|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»**  **(МГТУ им. Н. Э. Баумана)** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | «Информатика и системы управления » (ИУ) |
| КАФЕДРА | «Системы обработки информации и управления » (ИУ5) |

Отчёт по лабораторной работе №1

По курсу «Базовые компоненты интернет-технологий»

|  |  |
| --- | --- |
| Студент:  Козлов Егор Васильевич, группа ИУ5-32Б (2 курс) | (подпись, дата) |
| Проверил:  Гапанюк Юрий Евгеньевич | (подпись, дата) |

Москва, 2021

### **Задание:**

Разработать программу для решения [биквадратного уравнения.](https://ru.wikipedia.org/wiki/Уравнение_четвёртой_степени" \l "Биквадратное_уравнение)

1. Программа должна быть разработана в виде консольного приложения на языке Python.
2. Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).
3. Коэффициенты А, В, С могут быть заданы в виде параметров командной строки ([вариант задания параметров приведен в конце файла с примером кода](https://github.com/ugapanyuk/BKIT_2021/blob/main/code/lab1_code)). Если они не заданы, то вводятся с клавиатуры в соответствии с пунктом 2. [Описание работы с параметрами командной строки.](https://realpython.com/python-command-line-arguments/" \l "the-command-line-interface)
4. Если коэффициент А, В, С введен или задан в командной строке некорректно, то необходимо проигнорировать некорректное значение и вводить коэффициент повторно пока коэффициент не будет введен корректно. Корректно заданный коэффициент - это коэффициент, значение которого может быть без ошибок преобразовано в действительное число.

**Текст программы:**

import sys

import math

def get\_coef(index, prompt):

    '''

    Читаем коэффициент из командной строки или вводим с клавиатуры

    Args:

        index (int): Номер параметра в командной строке

        prompt (str): Сообщение с приглашением для ввода коэффицента

    Returns:

        float: Коэффициент квадратного уравнения

    '''

    try:

        # Пробуем прочитать коэффициент из командной строки

        coef\_str = sys.argv[index]

    except:

        # Вводим с клавиатуры

        print(prompt)

        coef\_str = input()

    # Переводим строку в действительное число

    while True:

        try:

            coef = float(coef\_str)

            break

        except ValueError:

            print(prompt)

            coef\_str = input()

    return coef

def get\_roots(a, b, c):

    '''

    Вычисление корней биквадратного уравнения

    Args:

        a (float): коэффициент А

        b (float): коэффициент B

        c (float): коэффициент C

    Returns:

        list[float]: Список корней

    '''

    roots = []

    #вычисление квадратного уравнения полученного после замены

    D = b\*b - 4\*a\*c

    print(D)

    if D == 0.0:

        root = -b / (2.0\*a)

        if b == 0:

            root = 0

        roots.append(root)

    elif D > 0.0:

        sqD = math.sqrt(D)

        root1 = (-b + sqD) / (2.0\*a)

        root2 = (-b - sqD) / (2.0\*a)

        if root1>0:

            base\_root1 = math.sqrt(root1)

            base\_root2 = - math.sqrt(root1)

            roots.append(base\_root1)

            roots.append(base\_root2)

        elif root1 == 0:

            base\_root1 = 0

            roots.append(base\_root1)

        if root2>0:

            base\_root3 = math.sqrt(root2)

            base\_root4 = - math.sqrt(root2)

            roots.append(base\_root3)

            roots.append(base\_root4)

        elif root2 == 0:

            base\_root2 = 0

            roots.append(base\_root2)

    return roots

def main():

    '''

    Основная функция

    '''

    a = get\_coef(1, 'Введите коэффициент А:')

    b = get\_coef(2, 'Введите коэффициент B:')

    c = get\_coef(3, 'Введите коэффициент C:')

    # Вычисление корней

    roots = get\_roots(a,b,c)

    # Вывод корней

    len\_roots = len(roots)

    if len\_roots == 0:

        print('Нет корней')

    elif len\_roots == 1:

        print('Один корень: {}'.format(roots[0]))

    elif len\_roots == 2:

        print('Два корня: {} и {}'.format(roots[0], roots[1]))

    elif len\_roots == 4:

        print('4 корня: {} и {} и {} и {}'.format(roots[0], roots[1], roots[2], roots[3]))

