Министерство науки и образования РФ

Федеральное государственное бюджетное учреждение

высшего образования

**«Тверской государственный технический университет»**

(ТвГТУ)

Кафедра программного обеспечения

**Отчет по лабораторной работе №4**

по дисциплине: «Теория алгоритмов»

|  |
| --- |
| Выполнил:  студент группы  Б.ПИН.РИС-22.06  Мельников М.Н. |
| Проверила:  старший преподаватель  кафедры ПО  Корнеева Е.И. |

Тверь 2024

Описание задачи.

Реализовать 3 алгоритма решения задач по варианту №17.

**Требования:**

* Решить 3 задачи своего варианта
* Исключительные ситуации должны обрабатываться
* Оформите README.md.

Репозиторий.

[TheoryAlgLabs/Labs/Lab4 at main · Zawiaha/TheoryAlgLabs](https://github.com/Zawiaha/TheoryAlgLabs/tree/main/Labs/Lab2)

Описание проделанной работы.

# Задача 1.

Имеется строка, содержащая цифры:

*d = "0123456789"*

Необходимо записать генератор, который бы формировал и выдавал все

сочетания из трех цифр. Выдавать он должен именно целые числа, а не строки. Вывести первые 50 сгенерированных значений на экран.

# Алгоритм –

def threeDigitCombinations(d):

    for combo in combinations(d, 3):

        if combo[0] == "0":

            continue

        yield int(''.join(combo))

def printDigitCombination():

    number = "0123456789"

    gen = threeDigitCombinations(number)

    for \_ in range(50):

        print(next(gen))

# Задача 2.

Определите функцию-генератор, которая бы возвращала значения некоторой функции f(x) для значений x∈[a;b] с шагом 0,01. Величины a, b, а также функцию f, определить как параметры функции-генератора.

Вызовите функцию-генератор со значениями a = −20; b = 100 для функции *f(x) = −1,5x + 2*, которую определите как лямбда-функцию непосредственно в аргументе функции-генератора. Выведите на экран первые 20 значений, возвращенные функцией-генератором. Какие-либо коллекции в программе не использовать.

# Алгоритм –

def generateFunctionValues(a, b, func):

    x = a

    while x <= b:

        yield func(x)

        x += 0.01

def printFunctionValues():

    a = -20

    b = 100

    f = lambda x: -1.5 \* x + 2

    gen = generateFunctionValues(a, b, f)

    for \_ in range(20):

        print(next(gen))

# Задача 3.

На вход функции с именем get\_sort поступает словарь, например, такой:

*d = {'cat': 'кот', 'horse': 'лошадь', 'tree': 'дерево', 'dog': 'собака', 'book': 'книга'}*

Необходимо отсортировать словарь d по убыванию ключей (лексикографическая сортировка строк) и возвратить список из соответствующих значений ключей словаря. Например, для указанного словаря d, результатом должен быть список:

*['дерево', 'лошадь', 'собака', 'кот', 'книга']*

Вызовите функцию get\_sort для заданного в программе произвольного словаря dсt и выведите возвращенный ею список на экран.

# Алгоритм –

def getSort(d):

    return [d[key] for key in sorted(d.keys(), reverse=True)]

def printSortedDict():

    d = {'cat': 'кот', 'horse': 'лошадь', 'tree': 'дерево', 'dog': 'собака', 'book': 'книга'}

    sortedValues = getSort(d)

    print(sortedValues)

Вывод

В ходе лабораторной работы было решено 3 задачи, которые познакомили нас с функциями-генераторами. Оператором yield.

Используемые материалы.

<https://docs.python.org/> - python документация