Харківський національний університет імені В. Н. Каразіна
Факультет комп'ютерних наук
Кафедра штучного інтелекту та програмного забезпечення

ЗВІТ
З ЛАБОРАТОРНОЇ РОБОТИ №7

дисципліна: «Крос-платформне програмування»

Виконав: студент групи КС23
Терещенко Євгеній Юрійович
Перевірив: доцент кафедри ІІПО
Споров Олександр. Є

Харків – 2024

Завдання 1.

Створіть розподілене клієнт/серверне програмне рішення, що працює з віддаленими об'єктами у відповідності до правил технології *RMI*. Серверна частина програми буде представлена віддаленим об'єктом, що отримує обчислювальні завдання від клієнтів, виконує їх та повертає отриманий результат. Завдання, з якими може працювати віддалений об'єкт — обчислювач, можуть бути створені і передані цьому серверному віддаленому об'єкту в будь-який момент під час роботи сервера. Організуйте роботу так, щоб байт-код, необхідний для виконання завдання, міг бути, при необхідності, автоматично "*завантажений*" віддаленим об'єктом засобами системи *RMI*. Клієнтська частина програми взаємодіє з користувачем, готує завдання для розрахунків, передає їх віддаленому серверному об'єкту та отримує результат обчислень.

Для перевірки роботи програмного рішення підготуйте до обчислень кілька задач:

- обчислення значення числа π із заданою великою точністю (більше ніж 16 значущих десяткових цифр). Для обчислення π скористуйтесь наступними формулами:
  - *Machine – like formula* (1961)
$$\frac{\pi}{4}=4\cdot\arctan\frac{1}{5}-\arctan\frac{1}{239};$$
  - *K. Takano formula* (1982)
$$\frac{\pi}{4}=12\cdot\arctan\frac{1}{49}+32\cdot\arctan\frac{1}{57}-5\cdot\arctan\frac{1}{239}+12\cdot\arctan\frac{1}{110443};$$
  - *F.C.W. Störmer* (1986)
$$\frac{\pi}{4}=44\cdot\arctan\frac{1}{57}+7\cdot\arctan\frac{1}{239}-12\cdot\arctan\frac{1}{682}+24\cdot\arctan\frac{1}{12943}.$$

  Для обчислення значення функції arctan із великою кількістю значущих цифр у випадку малих значень аргументу можна скористатись розкладанням у степеневий ряд:
$$\arctan x=x-\frac{x^3}{3}+\frac{x^5}{5}-\frac{x^7}{7}+\ldots=\sum_{n=0}^{\infty}\frac{(-1)^n x^{2n+1}}{2n+1}$$
- обчислення із заданою великою точністю (більше ніж 16 значущих десяткових цифр) значення константи e — основи натурального логарифму згідно із формулою:
$$\sum_{n=0}^{\infty}\frac{x^n}{n!}$$

Рисунок 1. – Завдання для роботи

Результати виконання завдання №1 наведені**:**

1.Лістинг 1-6. Вихідний код програми;

2.Рисунок 1-4 – Результат виконання програми.

Лістинг 1. Вихідний код програми

```
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.math.BigDecimal;

public class Client {
    public static void main(String[] args) {
        try {
```

```
            Registry registry = LocateRegistry.getRegistry(null);

            PiCalculator piStub = (PiCalculator) registry.lookup("PiCalculator");
            double piResponse = piStub.calculatePi(1000000);
            System.out.println("Value of Pi: " + piResponse);

            ECalculator eStub = (ECalculator) registry.lookup("ECalculator");
            BigDecimal eResponse = eStub.calculateE(20);
            System.out.println("Value of E: " + eResponse);

        } catch (Exception e) {
            System.err.println("Client exception: " + e.toString());
            e.printStackTrace();
        }
    }
}
```

Лістинг 2. Вихідний код програми

```
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.math.BigDecimal;

public interface ECalculator extends Remote {
    BigDecimal calculateE(int digits) throws RemoteException;
}
```

Лістинг 3. Вихідний код програми

```
import java.rmi.server.UnicastRemoteObject;
import java.rmi.RemoteException;
import java.math.BigDecimal;

public class ECalculatorImpl extends UnicastRemoteObject implements ECalculator {

    private static final BigDecimal ONE = BigDecimal.ONE;

    protected ECalculatorImpl() throws RemoteException {
        super();
    }

    @Override
    public BigDecimal calculateE(int digits) {
        return computeE(digits);
    }

    private BigDecimal computeE(int digits) {
        int scale = digits + 5;
        BigDecimal result = ONE;
```

```
        BigDecimal term = ONE;
        int i = 1;

        while (term.compareTo(BigDecimal.ZERO) != 0) {
            BigDecimal factorial = factorial(i);
            term = ONE.divide(factorial, scale, BigDecimal.ROUND_HALF_UP);
            result = result.add(term);
            i++;
        }

        return result.setScale(digits, BigDecimal.ROUND_DOWN);
    }

    private BigDecimal factorial(int n) {
        BigDecimal result = BigDecimal.ONE;
        for (int i = 2; i <= n; i++) {
            result = result.multiply(BigDecimal.valueOf(i));
        }
        return result;
    }
}
```

Лістинг 4. Вихідний код програми

```
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface PiCalculator extends Remote {
    double calculatePi(int terms) throws RemoteException;
}
```

Лістинг 5. Вихідний код програми

```
import java.rmi.server.UnicastRemoteObject;
import java.rmi.RemoteException;

public class PiCalculatorImpl extends UnicastRemoteObject implements PiCalculator
{
    protected PiCalculatorImpl() throws RemoteException {
        super();
    }

    @Override
    public double calculatePi(int terms) throws RemoteException {
        double result = 0;
        double sign = 1;
        for (int i = 0; i < terms; i++) {
            result += sign / (2 * i + 1);
            sign = -sign;
```

```
        }
        return result * 4;
    }
}
```

Лістинг 6. Вихідний код програми

```
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Server {
    public static void main(String[] args) {
        try {
            PiCalculator piCalculator = new PiCalculatorImpl();
            ECalculator eCalculator = new ECalculatorImpl();
            Registry registry = LocateRegistry.createRegistry(1099);
            registry.bind("PiCalculator", piCalculator);
            registry.bind("ECalculator", eCalculator);
            System.out.println("Server is running");
        } catch (Exception e) {
            System.err.println("Server exception: " + e.toString());
            e.printStackTrace();
        }
    }
}
```

```
PS D:\Lab7\L7>  & 'C:\Program Files\Java\jre1.8.0_51\bin\java
5e106fe58fa\redhat.java\jdt_ws\L7_ecb8d497\bin' 'Server'
Server is running
```

Рисунок 2. – Результат виконання програми

- *Machine – like formula* (1961)

$$\frac{\pi}{4} = 4 \cdot \arctan\frac{1}{5} - \arctan\frac{1}{239};$$

Рисунок 3. – Формула обчислення

```
PS D:\Lab7\L7>  & 'C:\Program Files\Java\jre1.8.0_51\bi
5e106fe58fa\redhat.java\jdt_ws\L7_ecb8d497\bin' 'Client'
Value of Pi: 3.1415916535897743
Value of E: 2.71828182845904523536
```

Рисунок 4. – Результат виконання програми

GitHub: https://github.com/zellii1/KPP