Студент группы ИС-23 Саруев П.А

Практическое занятие № 6

Тема: составление программ со списками в IDE PyCharm Community.

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

Постановка задачи. 1
Задание 1. Дан список A размера N. Найти минимальный элемент из его элементов с четными номерами: A2, A4, A6, ...

Тип алгоритма: интеративный алгоритм

Текст программы:

```
def find_min_even_indexed(A):
    if len(A) < 2:
        return None  # Если нет четных индексов

    min_value = float('inf')  # Инициализируем минимальное значение
    for i in range(1, len(A), 2):  # Проходим по четным индексам (A[2], A[4], ...)
        if A[i] < min_value:
            min_value = A[i]

    return min_value if min_value != float('inf') else None


# Пример использования
A = [5, 3, 8, 1, 4, 7]
min_even_indexed = find_min_even_indexed(A)
print("Минимальный элемент с четными индексами:", min_even_indexed)
```

Протокол работы программы:

```
Минимальный элемент с четными индексами: 1

Process finished with exit code 0
```

Постановка задачи. 2
Задание 2. Дан целочисленный список A размера N. Переписать в новый целочисленный список B все четные числа из исходного списка (в том же порядке) и вывести размер полученного список B и его содержимое.

Тип алгоритма: интеративный алгоритм

Текст программы:

```
def extract_even_numbers(A):
    B = []  # Инициализируем пустой список для четных чисел
    for number in A:  # Проходим по каждому элементу в списке A
        if number % 2 == 0:  # Проверяем, является ли число четным
```

```
            B.append(number)  # Добавляем четное число в список B

    return B  # Возвращаем новый список с четными числами


# Пример использования
A = [5, 3, 8, 1, 4, 7, 10, 12]
B = extract_even_numbers(A)

# Выводим размер и содержимое списка B
print("Размер списка B:", len(B))
print("Содержимое списка B:", B)
```

Протокол работы программы:

```
Размер списка B: 4
Содержимое списка B: [8, 4, 10, 12]

Process finished with exit code 0
```

Постановка задачи. 3

Задание 3. Дано множество A из N точек (N > 2, точки заданы своими координатами x, y). Найти наибольший периметр треугольника, вершины которого принадлежат различным точкам множества A, и сами эти точки (точки выводятся в том же порядке, в котором они перечислены при задании множества A). Расстояние R между точками с координатами (x1, y1) и (x2, y2) вычисляется по формуле: $R = \sqrt{(x2 – x1)2 + (y2 – y1)2}$. Для хранения данных о каждом наборе точек следует использовать по два список: первый список для хранения абсцисс, второй — для хранения ординат.

Тип алгоритма: интеративный алгоритм

Текст программы:

```
import math


def max_triangle_perimeter(x_coords, y_coords):
    n = len(x_coords)
    max_perimeter = 0
    best_triangle = None

    # Перебор всех троек точек
    for i in range(n):
        for j in range(i + 1, n):
            for k in range(j + 1, n):
                # Вычисление длин сторон треугольника
                a = math.sqrt((x_coords[i] - x_coords[j]) ** 2 + (y_coords[i] - y_coords[j]) ** 2)
                b = math.sqrt((x_coords[i] - x_coords[k]) ** 2 + (y_coords[i] - y_coords[k]) ** 2)
                c = math.sqrt((x_coords[j] - x_coords[k]) ** 2 + (y_coords[j] - y_coords[k]) ** 2)

                # Проверка существования треугольника
```

```
            if a + b > c and a + c > b and b + c > a:
                perimeter = a + b + c
                if perimeter > max_perimeter:
                    max_perimeter = perimeter
                    best_triangle = (i, j, k)

    if best_triangle is not None:
        print("Максимальный периметр:", max_perimeter)
        print("Координаты вершин треугольника:")
        print(f"Точка 1: ({x_coords[best_triangle[0]]}, {y_coords[best_triangle[0]]})")
        print(f"Точка 2: ({x_coords[best_triangle[1]]}, {y_coords[best_triangle[1]]})")
        print(f"Точка 3: ({x_coords[best_triangle[2]]}, {y_coords[best_triangle[2]]})")
    else:
        print("Не удалось найти треугольник.")


x_coordinates = [0, 3, 6, 2]
y_coordinates = [0, 4, 1, 5]
max_triangle_perimeter(x_coordinates, y_coordinates)
```

Протокол работы программы:

```
Максимальный периметр: 17.124781586925103
Координаты вершин треугольника:
Точка 1: (0, 0)
Точка 2: (6, 1)
Точка 3: (2, 5)

Process finished with exit code 0
```

Вывод: в процессе выполнения практического занятия выработал(а) навыки составления программ со списками в IDE PyCharm Community.Выполнены разработка кода, отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHub