

**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №1

Выполнил:
Студент группы ИУ5-33Б
Левкович Леонид

Проверил:
Преподаватель каф. ИУ5
Гапанюк Ю. Е.

Москва, 2025 г.

Задание:

Задание:

Разработать программу для решения [биквадратного уравнения](#).

1. Программа должна быть разработана в виде консольного приложения на языке Python.
2. Программа осуществляет ввод с клавиатуры коэффициентов A, B, C, вычисляет дискриминант и **ДЕЙСТВИТЕЛЬНЫЕ** корни уравнения (в зависимости от дискриминанта).
3. Коэффициенты A, B, C могут быть заданы в виде параметров командной строки ([вариант задания параметров приведен в конце файла с примером кода](#)). Если они не заданы, то вводятся с клавиатуры в соответствии с пунктом 2. [Описание работы с параметрами командной строки](#).
4. Если коэффициент A, B, C 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.
5. Дополнительное задание 1 (*). Разработайте две программы на языке Python - одну с применением процедурной парадигмы, а другую с применением объектно-ориентированной парадигмы.
6. Дополнительное задание 2 (*). Разработайте две программы - одну на языке Python, а другую на любом другом языке программирования (кроме C++).

Листинг программы:

```
import sys
import math
import cmath

def get_coefficient(coef_name, cmd_value=None):
    value = cmd_value

    while True:
        if value is None:
            value = input(f"Введите коэффициент {coef_name}: ")

        try:
            return float(value)
        except ValueError:
            print(f"Некорректное значение '{value}'. Попробуйте снова.")
            value = None

def solve_quadratic(a, b, c):
    if a == 0:
        if b == 0:
            if c == 0:
                return "infinite" # Бесконечное множество решений
            else:
                return [] # Нет решений
        else:
            return [-c / b] # Линейное уравнение

    # Вычисляем дискриминант
    discriminant = b * b - 4 * a * c

    if discriminant > 0:
        # Два действительных корня
        sqrt_d = math.sqrt(discriminant)
        t1 = (-b + sqrt_d) / (2 * a)
        t2 = (-b - sqrt_d) / (2 * a)
        return [t1, t2]
    elif discriminant == 0:
        # Один действительный корень
        t = -b / (2 * a)
        return [t]
```

```

        return [t]
    else:
        # Комплексные корни
        sqrt_d = cmath.sqrt(discriminant)
        t1 = (-b + sqrt_d) / (2 * a)
        t2 = (-b - sqrt_d) / (2 * a)
        return [t1, t2]

def solve_biquadratic(a, b, c):
    print(f"\nРешаем биквадратное уравнение: {a}x⁴ + {b}x² + {c} = 0")

    # Делаем замену t = x²
    print("Делаем замену t = x², получаем квадратное уравнение:")
    print(f"{a}t² + {b}t + {c} = 0")

    # Решаем квадратное уравнение относительно t
    t_roots = solve_quadratic(a, b, c)

    if t_roots == "infinite":
        print("\nУравнение имеет бесконечное множество решений")
        return []

    if not t_roots:
        print("\nКвадратное уравнение не имеет решений")
        print("Биквадратное уравнение не имеет действительных корней")
        return []

    # Вычисляем дискриминант для информации
    if a != 0:
        discriminant = b * b - 4 * a * c
        print(f"\nДискриминант D = b² - 4ac = {b}² - 4·{a}·{c} = {discriminant}")

    print(f"\nКорни квадратного уравнения (t): {t_roots}")

    # Находим корни биквадратного уравнения
    x_roots_real = []
    x_roots_complex = []

    for i, t in enumerate(t_roots, 1):
        print(f"\nДля t{i} = {t}:")

        # Проверяем, является ли t действительным числом
        if isinstance(t, complex):
            if abs(t.imag) < 1e-10: # Практически действительное число
                t = t.real
            else:
                # t - комплексное, находим комплексные корни x
                x1 = cmath.sqrt(t)
                x2 = -x1
                x_roots_complex.extend([x1, x2])
                print(f" x = ±√{t} = ±{x1} (комплексные корни)")
                continue

        if t > 0:
            # t положительное - два действительных корня
            x1 = math.sqrt(t)
            x2 = -x1
            x_roots_real.extend([x1, x2])
            print(f" x = ±√{t} = ±{x1:.6f} (действительные корни)")

        elif t == 0:
            # t = 0 - один корень x = 0
            x_roots_real.append(0)
            print(f" x = 0 (действительный корень)")

        else:
            # t отрицательное - два мнимых корня
            x1 = complex(0, math.sqrt(-t))

