Отчет по лабораторной работе

# 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 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. Если они не заданы, то вводятся с клавиатуры.
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 (Python, процедурный подход)

biquadratic\_procedural.py

import sys

import math

def get\_coefficient(name, args, index):

    while True:

        try:

*# Try to retrieve coefficient from args*

            if len(args) > index:

                return float(args[index])

*# or ask for it*

            return float(input(f"Введите коэффициент {name}: "))

        except ValueError:

            print(f"Некорректное значение для {name}, попробуйте снова.")

*# Clear argument value for next try if providede*

            if len(args) > index:

                args[index] = None

def solve\_biquadratic(a, b, c):

*# Discriminant*

    d = b\*b - 4\*a\*c

    print(f"Дискриминант: {d}")

    if d < 0:

        print("Действительных корней нет.")

        return []

    if d > 1e16:

        print("Дискриминант слишком большой.")

        return []

*# Use formula*

    sqrt\_d = math.sqrt(d)

    x1 = (-b + sqrt\_d) / (2\*a)

    x2 = (-b - sqrt\_d) / (2\*a)

*# Get roots result*

    roots = [x1]

    if x1 != x2:

        roots = [x1, x2]

    return roots

def main():

*# Get coefficients*

    args = sys.argv

    a = get\_coefficient("A", args, 1)

    if a == 0:

        print("Коэффициент A в квадратном уравнении не может быть равен 0.")

        return

    b = get\_coefficient("B", args, 2)

    c = get\_coefficient("C", args, 3)

*# Solve*

    roots = solve\_biquadratic(a, b, c)

    if roots:

        print("Действительные корни:", roots)

        return

    print("Корней нет.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

# Текст программы (Python, ООП)

biquadratic\_oop.py

import sys

import math

class BiquadraticSolver:

    def \_\_init\_\_(self, a: float, b: float, c: float):

        self.a = a

        self.b = b

        self.c = c

    def discriminant(self):

        return self.b\*self.b - 4\*self.a\*self.c

    def solve(self):

        d = self.discriminant()

        print(f"Дискриминант: {d}")

        if d < 0:

            print("Действительных корней нет.")

            return []

        if d > 1e16:

            print("Дискриминант слишком большой.")

            return []

*# Use formula*

        sqrt\_d = math.sqrt(d)

        x1 = (-self.b + sqrt\_d) / (2\*self.a)

        x2 = (-self.b - sqrt\_d) / (2\*self.a)

*# Get roots result*

        roots = [x1]

        if x1 != x2:

            roots = [x1, x2]

        return roots

*# Helper function*

def get\_coefficient(name: str, args: [str], index: int):

    while True:

        try:

*# Try to retrieve coefficient from args*

            if len(args) > index:

                return float(args[index])

*# or ask for it*

            return float(input(f"Введите коэффициент {name}: "))

        except ValueError:

            print(f"Некорректное значение для {name}, попробуйте снова.")

*# Clear argument value for next try if providede*

            if len(args) > index:

                args[index] = None

def main():

*# Get coefficients*

    args = sys.argv

    a = get\_coefficient("A", args, 1)

    if a == 0:

        print("Коэффициент A в квадратном уравнении не может быть равен 0.")

        return

    b = get\_coefficient("B", args, 2)

    c = get\_coefficient("C", args, 3)

*# Initialize class*

    solver = BiquadraticSolver(a, b, c)

*# solve*

    roots = solver.solve()

    if roots:

        print("Действительные корни:", roots)

        return

    print("Корней нет.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

# Текст программы (Go)

biquadratic.go

package main

import (

    "fmt"

    "math"

    "os"

    "strconv"

)

func getCoefficient(name string, args []string, index int) float64 {

    for {

        if len(args) > index {

            value, err := strconv.ParseFloat(args[index], 64)

            if err == nil {

                fmt.Printf("Коэффициент %s=%f\n", name, value)

                return value

            }

            fmt.Println("Некорректное значение для", name)

        }

        fmt.Printf("Введите коэффициент %s: ", name)

        var value float64

        if \_, err := fmt.Scan(&value); err == nil {

            return value

        }

        fmt.Println("Некорректное значение, попробуйте снова.")