# Если сценарий запущен из командной строки

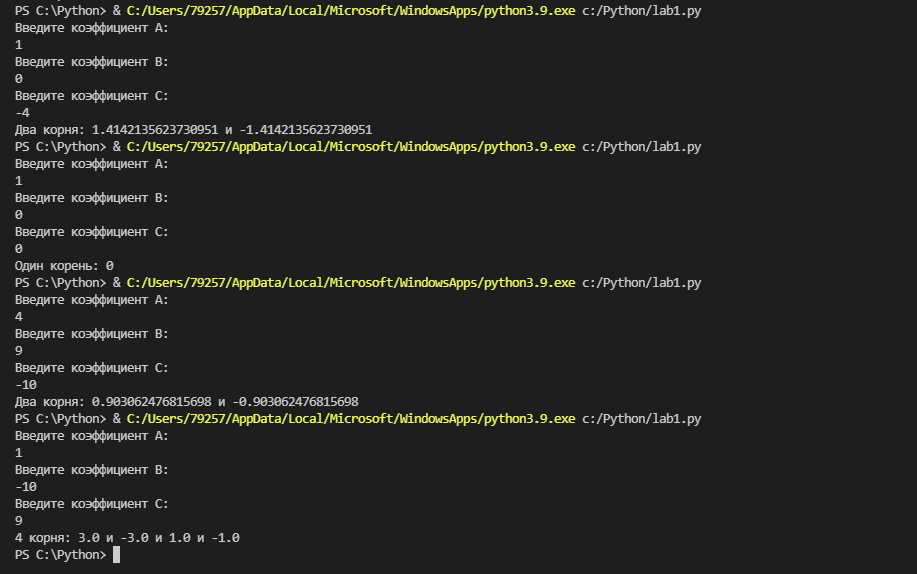
if \_\_name\_\_ == "\_\_main\_\_":

    main()

# Пример запуска

# qr.py 1 0 -4

**Примеры выполнения:**

****

# Задание:

Разработать программу. имитирующую работу цифровых часов для отображения времени с точностью до одной секунды Изменение показаний цифровых часов в соответствии ходом времени должны отображаться в одной строке потока стандартного вывода(stdout) формат вывода должен предусматривать отображение числа полных часов, минут и секунд, которые идентифицируют текущий момент времени и разделены символом двоеточия. Обновление показаний часов должно происходить с периодичностью одна секунда. При этом модифицироваться должны только те цифры строки вывода, значения которых изменились относительно предыдущего показания часов. Начальное показание часов должно соответствовать моменту вызова программы или передаваться ей через аргумент командной строки, спецификация которой должна быть идентична формату отображения показаний часов. Остановки часов должна имитироваться путем интерактивного прерывания работы программы по сигналу от клавиатуры.

# Входные данные:

# Никаких или время в формате hh:mm:ss.

# Описание алгоритма:

# При запуске программы пользователь вводит в качестве двух аргументов:

# 1) название файла, откуда будут считываться символы

# 2)диапазон кодировок символов через “ –“.

# В функции *main()* в начале осуществляется проверка на кол-во ведённых аргументов. Если пользователь забыл ввести название файла и диапазон, или ввёл больше, чем требуется, *отображается информационное сообщение в потоке протокола стандартной диагностики* cerr принадлежащей библиотеке iostream классу ostream и программа завершатся с кодом -1.

# Дальше создаётся переменная input класса ifstream, которая связывается с именем файла, который указан argv [1]. Команда open вызывается.

# Если файл успешно открыт, переменная input содержит ссылку файла, в противном случае 0. Если файл не получается открыть, выводится так же информационное сообщение с помощью cerr и программа завершается с кодом -2.

# Считываем argv[2], где хранится диапазон и заносим в переменные left и right.

# Так как диапазон указывается через символ «-» между цифрами, то запись sscanf будет представлять такой вид : “%d-%d”

# Проверяем на корректность введённого диапазона . Если left <0 и right>0, то выводится информационное сообщение через cerr и программа завершается с кодом -3.

# Создаём массив sym, в котором будем рассматривать максимальное количество повторов символов. Размер массива sym равен разнице между правой и левой гранью диапазона. Данный размер массива позволяет не взаимодействовать с кодами символов, которые не вошли в указанный диапазон.

# С помощью функции get, *которая является методом бесформатного ввода символов файловых потоков*, посимвольно считываем данные из файла, попутно увеличивая счётчик кол-во символов данной кодировки и сравнивая ее с максимальной частотой.

