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 №8

Выполнил:

студент группы ИВТАСбд-21

Зюзин Г.А.

Проверил:

преподаватель кафедры

«Вычислительная техника»

Исхаков И.И.

Ульяновск, 2023

**Техническое задание.**

Написать программу на языке программирования Python, которая выполняет действия над списком. Существует некий список чисел А. Необходимо из списка А удалить цепочки, которые состоят из нечётных чисел и ни один элемент из этого множества не содержиться в списке B. В программе ввод элементов множеств осуществляется пользователем или множества заполняются случайными генерируемыми числами.

**Описание реализации.**

Данная программа производит удалений цепочек (подпоследовательностей) в неком списке А, если ни один элемент подпоследовательности не содержиться в списке B.

Первое что нужно сделать - это выбрать каким способом будет происходить заполнение списков А и В (в ручную или генератором случайных чисел).

Если пользователь выбрал заполнение с клавиатуры, то ему необходимо через пробел в строку написать числа для обоих множеств. Программа получает на вход строку, поэтому необходимо разобрать её на целочисленный тип данных:

a = list(map(int, input().split()))

* input() получает пользовательский ввод и возвращает строку, например"1 2 3 4 5";
* input().split() разбивает этот ввод на пробелы, например["1", "2", "3", ...];
* int() преобразует строку в целое число, например"1" -> 1;
* map(fn, sequence) применяет функцию fn к каждому элементу в sequence, например fn(sequence[0]), fn(sequence[1]), ...;
* map(int, input().split()) применяется int к каждому элементу строки на входе, например["1", "2", "3", ...] -> int("1"), int("2"), …;
* list() превращает любой итератор/генератор в список, например int("1"), int("2"), ... => [1, 2, 3, ...];

Выбрав заполнение с помощью генератора случайных чисел, значение задаётся следующим образом:

b.append(random.randint(0, 9))

random - модуль, предоставляющий функции для генерации случайных чисел, букв, случайного выбора элементов последовательности.

random.randint(a, z) - метод, возвращающий случайное число в диапазоне от a до z.

В программе присутствует проверка корректности ввода, а именно: на пустоту множеств (каждый список должен иметь хотя бы один элемент), проверка выбора способа ввода.

Чтобы удалить необходимые цепочки их нужно выявить. Таких последовательностей может быть и больше одной, поэтому можно использовать для хранения список из списков, хранящие в себе индексы элементов каждой цепочки.

Пример. Дан некий список: 8 7 9 4 2 3 5 7 6 1. Сформировать подпоследовательность по тех. заданию, т.е. выявить цепочки из нечётных элементов. В данном списке следующие цепочки: [[7, 9], [3, 5, 7], [1]]. Если заменить эти элементы на их индексы, то: [[1, 2], [5, 6, 7], [9]].

Программная реализация имеет вид:

for i in range(len(a)):

if a[i] % 2 != 0:

# Добавление элемента в одну из цепочек

seq.append(i)

if i == len(a) - 1:

sequences.append(list(seq))

seq.clear()

elif len(seq) != 0:

sequences.append(list(seq))

seq.clear()

После этого начинаем удалять эти подпоследовательности, если ни один элемент не содержиться в множестве B. Для данного алгоритма можно использовать флаговую переменную. Если находится нечётный элемент в этой цепочке, который содержится в B, то остаётся в списке A:

if existence(a[i - count\_deleted\_elements], b):

flag = True

break

После удаление каждого элемента меняются индексы всех других, идущих после него, поэтому для дальнейшего удаления вычисляются новые позиции. count\_deleted\_elements - переменная, которая считает количество удалённых элементов на протяжении всей работы програмы. Перед каждым удалением элемента вычисляется его индекс:

i - count\_deleted\_elements

a.pop(j - count\_deleted\_elements)

Для того чтобы убрать дублирование кода, была проведена его декомпозиция. Функция для генерации случайного списка:

def generate\_random\_list(left, right, size):

list = []

for i in range(size):

list.append(random.randint(left, right))

return list

Исходный код программы имеет написан как с использованием с некоторых стандартных функций, так и без них.