```

```

        x2 = -x1
        x_roots_complex.extend([x1, x2])
        print(f"  x = ±√({t}) = ±{math.sqrt(-t):.6f}i (мнимые корни)")

    return x_roots_real, x_roots_complex

def main():
    print("=" * 60)
    print("РЕШЕНИЕ БИКВАДРАТНОГО УРАВНЕНИЯ ax⁴ + bx² + c = 0")
    print("=" * 60)

    # Получаем коэффициенты из командной строки или с клавиатуры
    a_cmd = sys.argv[1] if len(sys.argv) > 1 else None
    b_cmd = sys.argv[2] if len(sys.argv) > 2 else None
    c_cmd = sys.argv[3] if len(sys.argv) > 3 else None

    if a_cmd or b_cmd or c_cmd:
        print("Обнаружены параметры командной строки")

    a = get_coefficient("A", a_cmd)
    b = get_coefficient("B", b_cmd)
    c = get_coefficient("C", c_cmd)

    print(f"\nВведенные коэффициенты: A = {a}, B = {b}, C = {c}")

    # Проверка на нулевой старший коэффициент
    if a == 0:
        print("\nКоэффициент A = 0. Это не биквадратное уравнение.")
        if b == 0:
            if c == 0:
                print("Уравнение 0 = 0 имеет бесконечное множество решений")
            else:
                print(f"Уравнение {c} = 0 не имеет решений")
        else:
            # bx² + c = 0 => x² = -c/b
            print(f"Получаем уравнение {b}x² + {c} = 0")
            t = -c / b
            if t > 0:
                x = math.sqrt(t)
                print(f"Действительные корни: x₁ = {x:.6f}, x₂ = {-x:.6f}")
            elif t == 0:
                print("Действительный корень: x = 0")
            else:
                x = math.sqrt(-t)
                print(f"Комплексные корни: x₁ = {x:.6f}i, x₂ = {-x:.6f}i")
    return

    # Решаем биквадратное уравнение
    real_roots, complex_roots = solve_biquadratic(a, b, c)

    # Выводим результаты
    print("\n" + "=" * 60)
    print("РЕЗУЛЬТАТЫ:")
    print("=" * 60)

    if real_roots:
        # Удаляем дубликаты и сортируем
        real_roots = sorted(list(set(round(x, 10) for x in real_roots)))
        print(f"\nКоличество действительных корней: {len(real_roots)}")
        print("Действительные корни:")
        for i, root in enumerate(real_roots, 1):
            print(f"  x{i} = {root:.6f}")
    else:
        print("\nДействительных корней нет")

    if complex_roots:
        # Форматируем комплексные корни

```

```
print(f"\nКоличество комплексных корней: {len(complex_roots)}")
print("Комплексные корни:")
for i, root in enumerate(complex_roots, 1):
    if root.imag >= 0:
        print(f"  x{i} = {root.real:.6f} + {root.imag:.6f}i")
    else:
        print(f"  x{i} = {root.real:.6f} - {abs(root.imag):.6f}i")

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        print("\n\nПрограмма прервана пользователем")
    except Exception as e:
        print(f"\nОшибка: {e}")
```

Результат выполнения:

```
=====
```

РЕШЕНИЕ БИКВАДРАТНОГО УРАВНЕНИЯ $ax^4 + bx^2 + c = 0$

```
=====
```

Введите коэффициент A: 1

Введите коэффициент B: 1

1Введите коэффициент C:

Введенные коэффициенты: A = 1.0, B = 1.0, C = 1.0

Решаем биквадратное уравнение: $1.0x^4 + 1.0x^2 + 1.0 = 0$

Делаем замену $t = x^2$, получаем квадратное уравнение:

$1.0t^2 + 1.0t + 1.0 = 0$

Дискриминант $D = b^2 - 4ac = 1.0^2 - 4 \cdot 1.0 \cdot 1.0 = -3.0$

Корни квадратного уравнения (t): $[-0.5+0.8660254037844386j], [-0.5-0.8660254037844386j]$

Для $t_1 = -0.5+0.8660254037844386j$:

$x = \pm\sqrt{(-0.5+0.8660254037844386j)} = \pm(0.5+0.8660254037844386j)$ (комплексные корни)

Для $t_2 = -0.5-0.8660254037844386j$:

$x = \pm\sqrt{(-0.5-0.8660254037844386j)} = \pm(0.5-0.8660254037844386j)$ (комплексные корни)

```
=====
```

РЕЗУЛЬТАТЫ:

```
=====
```

Действительных корней нет

Действительных корней нет

Количество комплексных корней: 4

Комплексные корни:

$x_1 = 0.500000 + 0.866025i$

$x_2 = -0.500000 - 0.866025i$

$x_3 = 0.500000 - 0.866025i$

$x_4 = -0.500000 + 0.866025i$

Process finished with exit code 0