# Потом в цикле for реализуется вывод более часто встречаемых кодов и выводится с помощью cerr.

# Программа завершается с кодом возврата 0, что соответствует ее успешному завершению.

# Результат:

# Вывод времени в формате hh:mm:ss.

# Приложение 1. Текст программы.

# /\*Лабораторная работа №8 по ООП к 13.05.2021, РК6-22Б, Козлов Егор Васильевич

# #include <stdio.h>

# #include <time.h>

# #include <stdlib.h>

# #include <unistd.h>

# void clocks(int hours1, int hours2, int minutes1, int minutes2, int seconds1, int seconds2) {

# int flag=2;

# while(1) {

# if(seconds2>9) {

# seconds2=0;

# seconds1+=1;

# flag=1;

# if(seconds1>5){

# seconds1=0;

# minutes2+=1;

# flag=2;

# if(minutes2>9) {

# minutes2=0;

# minutes1+=1;

# flag=3;

# if(minutes1>5){

# minutes1=0;

# hours2+=1;

# flag=4;

# if(hours2>9) {

# hours2=0;

# hours1+=1;

# flag=5;

# } } } }

# if((hours2==4) && (hours1==2)){

# hours1=0;

# hours2=0;

# flag=6; } }

# switch(flag){

# case 1: printf("\r\t\b\b%d", seconds1); flag=0;

# case 2: printf("\r\t\b\b\b%d", minutes2); flag=0;

# case 3: printf("\r\t\b\b\b\b%d", minutes1); flag=0;

# case 4: printf("\r\t\b\b\b\b\b%d", hours2); flag=0;

# case 5: printf("\r\t\b\b\b\b\b\b%d", hours1); flag=0;

# case 6: printf("\r%d%d:%d%d:%d", hours1, hours2, minutes1, minutes2, seconds1); flag=0;

# default: printf("\r\t\b%d", seconds2++); }

# fflush(stdout); //сбрасывается буфер стандартного потока вывода

# sleep(1); }}

# int main(int argc, char\* argv[]) {

# if(argc>2){

# printf("Неверное количество аргументов\n");

# exit(1); }

# int ct, hours1, hours2, minutes1, minutes2, seconds1, seconds2;

# if(argc==1){

# ct = time(NULL);

# int hours=ct/3600%24+3;

# hours1 = (ct/3600%24+3)/10;

# hours2 = (ct/3600%24+3)%10;

# minutes1 = (ct/60%60)/10;

# minutes2 = (ct/60%60)%10;

# seconds1 = (ct%60)/10;

# seconds2 = (ct%60)%10;

# if(hours>23) {hours2%=4; hours1=0; } }

# else{

# if((sscanf(argv[1], "%1d%1d:%1d%1d:%1d%1d", &hours1, &hours2, &minutes1, &minutes2, &seconds1, &seconds2)) != 6){

# while( ((hours1==2) && (hours2>3)) || (hours1<0) || (hours1>2) || (hours2<0) || (minutes1>=6) || (minutes2<0) || (minutes1<0) || (seconds1>=6) || (seconds2<0) || (seconds1<0) ){

# printf("Ваш ввод неверен, введите заново\n");

# scanf("%1d%1d:%1d%1d:%1d%1d", &hours1,&hours2, &minutes1, &minutes2, &seconds1, &seconds2);

# } }

# while( ((hours1==2) && (hours2>3)) || (hours1<0) || (hours1>2) || (hours2<0) || (minutes1>=6) || (minutes2<0) || (minutes1<0) || (seconds1>=6) || (seconds2<0) || (seconds1<0) ){

# printf("Ваш ввод неверен, введите заново\n");

# scanf("%1d%1d:%1d%1d:%1d%1d", &hours1,&hours2, &minutes1, &minutes2, &seconds1, &seconds2);

# } }

# if(minutes1 == 6) { minutes1 = 0; }

# if(seconds1 == 6) { seconds1 = 0; }

# if((hours1 == 2) && (hours2==4)) { hours1=0; hours2=0; }

# clocks(hours1, hours2, minutes1, minutes2, seconds1, seconds2);

# return 0;

# }

# Приложение 2. Результаты тестирования:

# №1

# Запуск программы: *./a.out file1.txt 77-98*

# Результат:

# 97 3

# №2

# Запуск программы: *./a.out file1.txt 0-250*

# Результат:

# 10 6

# №3

# Данные внутри файла:

# aaa

# bb

# ccc

# Запуск программы: *./a.out file1.txt 456-23*

# Результат:

# Please, input correct range

# 

# Литература:

1. Учебное пособие«Прикладное программирование на С++» Т.М. Волосатова, С.В. Родионов, Д.Т. Шварц
2. Бьёрн Страуструп «Программирование: принципы и практика с использованием С++» 2 издание.
3. Бьёрн Страуструп «Язык программирования С++» второе дополненное издание
4. База и Генератор Образовательных Ресурсов - <http://bigor.bmstu.ru/>