Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Ульяновский государственный технический университет»

Кафедра «Вычислительная техника»

**Отчет по лабораторной работе №1**

Дисциплина: «Разработка профессиональных приложений»

Обработка списков

Вариант № 9

Выполнил:

студент группы ИВТАПбд-21

Карпова А.С.

Проверил:

преподаватель кафедры

«Вычислительная техника»

Исхаков И.И.

Ульяновск, 2023

**Задание по варианту**

Необходимо реализовать 9 вариант задания, а именно:

Между списками A и B обменять их самые длинные цепочки из одинаковых элементов. Пример: A[10]: 3 2 2 5 2 1 1 1 3 9; B[8]: 1 2 5 5 4 8 3 3

В списке A самая длинная цепочка: 1 1 1 (элементы с 7 по 9). В списке B самая длинная цепочка: 5 5 (элементы с 3 по 4).

Список A после перестановки в него цепочки из списка B: A[9]: 3 2 2 5 2 5 5 3 9. Список B после перестановки в него цепочки из списка A: B[9]: 1 2 1 1 1 4 8 3 3.

**Требования к программе**

1. Ввод элементов списка должен быть доступен с клавиатуры и путем автоматической генерации
2. Программа должна корректно обрабатывать некорректный ввод.
3. Исходный код должен быть откомментирован
4. Необходимо реализовать правильную декомпозицию программы на методы.
5. Необходимо реализовать предложенный алгоритм без использования стандартных функций и с использованием (необходимо реализовать два отдельных метода).

**Описание реализации**

Для решения данного задания была написана программа, которая находит максимальные последовательности одинаковых элементов в двух списках целых чисел A и B, затем заменяет эти последовательности между списками, и выводит результат. Пользователь может выбрать способ ввода элементов (с помощью автоматической генерации или с помощью клавиатуры).

Функция permutation является функцией перестановки. Она принимает на вход список arr и возвращает кортеж, содержащий индексы начала и конца максимальной последовательности одинаковых элементов в списке.

Затем программа получает элементы списков A и B в зависимости от выбора пользователя. Если пользователь выбрал автоматическую генерацию, то программа генерирует случайные целые числа в диапазоне от 1 до 7 для каждого элемента списка. Если пользователь выбрал ввод с клавиатуры, то программа запрашивает у пользователя ввод элементов списков и проверяет, являются ли эти элементы целыми числами.

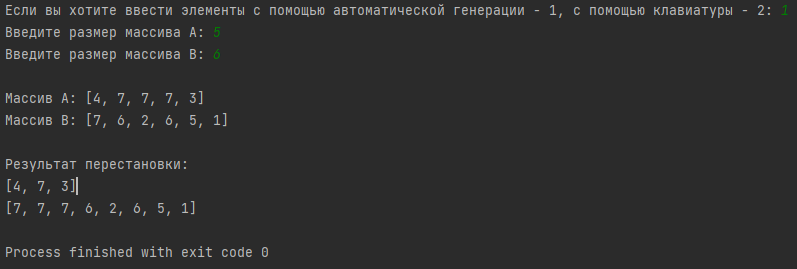
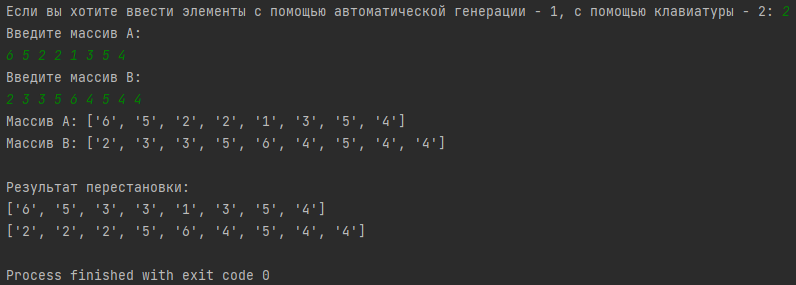
Затем программа находит максимальные последовательности одинаковых элементов в каждом списке с помощью функции permutation() и сохраняет их в a\_slice и b\_slice. Затем происходит замена максимальных последовательностей между массивами A и B.

Наконец, программа выводит в консоль измененные списки A и B.

Программа состоит из нескольких частей - функции перестановки, получения элементов списков, нахождения максимальных последовательностей одинаковых элементов, и замены максимальных последовательностей между списками. Каждая часть реализована как отдельный блок кода. Общая логика программы заключается в нахождении максимальных последовательностей одинаковых элементов в каждом списке с помощью функции permutation(), сохранении их в отдельные переменные, замене максимальных последовательностей между списками и выводе результатов.

**Описание возникших затруднений**

При написании кода трудностей не возникло.

**Результаты работы программы***Рис. 1.1 Заполнение случайными числами Рис 1.2. Заполнение с клавиатуры*

**Код программы**

import random

#функция перестановки

def permutation(arr):

# индекс элемента

index\_element = 1

# счетчик и максимальный счетчик; счетчик - это длина последовательности одинаковых элементов

counter = max\_counter = 1

for i in range(1, len(arr)):

#если текущий элемент списка равен предыдущему

if arr[i] == arr[i - 1]:

#то счетчик увеличивается на 1

counter += 1

else:

#если длина текущей последовательности одинаковых элементов больше максимальной длины

if counter > max\_counter:

#присваиваем максимальной длине последовательности одинаковых элементов новое значение

max\_counter = counter

index\_element = i

counter = 1

#если нашли самую максимальную длину

if counter > max\_counter:

max\_counter = counter

index\_element = i + 1

return index\_element - max\_counter, index\_element

command = int(input("Если вы хотите ввести элементы с помощью автоматической генерации - 1, с помощью клавиатуры - 2: "))

match command:

case 1:

N = int(input("Введите размер массива A: "))

M = int(input("Введите размер массива B: "))

A = []

B = []

#рандомное заполнение списка

for i in range(N):

A.append(random.randint(1, 7))

print("\nМассив A:", A)

for i in range(M):

B.append(random.randint(1, 7))

print("Массив B:", B)

case 2:

#ввод элементов с клавиатуры

print("Введите массив А: ")

A = input().split()

print("Введите массив В: ")

B = input().split()

#проверка списка на корректный ввод данных

res = all(ele.isdigit() for ele in A)

if res == False:

print("\nВведите, пожалуйста, целое число, а не иной символ.")

raise SystemExit(1)

res = all(ele.isdigit() for ele in B)

if res == False:

print("\nВведите, пожалуйста, целое число, а не иной символ.")

raise SystemExit(1)

print("Массив A:", A)

print("Массив B:", B)

#замена цепочек

a\_start, a\_end = permutation(A)

b\_start, b\_end = permutation(B)

#срез из списка А с максимальной последовательностью одинаковых элементов

a\_slice = A[a\_start:a\_end].copy()

#срез из списка В с максимальной последовательностью одинаковых элементов

b\_slice = B[b\_start:b\_end].copy()

#соединяем срез из начала списка, максимальную последовательность другого списка и конец текущего списка

A = A[:a\_start] + b\_slice + A[a\_end:]

B = B[:b\_start] + a\_slice + B[b\_end:]

print("\nРезультат перестановки: ")

print(A)

print(B)