Были реализованы стандартная функция len() и оператор in. len() - функция, которая возвращает длину списка. in - оператор, который проверяет есть ли значение или переменная в последовательности (строке, списке, кортеже, множестве или словаре):

def get\_len\_list(list):

count = 0

for i in list:

count += 1

return count

def existence(element, list):

for i in list:

if i == element:

return True

return False

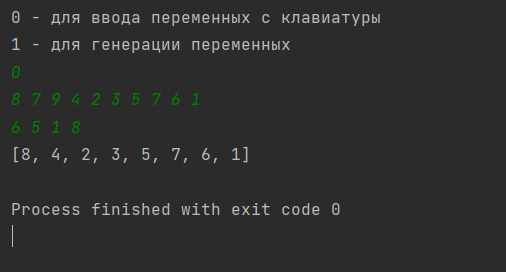
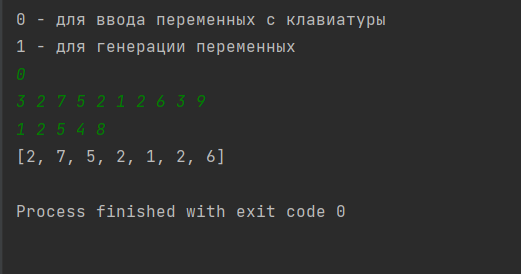
**Описание возникших затруднений.**

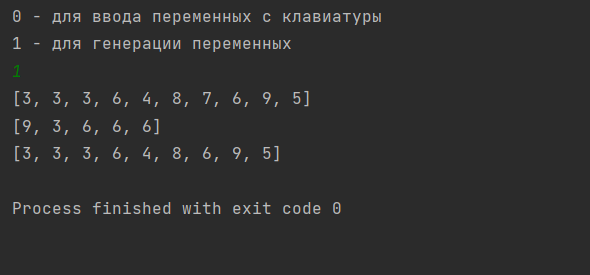
При написании исходного кода возникла трудность определения подпоследовательности в списке. Трудность заключается в следующем: подпоследовательность некого списка, может являться частью другой подпоследовательностью («под-подпоследовательностью»).

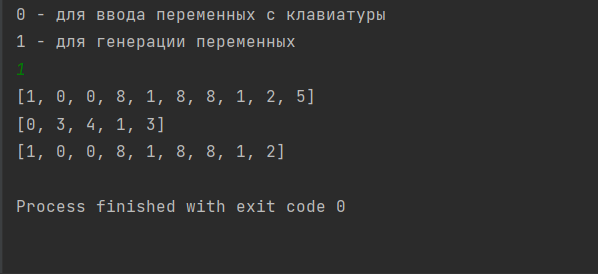
Данная ситуация может привести к удалению подпоследовательности не полностью, т.е. при некой цепочки из 7 элементов (к примеру), которая является частью другой цепочки из 9 элементов (к примеру), может удалиться другая цепочка и оставить 2 элемента, которые должны были быть удалены.

Данная проблема была решена с помощью индексов.

**Тестирование программы.**

Заполнение списков с помощью ввода с клавиатуры:

Заполнение списков путём автоматической генерации чисел:



**Листинг кода.**

Исходный код программы без использования стандартных функций:

# Лабораторная работа №1. Работа со списком

# Вариант 8

import random

def generate\_random\_list(left, right, size):

list = []

for i in range(size):

list.append(random.randint(left, right)) # Метод, который возвращает случайное число в заданном диапозоне

return list

def get\_len\_list(list):

count = 0

for i in list:

count += 1

return count

def existence(element, list): # Функция проверки наличия элемента (аналог in)

for i in list:

if i == element:

return True

return False

print("0 - для ввода переменных с клавиатуры"

