Московский государственный технический университет им. Н.Э. Баумана

Факультет ИУ
Кафедра ИУ5

Курс «Основы информатики»
Отчет по лабораторной работе №3

Выполнил студент группы ИУ5-33Б:
Хасанова К.М.
Подпись и дата:

Проверил преподаватель каф.:
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2024 г

Описание задания

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 2 (*). Разработайте две программы - одну на языке Python, а другую на любом другом языке программирования (кроме C++).

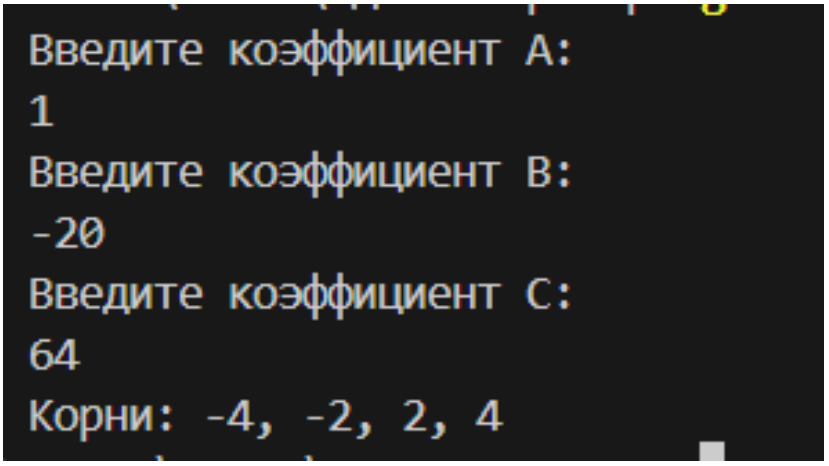
Текст программы

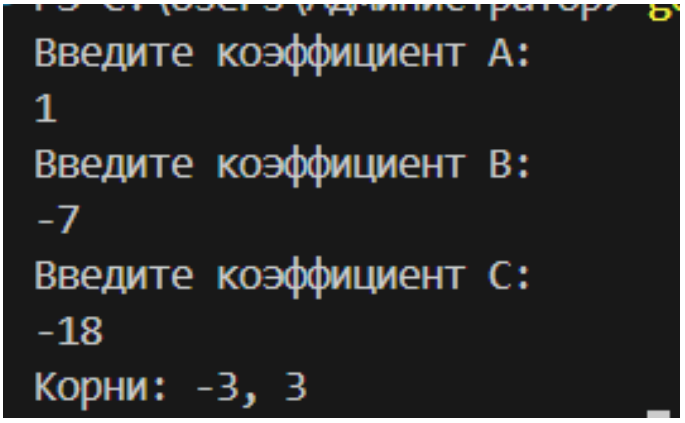
```go
package main

import (
    "fmt"
    "math"
    "os"
    "sort"
    "strconv"
)

// SquareRoots структура для хранения коэффициентов и корней уравнения
type SquareRoots struct {
    coefA, coefB, coefC float64
    numRoots            int
    rootsList           map[float64]bool
}

// getCoef пытается получить коэффициент из аргумента командной строки или с помощью ввода
с клавиатуры
func (sr *SquareRoots) getCoef(index int, prompt string) float64 {
    if len(os.Args) > index {
        coefStr := os.Args[index]
```

```
        coef, err := strconv.ParseFloat(coefStr, 64)
        if err == nil {
            return coef
        }
    }
    // Если аргумент командной строки не предоставлен, спрашиваем с клавиатуры
    fmt.Println(prompt)
    var coefStr string
    fmt.Scanln(&coefStr)
    coef, err := strconv.ParseFloat(coefStr, 64)
    if err != nil {
        fmt.Println("Ошибка ввода. Пожалуйста, введите число.")
        os.Exit(1)
    }
    return coef
}

// getCoefs считывает все коэффициенты
func (sr *SquareRoots) getCoefs() {
    sr.coefA = sr.getCoef(1, "Введите коэффициент A:")
    sr.coefB = sr.getCoef(2, "Введите коэффициент B:")
    sr.coefC = sr.getCoef(3, "Введите коэффициент C:")
}

// calculateRoots вычисляет корни уравнения и заполняет rootsList
func (sr *SquareRoots) calculateRoots() {
    a := sr.coefA
    b := sr.coefB
    c := sr.coefC

    // Инициализация map для хранения корней
    sr.rootsList = make(map[float64]bool)

    // Вычисление дискриминанта
    D := b*b - 4*a*c

    if D > 0 {
        t1 := (-b + math.Sqrt(D)) / (2 * a)
        t2 := (-b - math.Sqrt(D)) / (2 * a)

        // Для t1 и t2 вычисляем корни
        if t1 >= 0 {
            sr.rootsList[math.Sqrt(t1)] = true
            sr.rootsList[-math.Sqrt(t1)] = true
        }
        if t2 >= 0 {
            sr.rootsList[math.Sqrt(t2)] = true
            sr.rootsList[-math.Sqrt(t2)] = true
        }

    } else if D == 0 {
        t := -b / (2 * a)
        if t >= 0 {
            sr.rootsList[math.Sqrt(t)] = true
```

```
            sr.rootsList[-math.Sqrt(t)] = true
        }
    }

    // Устанавливаем количество уникальных корней
    sr.numRoots = len(sr.rootsList)
}

// printRoots выводит корни уравнения
func (sr *SquareRoots) printRoots() {
    // Преобразуем map в срез и сортируем корни
    var roots []float64
    for root := range sr.rootsList {
        roots = append(roots, root)
    }
    // Сортируем корни для удобства вывода
    sort.Float64s(roots)

    // Печать корней
    if len(roots) == 0 {
        fmt.Println("Нет действительных корней")
    } else {
        fmt.Printf("Корни: ")
        for i, root := range roots {
            if i > 0 {
                fmt.Print(", ")
            }
            fmt.Print(root)
        }
        fmt.Println()
    }
}

func main() {
    // Создаем объект структуры
    r := SquareRoots{}
    // Получаем коэффициенты
    r.getCoefs()
    // Вычисляем корни
    r.calculateRoots()
    // Печатаем корни
    r.printRoots()
}
```

Выполнение программы


Введите коэффициент A:
1
Введите коэффициент B:
-20
Введите коэффициент C:
64
Корни: -4, -2, 2, 4


Введите коэффициент A:
1
Введите коэффициент B:
-7
Введите коэффициент C:
-18
Корни: -3, 3