Московский государственный технический университет им. Н.Э. Баумана

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

«Парадигмы и конструкции языков программирования»
Отчет по лабораторной работе №1
«Основные конструкции языка Python»

Выполнил:
студент группы ИУ5-36Б

Мироненков Арсений Максимович

Подпись и дата:

Проверил:
преподаватель каф. ИУ5

Нардид А.Н.

Подпись и дата:

Москва, 2024

Цель лабораторной работы: изучение основных конструкций языка Python.

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++).

Такст программы:

с применением процедурной парадигмы

```
import sys
import math

def get_coef(index, prompt):
    try:
        coef_str = sys.argv[index]
    except:
        print(prompt)
        coef_str = input()
    coef = float(coef_str)
    return coef
def solve(a, b, c):
    result = []
    D = b**2-4*a*c
    if D<0:
        print('нет решений')
        return []
    elif D==0:
        root = -b/2*a
        result.append(root)
    elif D>0:
        root1 = (-b-math.sqrt(D))/2*a
        root2 = (-b+math.sqrt(D))/2*a
        result.append(root1)
```

```
        result.append(root2)
    return result
def main():
    a = get_coef(1, 'A')
    b = get_coef(2, 'B')
    c = get_coef(3, 'C')
    roots = solve(a,b,c)
    len_roots = len(roots)
    if len_roots == 0:
        print("/")
    elif len_roots ==1:
        print('Один корень: x = {}'.format(roots[0]))

    elif len_roots == 2:
        print('Два корня: x1 = {}, x2 = {}'.format(roots[0], roots[1]))
if __name__ == '__main__':
    main()
```

с применением объектно-ориентированной парадигмы

```
import sys
import math

class BiquadraticEquation:
    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c
    def solve(self):
        if self.a == 0:
            raise ValueError("Коэффициент A не должен быть равен 0.")
        print(f"Решаем уравнение {self.a}x^4 + {self.b}x^2 + {self.c} = 0")
        discriminant = self.b ** 2 - 4 * self.a * self.c
        print(f"Дискриминант: {discriminant}")
        if discriminant > 0:
            z1 = (-self.b + math.sqrt(discriminant)) / (2 * self.a)
            z2 = (-self.b - math.sqrt(discriminant)) / (2 * self.a)
            roots = []
            if z1 >= 0:
                roots.append(math.sqrt(z1))
                roots.append(-math.sqrt(z1))
            if z2 >= 0:
                roots.append(math.sqrt(z2))
                roots.append(-math.sqrt(z2))
            if roots:
                print(f"Действительные корни: {sorted(set(roots))}")
            else:
                print("Нет действительных корней.")
        elif discriminant == 0:
            z = -self.b / (2 * self.a)
            if z >= 0:
                print(f"Действительные корни: {math.sqrt(z)}, {-math.sqrt(z)}")
            else:
                print("Нет действительных корней.")
        else:
            print("Нет действительных корней.")
def get_coefficient(name):
    while True:
        try:
            value = float(input(f"Введите коэффициент {name}: "))
```

```
            return value
        except ValueError:
            print(f"Коэффициент {name} должен быть числом. Попробуйте снова.")
if len(sys.argv) == 4:
    try:
        a = float(sys.argv[1])
        b = float(sys.argv[2])
        c = float(sys.argv[3])
    except ValueError:
        print("Один или несколько параметров командной строки некорректны.")
        a = get_coefficient('A')
        b = get_coefficient('B')
        c = get_coefficient('C')
else:
    a = get_coefficient('A')
    b = get_coefficient('B')
    c = get_coefficient('C')
equation = BiquadraticEquation(a, b, c)
equation.solve()
```

На языке JavaScript

```
class BiquadraticEquation {
    constructor(a, b, c) {
        this.a = a;
        this.b = b;
        this.c = c;
    }

    solve() {
        if (this.a === 0) {
            throw new Error("Коэффициент A не должен быть равен 0.");
        }
        console.log(`Решаем уравнение ${this.a}x^4 + ${this.b}x^2 + ${this.c} = 0`);
        const discriminant = this.b ** 2 - 4 * this.a * this.c;
        console.log(`Дискриминант: ${discriminant}`);
        if (discriminant > 0) {
            const z1 = (-this.b + Math.sqrt(discriminant)) / (2 * this.a);
            const z2 = (-this.b - Math.sqrt(discriminant)) / (2 * this.a);
            let roots = [];
            if (z1 >= 0) {
                roots.push(Math.sqrt(z1));
                roots.push(-Math.sqrt(z1));
            }
            if (z2 >= 0) {
                roots.push(Math.sqrt(z2));
                roots.push(-Math.sqrt(z2));
            }
            if (roots.length > 0) {
                // Ensure roots are numbers and remove NaN
                const validRoots = roots.filter(root => typeof root === 'number'
&& !isNaN(root));
                const uniqueRoots = [...new Set(validRoots)].sort((a, b) => a - b);
                if(uniqueRoots.length > 0) {
                    console.log(`Действительные корни: ${uniqueRoots}`);
                } else {
                  console.log("Нет действительных корней.");
                }
            } else {
                console.log("Нет действительных корней.");
            }
```

```
        } else if (discriminant === 0) {
            const z = -this.b / (2 * this.a);
             if (z >= 0) {
                const root = Math.sqrt(z)
                 console.log(`Действительные корни: ${root}, ${-root}`);
             } else {
                console.log("Нет действительных корней.");
             }
        } else {
            console.log("Нет действительных корней.");
        }
    }
}
function getCoefficient(name) {
    while (true) {
        const value = prompt(`Введите коэффициент ${name}: `);
        const parsedValue = parseFloat(value);
        if (!isNaN(parsedValue)) {
            return parsedValue;
        }
        console.log(`Коэффициент ${name} должен быть числом. Попробуйте снова.`);
    }
}
function main() {
    let a, b, c;
    const args = process.argv.slice(2);
    if (args.length === 3) {
        try {
            a = parseFloat(args[0]);
            b = parseFloat(args[1]);
            c = parseFloat(args[2]);
            if (isNaN(a) || isNaN(b) || isNaN(c)) {
                console.log("Один или несколько параметров командной строки некорректны.")
                a = getCoefficient('A');
                b = getCoefficient('B');
                c = getCoefficient('C');
            }
        } catch (e) {
            console.log("Один или несколько параметров командной строки некорректны.")
            a = getCoefficient('A');
            b = getCoefficient('B');
            c = getCoefficient('C');
        }
    } else {
        a = getCoefficient('A');
        b = getCoefficient('B');
        c = getCoefficient('C');
    }
    const equation = new BiquadraticEquation(a, b, c);
    equation.solve();
}
main();
```

Вывод результата:

```
PS C:\Users\DNS\PycharmProjects> node fuuu.js 1 -5 4
Решаем уравнение 1x^4 + -5x^2 + 4 = 0
Дискриминант: 9
Действительные корни: -2,-1,1,2
```

```
PS C:\Users\DNS\PycharmProjects\pythonProject1> python main.py
Введите коэффициент A: 2
Введите коэффициент B: -4
Введите коэффициент C: 2
Решаем уравнение 2.0x^4 + -4.0x^2 + 2.0 = 0
Дискриминант: 0.0
Действительные корни: 1.0, -1.0
```

```
PS C:\Users\DNS\PycharmProjects\pythonProject1> python main1.py
A
2
B
-4
C
1
Два корня: x1 = 1.1715728752538097, x2 = 6.82842712474619
```