**НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ** **УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ** **«МОСКОВСКИЙ ФИНАНСОВО-ПРОМЫШЛЕННЫЙ УНИВЕРСИТЕТ** **“СИНЕРГИЯ”»**

|  |  |  |
| --- | --- | --- |
| **Факультет/Институт** |  | Информационных технологий |
|  |  | (наименование факультета/ Института) |
| **Направление/специальность** |  | Информационные технологии и программирование |
| **подготовки:** |  | (код и наименование направления /специальности подготовки) |
| **Форма обучения:** |  | Очная |
|  |  | (очная, очно-заочная, заочная) |
|  |  |  |

**Отчет по практической работе № 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **на тему** |  | Разработка тестового сценария проекта | | |
|  |  | (наименование темы) | | |
|  |  |  | | |
| **по дисциплине** | | |  | Тестирование информационных систем |
|  | | |  | (наименование дисциплины) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Обучающийся** |  | Алимбеков Эрзат Данярович |  |  |
|  |  | (ФИО) |  | (подпись) |
| **Группа** |  | ДКИП-481 |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Преподаватель** |  | Авдеенков В.А. |  |  |
|  |  | (ФИО) |  | (подпись) |

**Москва 2025 г.**

**Лабораторный практикум 1**

**Задание:**

1. Написать программу, аналогичную Ч. 1, но уравнение кубическое, a⋅x³+b⋅х²+c⋅х+d=0;
2. Составить минимальный набор тестов (наборов коэффициентов a, b, c и d) для проверки работоспособности программы для разных случаев. Тестовые наборы должны давать следующие ситуации с корнями:
   * При a ≠ 0 (кубическое уравнение):
     + единственный вещественный корень;
     + один вещественный и два комплексных корня;
     + два вещественных корня;
     + три вещественных корня;
   * При a = 0 (квадратное, линейное или особые случаи) – аналогично Ч. 1.

**Код**

import math

def clean\_complex(z, eps=1e-10):

    real = z.real

    imag = z.imag

    if abs(imag) < eps:

        imag = 0.0

    if abs(real) < eps:

        real = 0.0

    return complex(real, imag)

def print\_roots(roots):

    for i, root in enumerate(roots):

        if abs(root.imag) < 1e-10:

            print(f"x{i+1} = {root.real:.6f}")

        else:

            print(f"x{i+1} = {root.real:.6f} {('+' if root.imag >= 0 else '')}{root.imag:.6f}i")

try:

    a, b, c, d = [int(i) for i in input('Enter parameters: ').split()]

except:

    print('Invalid input')

    exit()

eps = 1e-10

if (abs(a) < eps):

    D = b\*\*2 - 4\*a\*c

    result = ''

    if (a==0 and b==0 and c==0):

        result = ['infinitely many solutions']

    elif (a==0 and b==0):

        result = ['no solutions']

    elif (a==0):

        result = [-c/b]

    else:

        if (D < 0):

            result = [f"{-b} + i\*{round(abs(D)\*\*0.5, 2)} / {2\*a}", f"{-b} - i\*{round(abs(D)\*\*0.5, 2)} / {2\*a}"]

        elif (D == 0):

            result = [-b / 2 \* a]

        else:

            result = [-b + D\*\*0.5/ 2 \* a, -b - D\*\*0.5/ 2 \* a]

    if (len(result) > 1):

        print(f"x1 = {result[0]}\nx2 = {result[1]}")

    else:

        print(f"x = {result[0]}")

else:

    A = b / a

    B = c / a

    C = d / a

    p = B - A\*\*2/3.0

    q = C + (2\*(A\*\*3) - 9\*A\*B)/27.0

    D = (q\*\*2/4.0) + (p\*\*3/27.0)

    if (d > eps):

        u = -q/2.0 + math.sqrt(D)

        v = -q/2.0 - math.sqrt(D)

        u = u\*\*(1/3) if u>=0 else -((u)\*\*(1/3))

        v = v\*\*(1/3) if v>=0 else -((v)\*\*(1/3))

        y1 = 2\*u

        y2 = complex(-(u+v)/2, (u-v)\*math.sqrt(3)/2)

        y3 = complex(-(u+v)/2, -(u-v)\*math.sqrt(3)/2)

    elif (abs(D) <= eps):

        u = -q/2.0

        u = u\*\*(1/3) if u >= 0 else -(-(u)\*\*(1/3))

        y1 = 2\*u

        y2 = -u

        y3 = -u

    else:

        r = math.sqrt(-p\*p\*p/27.0)

        phi = math.acos(-q/(2\*r))

        y1 = 2 \* math.pow(r, 1/3) \* math.cos(phi/3)

        y2 = 2 \* math.pow(r, 1/3) \* math.cos((phi + 2\*math.pi)/3)

        y3 = 2 \* math.pow(r, 1/3) \* math.cos((phi + 4\*math.pi)/3)

    x1 = y1 - A/3.0

    x2 = y2 - A/3.0 if isinstance(y2, complex) else complex(y2 - A/3.0, 0)

    x3 = y3 - A/3.0 if isinstance(y3, complex) else complex(y3 - A/3.0, 0)

    roots = [clean\_complex(x1), clean\_complex(x2), clean\_complex(x3)]

    print\_roots(set(roots))

Рисунок 1 - Код на питоне

**Набор тестов**

* **Комплексные корни**

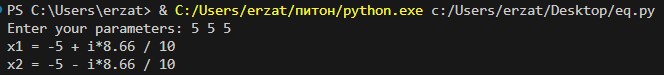


Рисунок 2 - Комплексный корень

* **Вещественные корни: нет, один, два, бесконечно много**

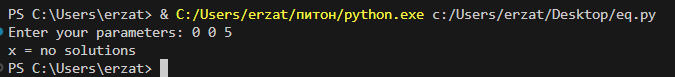


Рисунок 3 - Корней нет

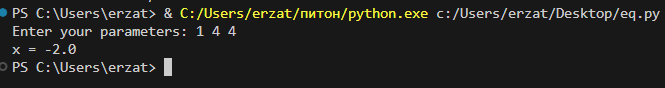


Рисунок 7 - Один корень

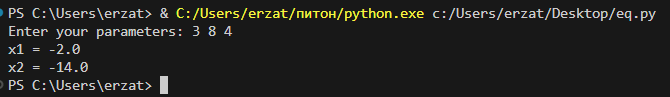


Рисунок 8 - Два корня

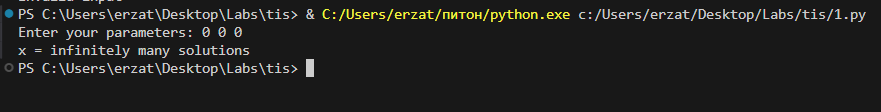


Рисунок 9 - Бесконечно много корней

* **Неверный ввод**

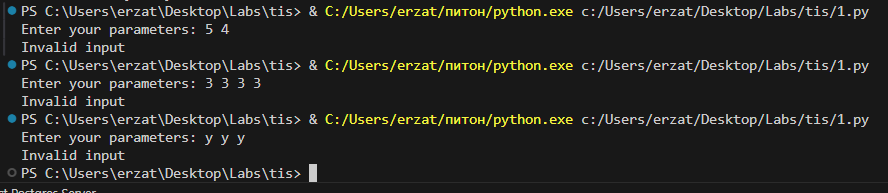


Рисунок 11 – Некорректный ввод