    }

}

func solveBiquadratic(a, b, c float64) []float64 {

*// Discriminant*

    d := b\*b - 4\*a\*c

    fmt.Printf("Дискриминант = %f\n", d)

*// Validate discriminant*

    if d < 0 {

        fmt.Println("Действительных корней нет.")

        return []float64{}

    }

*// Use formula*

    sqrtD := math.Sqrt(d)

    x1 := (-b + sqrtD) / (2 \* a)

    x2 := (-b - sqrtD) / (2 \* a)

*// Get roots result as*

    roots := []float64{x1}

    if x1 != x2 {

        roots = []float64{x1, x2}

    }

    return roots

}

func main() {

*// Get coefficients*

    args := os.Args

    a := getCoefficient("A", args, 1)

*// Validate A*

    if a == 0 {

        fmt.Println("Коэффициент A в квадратном уравнении не может быть равен 0.")

        return

    }

*// No rules for B & C*

    b := getCoefficient("B", args, 2)

    c := getCoefficient("C", args, 3)

*// Solve*

    roots := solveBiquadratic(a, b, c)

*// If ANY roots*

    if len(roots) > 0 {

        fmt.Println("Действительные корни:", roots)

        return

    }

    fmt.Println("Корней нет.")

}

Скриншоты выполнения программы:

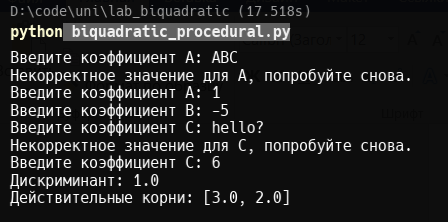


Рис. 1. Biquadratic\_procedural.py запуск программы.

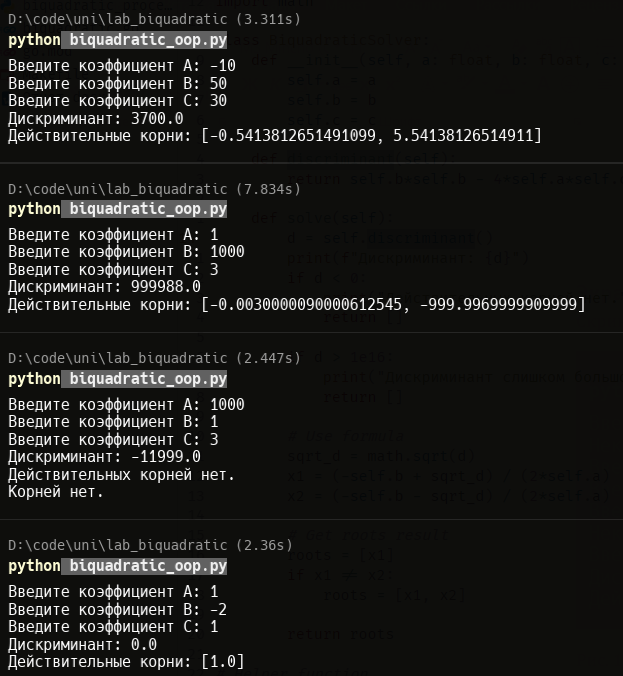


Рис. 2 Biquadratic\_oop.py запуск программы.

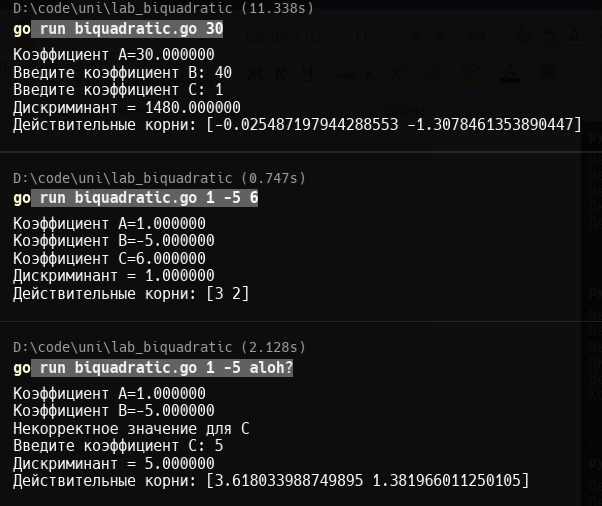


Рис. 3 Biquadratic.go запуск программы.