**Практическое занятие № 6**

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

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

**Постановка задач.**

1. Дан список размером 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.
2. Даны списки А и В одинакового размера N. Поменять местами их содержимое и вывести вначале элементы преобразованного списка А, а затем – элементы преобразованного списка В.
3. 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.

Расстояние R между точками с координатами (x1, y1) и (x2, у2) вычисляется по формуле: R = √(x2 – x1)2 + (у2 – y1)2 . Для хранения данных о каждом наборе точек следует использовать по два списка: первый список для хранения абсцисс, второй — для хранения ординат.

**Текста программ.**



# 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 К до L.  
**import** random  
  
**try**: # Обработчик исключений  
 N = int(input(**'Введите размер списка: '**)) # Ввод размера списка  
 print(**'Введите два целых числа K и L, которые (1 < K <= L <= N)'**)  
 K = int(input(**'Значение К: '**))   
 L = int(input(**'Значение L: '**))   
 **if** (K > 1) **and** (L >= K) **and** (N >= L):  
 a = [] # Вводим пустой список для записи чисел  
 b = N # Счетчик для цикла  
 **while** b != 0:  
 a.append(random.randint(-10, 10)) # Заполняем список значениями в кол-ве N  
 b -= 1  
 print(**f'Список: {**a**}'**)  
 x = a[:K] # Делаем срез списка до элемента с номером К  
 z = a[L + 1:] # Срез списка после элемента с номером L  
 x.extend(z) # Увеличиваем список x значениями из списка z  
 print(sum(x) / len(x)) # Находим среднее арифметическое  
 **else**:  
 print(**'Введенные числа не подходят'**)  
**except** ValueError:  
 print(**'Неверно введенные данные'**)



*# Даны списки А и В одинакового размера N.  
# Поменять местами их содержимое и вывести вначале элементы пребразованного списка А  
# а затем элементы преобразованного списка В***import** random  
  
**try**: *# Обработчик исключений* N = int(input(**'Введите размер списков А и В (целое число): '**)) *# Ввод размера списков* A = [] *# Ввод пустого списка А* B = [] *# Ввод пустого списка В* c = N *# Счетчик для цикла* **while** c != 0:  
 A.append(random.randint(-10, 10)) *# Ввод значений для списка А* B.append(random.randint(-10, 10)) *# Ввод значений для списка B* c -= 1  
 A1 = A *# Сохранение изначального списка А* B1 = B *# Сохранение изначального списка В* A = B1 *# Меняем содержимое списка А на элементы списка В* B = A1 *# Меняем содержимое списка В на элементы списка А* print(**f'Изначальный список А: {**A1**}\nИзначальный список B: {**B1**}\nРезультат: '**)  
 print(\*A, end=**' '**)  
 print(\*B)  
**except** TypeError:  
 print(**'Введите целое число'**)

*# Дано множество А из точек N (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.***from** math **import** \*  
**import** random  
  
  
a = [] *# Пустой список для значений расстояния*maxi\_X = 0 *# Стартовая координата самой удаленной точки (х)*maxi\_Y = 0 *# Стартовая координата самой удаленной точки (y)***def** cor(x1, y1, x2, y2):  
 R = sqrt(((x2 - x1) \*\* 2) + ((y2 - y1) \*\* 2))  
 a.append(R)  
  
  
x2 = 0 *# Абцисса начала координат*y2 = 0 *# Ордината начала координат*maxi = 0 *# Стартовое максимальное значение расстояния***try**:  
 N = int(input(**'Введите количество точек: '**))  
 **if** N >= 0:  
 X = [] *# Начальный список абцисс* Y = [] *# Начальный список ординат* X1 = [] *# Список подходящих абцисс точек* Y1 = [] *# Список подходящих ординат точек* b = N *# Счетчик для цикла* **while** b != 0:  
 X.append(random.randint(-10, 10))  
 Y.append(random.randint(-10, 10))  
 b -= 1  
 **for** i **in** range(0, len(X)):  
 **if** (X[i] <= 0) **and** (Y[i] >= 0): *# Проверка условия, что точка находится во 2 четверти* X1.append(X[i])  
 Y1.append(Y[i])  
 print(**f'Абциссы точек: {**X**}\nОрдинаты: {**Y**}'**)  
 print(**f'Подходящие абциссы: {**X1**}\nОрдинаты: {**Y1**}'**)  
 **if** len(X1) != 0: *# Проверяем есть ли подходящие точки* **for** j **in** range(0, len(X1)):  
 cor(X1[j], Y1[j], x2, y2)  
 **for** l **in** range(0, len(a)):  
 **if** maxi < a[l]: *# Находим максимальное значение расстояния* maxi = a[l]  
 maxi\_X = X1[l] *# Записываем координаты самых удаленных точек* maxi\_Y = Y1[l]  
 **else**:  
 **continue** print(**f'Наиболее удалённые от начала координат: ({**maxi\_X**}, {**maxi\_Y**})'**)  
 **else**:  
 print(**f'Наиболее удалённые от начала координат: ({**maxi\_X**}, {**maxi\_Y**})'**)  
 **else**:  
 print(**'Введите целое число'**)  
**except** ValueError:  
 print(**'Введите целое число'**)

**Протокол работ программ.**



Введите размер списка: 7

Введите два целых числа K и L, которые (1 < K <= L <= N)

Значение К: 2

Значение L: 4

Список: [2, -1, -8, -3, 5, 0, 6]

1.75

Process finished with exit code 0



Введите размер списков А и В (целое число): 7

Изначальный список А: [-7, -7, -6, 5, 9, 0, 8]

Изначальный список B: [-6, -10, -3, 9, 4, -3, 5]

Результат:

-6 -10 -3 9 4 -3 5 -7 -7 -6 5 9 0 8

Process finished with exit code 0



Введите количество точек: 7

Абциссы точек: [-6, 4, -9, -6, 6, -2, -7]

Ординаты: [1, -1, 9, 10, 4, -1, 9]

Подходящие абциссы: [-6, -9, -6, -7]

Ординаты: [1, 9, 10, 9]

Наиболее удалённые от начала координат: (-9, 9)

Process finished with exit code 0

**Вывод:** закрепилаусвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрела навыки составления программ со списками в IDE PyCharm Community.

Выполнена разработка кода, отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHubю