

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

Лабораторная работа №2
по курсу: «Языки и методы программирования»
«Разработка простейшего класса на языке Java»

Выполнил:
Студент группы ИУ9-21Б

Проверил:
Посевин Д. П.

Москва, 2024

1. Цель

Целью данной работы является изучение базовых возможностей языка Java.

2. Персональный вариант

Класс бинарных отношений на множестве целых чисел от 0 до n с двумя операциями: проверка принадлежности пары чисел отношению; вычисление транзитивного замыкания отношения.

3. Решение

3.1. Код

```
import java.util.Random;

public class Test {
    public static void main(String[] args) {
        Random rand = new Random();
        int n = 5 ; //+ rand.nextInt(10);
        BinaryRelation bin = new BinaryRelation(n);
        System.out.printf("Current size: %d\n", n+1);

        for (int i = 0; i < n; i++) {
            int x = rand.nextInt(n+1), y = rand.nextInt(n+1);
            System.out.printf("Adding new pair: (%d, %d)\n", x,
                → y);
            bin.addPair(x, y);
        }

        System.out.println("Relations matrix:");
        bin.printRelations();

        bin.computeTransitiveClosure();
        System.out.println("Transitive closure:");
        System.out.println(bin.toString());

        for (int i = 0; i < n / 2; i++) {
            int x = rand.nextInt(n+1), y = rand.nextInt(n+1);
            System.out.printf("Pair (%d, %d) in relation? %s\n",
                → x, y, bin.isPairInRelations(x, y) ? "yes" :
                → "no");
        }
    }
}
```

Код 1: Test.java

```
public class BinaryRelation {  
    private int size;  
    private int[][] relations;  
  
    public BinaryRelation(int n) {  
        this.size = n;  
        this.relations = new int[n+1] [n+1];  
    }  
  
    public void addPair(int x, int y) {  
        if (Math.min(x, y) < 0 || Math.max(x, y) > this.size) {  
            System.out.printf("Can't add pair, invalid values  
                  → (%d, %d).\n", x, y);  
        } else {  
            relations[x] [y] = 1;  
        }  
    }  
  
    public boolean isPairInRelations(int x, int y) {  
        if (Math.min(x, y) < 0 || Math.max(x, y) > this.size) {  
            System.out.printf("Invalid pair (%d, %d).\n", x, y);  
            return false;  
        } else {  
            return relations[x] [y] == 1;  
        }  
    }  
  
    //  
    ← https://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Флойда  
    public void computeTransitiveClosure() {  
        for (int k = 0; k <= this.size; k++) {  
            for (int i = 0; i <= this.size; i++) {  
                for (int j = 0; j <= this.size; j++) {  
                    this.relations[i] [j] |=  
                        (this.relations[i] [k] &  
                        this.relations[k] [j]);  
                }  
            }  
        }  
    }  
  
    public String toString() {
```

```

        String result = "";
        for (int i = 0; i <= this.size; i++) {
            for (int j = 0; j <= this.size; j++) {
                result += (relations[i][j] == 0) ? "0" : "1";
                result += (j != this.size) ? " " : "";
            }
            result += (i != this.size) ? "\n" : "";
        }
        return result;
    }

    public void printRelations() {
        System.out.println(this.toString());
    }
}

```

Код 2: BinaryRelation.java

3.2. Скриншоты

```

[thecakeisfalse@desktop-sc src]$ java Test
Current size: 6
Adding new pair: (2, 2)
Adding new pair: (0, 4)
Adding new pair: (2, 4)
Adding new pair: (5, 3)
Adding new pair: (3, 1)
Relations matrix:
0 0 0 0 1 0
0 0 0 0 0 0
0 0 1 0 1 0
0 1 0 0 0 0
0 0 0 0 0 0
0 0 0 1 0 0
Transitive closure:
0 0 0 0 1 0
0 0 0 0 0 0
0 0 1 0 1 0
0 1 0 0 0 0
0 0 0 0 0 0
0 1 0 1 0 0
Pair (0, 3) in relation? no
Pair (0, 2) in relation? no
[thecakeisfalse@desktop-sc src]$

```

Рис. 1: Пример работы

The image shows two side-by-side Java code editors within a development environment. Both editors have a dark theme and display code in white text.

Editor 1 (Top):

```

1 import java.util.Random;
2
3 public class Test {
4     public static void main(String[] args) {
5         Random rand = new Random();
6         int n = 5 + rand.nextInt(10);
7         BinaryRelation bin = new BinaryRelation(n);
8         System.out.printf("Current size: %d\n", n+1);
9
10        for (int i = 0; i < n; i++) {
11            int x = rand.nextInt(n+1), y = rand.nextInt(n+1);
12            System.out.printf("Adding new pair: (%d, %d)\n", x, y);
13            bin.addPair(x, y);
14        }
15
16        System.out.println("Relations matrix:");
17        bin.printRelations();
18
19        bin.computeTransitiveClosure();
20        System.out.println("Transitive closure:");
21        System.out.println(bin.toString());
22
23        for (int i = 0; i < n / 2; i++) {
24            int x = rand.nextInt(n+1), y = rand.nextInt(n+1);
25            System.out.printf("Pair (%d, %d) in relation? %s\n", x, y, bin.isPairInRelations(x, y) ? "yes" : "no");
26        }
27    }
}

```

Editor 2 (Bottom):

```

1 usages
public class BinaryRelation {
2
3     private int size;
4
5     private int[][] relations;
6
7     public BinaryRelation(int n) {
8         this.size = n;
9         this.relations = new int[n+1][n+1];
10    }
11
12    public void addPair(int x, int y) {
13        if (Math.min(x, y) < 0 || Math.max(x, y) > this.size) {
14            System.out.printf("Can't add pair, invalid values (%d, %d).\n", x, y);
15        } else {
16            relations[x][y] = 1;
17        }
18    }
19
20    public boolean isPairInRelations(int x, int y) {
21        if (Math.min(x, y) < 0 || Math.max(x, y) > this.size) {
22            System.out.printf("Invalid pair (%d, %d).\n", x, y);
23            return false;
24        } else {
25            return relations[x][y] == 1;
26        }
27    }
28
29 // https://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Флойда
30
31    public void computeTransitiveClosure() {
32        for (int k = 0; k <= this.size; k++) {
33            for (int i = 0; i <= this.size; i++) {
34                for (int j = 0; j <= this.size; j++) {
35                    this.relations[i][j] |= (this.relations[i][k] & this.relations[k][j]);
36                }
37            }
38    }
39
40    public String toString() {
41        String result = "";
42        for (int i = 0; i <= this.size; i++) {
43            for (int j = 0; j <= this.size; j++) {
44                result += (relations[i][j] == 0) ? "0" : "1";
45            }
46            result += (i != this.size) ? "\n" : "";
47        }
48        return result;
49    }
50
51    public void printRelations() { System.out.println(this.toString()); }
}

```

Рис. 2: Исходный код