должно быть от 2 до N-1, чтобы 1 < K ≤ L ≤ N
L = random.randint(K, N)#L должно быть от K до N

#создаем список из N элементов
lst = list(range(1, N + 1))  # Например, список от 1 до N
print(f"Созданный список: {lst}")#выводим созданный список и числа для наглядности
print(f"Случайное значение K: {K}")
print(f"Случайное значение L: {L}")

def average_excluding_range(lst, K, L): #создаем функцию
    #исключаем элементы с индексами от K-1 до L
    excluded_elements = lst[:K - 1] + lst[L:]

    # Вычисляем сумму оставшихся элементов
    total_sum = sum(excluded_elements)

    # Вычисляем среднее арифметическое
    average = total_sum / len(excluded_elements)

    return average

# Вычисляем среднее арифметическое
result = average_excluding_range(lst, K, L)
print(f"Среднее арифметическое: {result}")
```

Протокол программ:

Введите размер списка N: 10
Созданный список: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Случайное значение K: 6
Случайное значение L: 6
Среднее арифметическое: 5.444444444444445

Постановка задачи:

2. Даны списки A и B одинакового размера N. Поменять местами их содержимое и вывести вначале элементы преобразованного списка A, а затем — элементы преобразованного списка B.

Тип алгоритма: Циклический

Текст программы:

```
# Даны списки A и B одинакового размера N. Поменять местами их содержимое и
# вывести вначале элементы преобразованного списка A, а затем — элементы
# преобразованного списка B.
import random

#вводим размер списка
N = input("Введите размер списков N: ")

while type(N) != int: #проверка для исключения ошибок
    try:
        N = int(N)
    except:
        print('Число должно быть целым!')
        N = input("Введите размер списка N: ")

#генерируем рандомные числа для списка A и B
A = [random.randint(1, 100) for _ in range(N)]#рандомные числа от 1 до 100
B = [random.randint(1, 100) for _ in range(N)]#рандомные числа от 1 до 100

#показываем изначальные списки
print("Исходный список A:", A)
print("Исходный список B:", B)

#меняем местами содержимое списков
A, B = B, A

#выводим получившийся список A
print("Список A после замены:", A)

#выводим получившийся список B
print("Список B после замены:", B)
```

Протокол программ:

Введите размер списков N: 5
Исходный список A: [38, 38, 69, 77, 69]
Исходный список B: [2, 78, 82, 38, 96]
Список A после замены: [2, 78, 82, 38, 96]
Список B после замены: [38, 38, 69, 77, 69]

Process finished with exit code 0

Постановка задачи:

3. множество A из N точек (точки заданы своими координатами x, y). Среди всех точек этого множества, лежащих во второй четверти, найти точку, наиболее удаленную от начала координат. Если таких точек нет, то вывести точку с нулевыми координатами.

Тип алгоритма: Циклический

Текст программы:

```
# Дано множество A из N точек (точки заданы своими координатами x, y). Среди всех
# точек этого множества, лежащих во второй четверти, найти точку, наиболее
# удаленную от начала координат. Если таких точек нет, то вывести точку с нулевыми
# координатами.
# краткий экскурс в понятия математики используемые для этой задачи
#координаты делятся на 4 четверти
# первая четверть: x>0 y>0
# вторая четверть: x<0 y>0
# третья четверть: x<0 y<0
# четвертая четверть: x>0 y<0
# Расстояние от точки (x,y) до начала координат(0,0) = sqrt(x**2 + y**2)

import math
#ввод данных
N = input("Введите количество точек N: ")

while type(N) != int: #проверка для исключения ошибок
    try:
        N = int(N)
    except:
        print('Число должно быть целым!')
        N = input("Введите размер списка N: ")

points = [] #создаем список для хранения введенных точек
for _ in range(N): #ввод точек для указанного количества точек
    x = float(input(f"Введите координату x для точки {_ + 1}: "))
    y = float(input(f"Введите координату y для точки {_ + 1}: "))
    points.append((x, y)) #добавляем наши значения в список

#функция для вычисления расстояния от точки до начала координат
def distance_to_origin(x, y):
    return math.sqrt(x**2 + y**2)

#функция для поиска точки, наиболее удаленной от начала координат во второй
четверти
def find_farthest_point(points):
    farthest_point = (0, 0) #это точка по умолчанию, которая будет возвращена,
если не найдено ни одной точки во второй четверти
    max_distance = -1  #переменная для хранения максимального расстояние
```

```
    for x, y in points: #перебираем все точки из списка points
        #проверяем, лежит ли точка во второй четверти
        if x < 0 and y > 0:
            #если точка лежит во второй четверти, вычисляем её расстояние до
начала координат
            distance = distance_to_origin(x, y)
            #если текущее расстояние больше максимального, обновляем данные
            if distance > max_distance:
                max_distance = distance
                farthest_point = (x, y)

    return farthest_point


# Находим точку, наиболее удаленную от начала координат во второй четверти
result = find farthest point(points)

# Выводим результат
print("Точка, наиболее удаленная от начала координат во второй четверти:", result)
```

Протокол программ:

```
Введите количество точек N: 2
Введите координату x для точки 1: -1
Введите координату y для точки 1: 2
Введите координату x для точки 2: -4
Введите координату y для точки 2: 5
Точка, наиболее удаленная от начала координат во второй четверти: (-4.0, 5.0)

Process finished with exit code 0
```

Вывод:

В процессе работы я закрепил полученные ранее навыки, приобрел новые навыки в использование списков и работы с ними, научился создавать программы с использованием библиотеки random и библиотеки math в IDE PyCharm Community.