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) Дан список размера N и целые числа K и L (1 < K < L < N). Найти сумму всех элементов списка, кроме элементов с номером от K до L включительно.

2) Дан целочисленный список размера N. Если он является перестановкой, то есть

содержит все числа от 1 до N, то вывести 0; в противном случае вывести номер

первого недопустимого элемента.

3) Дано множество A из N точек на плоскости и точка B (точки заданы своими

координатами x, y). Найти точку из множества A, наиболее близкую к точке B.

Расстояние R между точками с координатами (XI, y) и (x2, У2) вычисляется по

формуле:

R = 4x2 — x 1)2+ (y2 — y 1)2.

Для хранения данных о каждом наборе точек следует использовать по два списка: первый

список для хранения абсцисс, второй — для хранения ординат.

Тип алгоритма: Линейный

Текст программы:

```
1) # Дан список размера N и целые числа K и L (1 < K < L < N). Найти сумму
всех элементов списка, кроме элементов с номером
# от K до L включительно.
def find_sum_except_range(numbers, k, l):
    if k >= l or l >= len(numbers):
        return "Некорректные значения K и L"

    sum_except_range = 0
    for i in range(len(numbers)):
        if i < k or i > l:
            sum_except_range += numbers[i]

    return sum_except_range

# Пример использования
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
k = int(input("Значение K: "))
l = int(input("Значение L: "))
result = find_sum_except_range(numbers, k, l)
print(result)
2) def check_permutation(lst):
    n = len(lst)
    visited = [False] * n
```

```
    for num in lst:
        if num < 1 or num > n or visited[num - 1]:
            return num
        visited[num - 1] = True

    return 0

# Пример использования функции
input_list = [4, 1, 3, 2, 5]
result = check_permutation(input_list)
print(input_list)
print(result)
```

```
3) # Дано множество A из N точек на плоскости и точка B (точки заданы
своими
# координатами x, y). Найти точку из множества A, наиболее близкую к точке
B.
# Расстожше R между точками с координатами (XI, y) и (м, y) вычисляется по
формуле: R = 4x2 - x1)2 + (y2 - y1)2.
#Для хранения данных о каждом наборе точек следует использовать по два
списка: первый список для хранения абсцисс,
# второй — для хранения ординат.
import math


def find_closest_point(A, B):
    closest_point = None
    min_distance = float('inf')

    for i in range(len(A[0])):
        x = A[0][i]
        y = A[1][i]

        distance = math.sqrt((x - B[0]) ** 2 + (y - B[1]) ** 2)

        if distance < min_distance:
            min_distance = distance
            closest_point = (x, y)

    return closest_point


# Пример использования функции
A = [[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]]
B = (2, 8)

closest_point = find_closest_point(A, B)
print("Ближайшая точка:", closest_point)
```

Протокол программы

1)

Значение K: 2

Значение L: 6

30

Process finished with exit code 0

2)

[4, 1, 3, 2, 5]

0

3)

Ближайшая точка: (2, 7)

Process finished with exit code 0

Вывод: Я усвоенные знания понятия алгоритмы основные принципы составления программ