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
«Изучение основных конструкций языка Python»

Выполнил:
студент группы ИУ5-33Б
Сикоринский Артемий

Проверил:
Гапанюк Е.Ю.

2024 г.

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.

Текст программы

Main.py

```
import sys

def get_coef(index, prompt):
    try:
        coef_str = sys.argv[index]
    except:
        print(prompt)
        coef_str = input()
    return coef_str


def Dis(A, B, C):
    d = B ** 2 - 4 * A * C
    return d


def roots(A, B, C):
    D = Dis(A, B, C)
    if D > 0:
        return [(-B + (D ** (0.5))) / 2 * A, (-B - (D ** (0.5))) / 2 * A]
    elif D == 0:
        return [(-B) / 2 * A]
    else:
        return []


def main():
    A = get_coef(1, "A: ")
    B = get_coef(2, "B: ")
    C = get_coef(3, "C: ")
```

```
    try:
        A = float(A)
        B = float(B)
        C = float(C)
    except:
        print("Error!")
        return main()
    l_roots = roots(A, B, C)
    print(f"Discriminant: {Dis(A, B, C)}")
    if len(l_roots) == 0:
        print("No roots.")
    elif len(l_roots) == 1:
        print(f"One root: {l_roots[0]}")
    else:
        print(f"Two roots: {l_roots[0]}, {l_roots[1]}")


if __name__ == "__main__":
    main()
```

main2.py

```
import sys

class FindRoots:
    def __init__(self, A=0, B=0, C=0):
        self.A = A
        self.B = B
        self.C = C
    @staticmethod
    def get_coef(index, prompt):
        try:
            coef_str = sys.argv[index]
        except:
            print(prompt)
            coef_str = input()
        return coef_str

    def input_coef(self):
        A = self.get_coef(1, "A: ")
        B = self.get_coef(2, "B: ")
        C = self.get_coef(3, "C: ")
        try:
            A = float(A)
            B = float(B)
            C = float(C)
        except:
            print("Error!")
            return self.input_coef()
        self.__init__(A, B, C)

    def Dis(self):
        d = self.B ** 2 - 4 * self.A * self.C
        return d

    def roots(self):
        D = self.Dis()
        if D > 0:
            return [(-self.B + (D ** (0.5))) / 2 * self.A, (-self.B - (D ** (0.5)))
/ 2 * self.A]
        elif D == 0:
            return [(-self.B) / 2 * self.A]
        else:
```

```
            return []
    def print_roots(self):
        l_roots = self.roots()
        print(f"Discriminant: {self.Dis()}")
        if len(l_roots) == 0:
            print("No roots.")
        elif len(l_roots)==1:
            print(f"One root: {l_roots[0]}")
        else:
            print(f"Two roots: {l_roots[0]}, {l_roots[1]}")


def main():
   obj = FindRoots()
   obj.input_coef()
   obj.print_roots()

if __name__ == "__main__":
    main()
```

на Go:

```
package main

import (
 "bufio"
 "fmt"
 "math"
 "os"
 "strconv"
 "strings"
)

func getCoef(index int, prompt string) string {
 if len(os.Args) > index {
  return os.Args[index]
 }
 fmt.Print(prompt)
 reader := bufio.NewReader(os.Stdin)
 coefStr, _ := reader.ReadString('\n')
 return strings.TrimSpace(coefStr)
}

func dis(a, b, c float64) float64 {
 return b*b - 4*a*c
}

func roots(a, b, c float64) []float64 {
 d := dis(a, b, c)
 if d > 0 {
  x1 := (-b + math.Sqrt(d)) / (2 * a)
  x2 := (-b - math.Sqrt(d)) / (2 * a)
  return []float64{x1, x2}
 } else if d == 0 {
  x := (-b) / (2 * a)
  return []float64{x}
```

```
 } else {
  return []float64{}
 }
}

func main() {
 for {
  aStr := getCoef(1, "A: ")
  bStr := getCoef(2, "B: ")
  cStr := getCoef(3, "C: ")
  a, errA := strconv.ParseFloat(aStr, 64)
  b, errB := strconv.ParseFloat(bStr, 64)
  c, errC := strconv.ParseFloat(cStr, 64)

  if errA != nil || errB != nil || errC != nil {
   fmt.Println("Error! Invalid input. Please provide numbers.")
   continue // Restart the loop
  }

  lRoots := roots(a, b, c)
  fmt.Printf("Discriminant: %f\n", dis(a, b, c))

  if len(lRoots) == 0 {
   fmt.Println("No roots.")
  } else if len(lRoots) == 1 {
   fmt.Printf("One root: %f\n", lRoots[0])
  } else {
   fmt.Printf("Two roots: %f, %f\n", lRoots[0], lRoots[1])
  }
  break // Exit after successful calculation
 }
}
```

Экранные формы с примерами выполнения программы

```
➜  lab1 (main) python3 lab1.py
Введите коэффициент A: 1
Введите коэффициент B: 11
Введите коэффициент C: 10
Дискриминант: 81.0
Нет корней
➜  lab1 (main) python3 lab1.py 10 0
Введите коэффициент C: 0
Дискриминант: 0.0
Один корень: 0
➜  lab1 (main) python3 lab1.py 1 -2 -8
Дискриминант: 36.0
Два корня: -2.0, 2.0
➜  lab1 (main) python3 lab1.py -4 16 0
Дискриминант: 256.0
Три корня: -2.0, -0.0, 2.0
➜  lab1 (main) python3 lab1.py 1 -10 9
Дискриминант: 64.0
Четыре корня: -3.0, -1.0, 1.0, 3.0
➜  lab1 (main)
```