"\n1 - для генерации переменных")

w = int(input())

if w != 0 and w != 1:

print("Нет такого ответа...")

a, b = [], []

if not(bool(w)):

# Наполнение множества А. map

a = list(map(int, input().split()))

# Наполнение множества B. map

b = list(map(int, input().split()))

else:

# Наполнение множеств случайными числами

a, b = generate\_random\_list(0, 9, 10), generate\_random\_list(0, 9, 5)

print(a)

print(b)

if get\_len\_list(a) == 0 or get\_len\_list(b) == 0:

print("Один из списков пустой...")

quit()

# Список, содержащий ключи элементов подпоследовательности в множестве А

seq = []

# Список из seq

sequences = []

# Добавление цепочек

for i in range(get\_len\_list(a)):

if a[i] % 2 != 0:

# Добавление элемента в одну из цепочек

seq.append(i)

if i == get\_len\_list(a) - 1:

sequences.append(list(seq)) # Метод list() позволяет запушить список в список.

seq.clear() # Очищение списка

elif get\_len\_list(seq) != 0:

sequences.append(list(seq))

seq.clear()

# Удаление цепочек

count\_deleted\_elements = 0 # Кол-во удалённых элементов, требуется для вычисления индекса

for s in sequences: # foreach для прохождения списком по списку из список

flag = False

for i in s: # foreach для прождению списку ключей одной из цепочек

if existence(a[i - count\_deleted\_elements], b): # При удалении элемента индекс последующих меняется, но он не меняется в списках

flag = True # Флаговая меняет своё значение, если находит элемент содержится в B,

break

if not(flag):

for j in s:

a.pop(j - count\_deleted\_elements)

count\_deleted\_elements += 1

print(a)

Исходный код программы с использованием стандартных функций:

def generate\_random\_list(left, right, size):

list = []

for i in range(size):

list.append(random.randint(left, right)) # Метод, который возвращает случайное число в заданном диапазоне

return list

print("0 - для ввода переменных с клавиатуры"

"\n1 - для генерации переменных")

w = int(input())

if w != 0 and w != 1:

print("Нет такого ответа...")

a, b = [], []

if not(bool(w)):

# Наполнение множества А. map

a = list(map(int, input().split()))

# Наполнение множества B. map

b = list(map(int, input().split()))

else:

# Наполнение множеств случайными числами

a, b = generate\_random\_list(0, 9, 10), generate\_random\_list(0, 9, 5)

print(a)

print(b)

if len(a) == 0 or len(b) == 0:

print("Один из списков пустой...")

quit()

# Список, содержащий ключи элементов подпоследовательности в множестве А

seq = []

# Список из seq

sequences = []

# Добавление цепочек

for i in range(len(a)):

if a[i] % 2 != 0:

# Добавление элемента в одну из цепочек

seq.append(i)

if i == len(a) - 1:

sequences.append(list(seq)) # Метод list() позволяет запушить список в список.

seq.clear() # Очищение списка

elif len(seq) != 0:

sequences.append(list(seq))

seq.clear()

# Удаление цепочек

count\_deleted\_elements = 0 # Кол-во удалённых элементов, требуется для вычисления индекса

for s in sequences: # foreach для прохождения списком по списку из список

flag = False

for i in s: # foreach для прождению списку ключей одной из цепочек

if a[i - count\_deleted\_elements] in b: # При удалении элемента индекс последующих меняется, но он не меняется в списках

flag = True # Флаговая меняет своё значение, если находит элемент содержится в B,

break

if not(flag):

for j in s:

a.pop(j - count\_deleted\_elements)

count\_deleted\_elements += 1

print(a)

**Вывод.**

Программный код протестирован и работает успешно. Удаляет цепочки, состоящих только из нечётных элементов, из списка А, если в списке B нет ни одного элемента этой цепи. Исходный код написан на языке программирования Python, в котором были так же реализованы стандартныые функции.