

## Практическое занятие № 6\_1

**Тема:** составление программ со списками в IDE PyCharm Community.

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

**Постановка задачи.**

Дан целочисленный список размера N, не содержащий одинаковых чисел. Проверить, образуют ли его элементы арифметическую прогрессию. Если образуют, то вывести разность прогрессии, если нет — вывести 0.

**Тип алгоритма:** Циклический

**Текст программы:**

```
def my_def(lst):
    n = len(lst)
    if n < 2:
        #Список меньше двух элементов, не может быть прогрессии
        return 0

    #Находим разность между первыми двумя элементами
    diff = lst[1] - lst[0]

    #Проверяем, что все остальные пары имеют такую же разность
    for i in range(2, n):
        if lst[i] - lst[i - 1] != diff:
            # Прогрессия нарушена
            return 0

    #Все условия выполнены, возвращаем разность прогрессии
    return diff

#Вывод: 2
lst = [2, 4, 6, 8, 10]
print(my_def(lst))

lst = [1, 3, 5, 9]
#Вывод: 0
print(my_def(lst))
```

## Протокол работы программы:

2

0

Process finished with exit code 0

## Практическое занятие № 6\_2

**Тема:** составление программ со списками в IDE PyCharm Community.

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

### Постановка задачи.

Дан список A размера N. Сформировать новый список B того же размера, элементы которого определяются следующим образом:  $BK = 2 * AK$ , если  $AK < 5$ ,  $AK/2$  в противном случае.

**Тип алгоритма:** Циклический.

### Текст программы.

```
#Функция создает новый список B на основе списка A.
def list(A):
    #Создаем пустой список B, в который будем добавлять преобразованные
    элементы
    B = []
    #Проходим по каждому элементу списка A
    for element in A:
        try:
            #Если элемент из A меньше 5, то в B записывается удвоенный
            элемент.
            if element < 5:
                B.append(2 * element)
            #Либо записывается половина элемента.
            else:
                B.append(element / 2)
        except ValueError:
            print("Ошибка: введено некорректное значение")
    #Возращение функции в B
    return B

#Даем список A
A = [1, 3, 5, 7, 9]
#Сформатировали новый список в B
B = list(A)
#Вывод B
print(B)
```

A

**Протокол работы программы:**

**[2, 6, 2.5, 3.5, 4.5]**

**Process finished with exit code 0**

## Практическое занятие № 6\_3

**Тема:** составление программ со списками в IDE PyCharm Community.

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

### Постановка задачи.

Дано множество  $A$  из  $N$  точек (точки заданы своими координатами  $x, y$ ). Среди всех точек этого множества, лежащих в первой или третьей четверти, найти точку, наиболее близкую к началу координат. Если таких точек нет, то вывести точку с нулевыми координатами. Расстояние  $R$  между точками с координатами  $(x_1, y_1)$  и  $(x_2, y_2)$  вычисляется по формуле:  $R = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ . Для хранения данных о каждом наборе точек следует использовать по два списка: первый список для хранения абсцисс, второй — для хранения ординат.

**Тип алгоритма:** Циклический

### Текст программы.

```
#Импортируем модуль math для использования функции sqrt
import math

#Функция находит ближайшую точку к началу координат среди точек,
расположенных в первой или третьей четверти.
def my_def(x_coordinates, y_coordinates):
    #Проверяем, что списки одинаковой длины
    if len(x_coordinates) != len(y_coordinates):
        #Если списки разной длины, выбрасываем ошибку
        raise ValueError("Списки должны быть одинаковой длины")

    # Переменная для хранения ближайших координат
    A = None
    #Начальное значение минимального расстояния
    num2 = float('inf')

    #Проходим по каждому элементу в списках
    for i in range(len(x_coordinates)):
        #Текущая x
        x = x_coordinates[i]
        #Текущая y
        y = y_coordinates[i]
```

```
# Если произведение координат больше нуля, точка в первой или третьей
четверти
    if x * y > 0:
        # Вычисляем расстояние до начала координат
        num1 = math.sqrt(x ** 2 + y ** 2)
        # Если расстояние меньше текущего минимума
        if num1 < num2:
            #Обновляем минимальное расстояние
            num2 = num1 # Обновляем минимальное расстояние
            #Сохраняем координаты ближайшей точки
            A = (x, y)

#Возвращаем ближайшую точку или (0, 0), если подходящей точки нет
return A or (0, 0)

#Обработка исключения
try:
    #Считываем количество точек
    N = int(input())
    #Считываем абсциссы точек
    x_coordinates = list(map(int, input().split()))
    #Считываем ординаты точек
    y_coordinates = list(map(int, input().split()))

    #Вызываем функцию для поиска ближайшей точки
    B = my_def(x_coordinates, y_coordinates)
    #Выводим результат
    print(B[0], B[1])

#Обработка исключения
except ValueError:
    print("Ошибка: введено некорректное значение.")
```

## Протокол работы программы:

12

32

512

32 512

Process finished with exit code 0

**Вывод:** в процессе выполнения практического занятия выработал навыки составления программ циклической структуры в IDE PyCharm Community. Были использованы языковые конструкции Try, Except. Выполнены разработка кода, отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHub.