Практическая работа №13

Вариант №1 – Основные алгоритмы работы с графами.

Автор: Николаев-Аксенов И. С.

Группа: ИКБО-20-19

Задание:

Составить программу реализации алгоритма Крускала построения остовного дерева минимального веса.

Выбрать и реализовать способ представления графа в памяти.

Предусмотреть ввод с клавиатуры произвольного графа.

Разработать доступный способ (форму) вывода результирующего дерева на экран монитора.

Провести тестовый прогон программы для заданного графа в соответствии с индивидуальным заданием

Код программы:

Graph.java:

```
1.  public class Graph implements Comparable<Graph> {
2.      public int A;
3.      public int mass;
4.      public int B;
5.
6.      Graph(int A, int mass, int B) {
7.          this.A=A;
8.          this.B=B;
9.          this.mass=mass;
10.     }
11.
12.     @Override
13.     public int compareTo(Graph o) {
14.         if(mass!=o.mass){
15.             return mass<o.mass? -1:1;
16.         }
17.         return 0;
18.     }
19. }
```

SetGraph.java:

```
1.  public class SetGraph {
2.      int[] number, rang;
3.
4.      SetGraph(int size) {
5.          number=new int[size];
6.          rang=new int[size];
7.          for(int i=0;i<size;++i){
8.              number[i]=i;
9.          }
10.     }
11.
```

```
    int set(int x) {
        return x==number[x]? x:(number[x]=set(number[x]));
    }

    boolean merge(int A, int B){
        if(set(A)==set(B))
            return false;

        if(rang[A]<rang[B])
            number[A]=B;

        else {
            number[B]=A;
            if(rang[A]==rang[B])
                rang[A]++;
        }
        return true;
    }
}
```

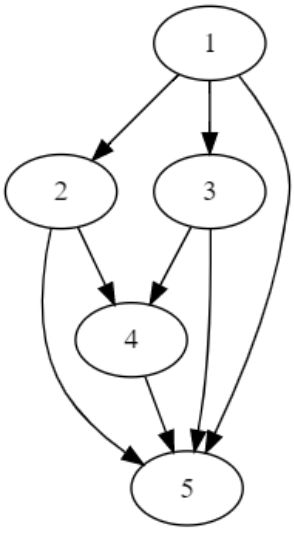
StartKruskalsAlgorithm.java:

```
import java.util.*;

public class StartKruskalsAlgorithm {
    public static int KruskalAlgorithm(ArrayList<Graph> graph) {
        SetGraph union = new SetGraph(graph.size()+1);
        Collections.sort(graph);

        ArrayList<Graph> buff = new ArrayList<>();
        for(Graph i: graph) {
            if(union.merge(i.A,i.B)){
                buff.add(i);
            }
        }
        graph.clear();
        graph.addAll(buff);
        return 0;
    }

    public static void main(String[] args) {
        ArrayList<Graph> graph = new ArrayList<>();
        Scanner scanner = new Scanner(System.in);
        int a,b,m;

        System.out.println("Ввод ребер графа: ");
        while(true){
            a = scanner.nextInt();
            if(a==0) break;
            m = scanner.nextInt();
            b = scanner.nextInt();
            graph.add(new Graph(a,m,b));
        }
        System.out.println("Исходный граф: ");
        for (Graph item : graph) {
            System.out.println(item.A + "->" + item.B);
        }
        System.out.println(KruskalAlgorithm(graph));
        System.out.println("Результат: ");
        for (Graph value : graph) {
            System.out.println(value.A + "->" + value.B);
        }
        System.out.println("В данной работе используется язык graphviz. Посмотреть граф
можно на сайте graphvix.org.");
    }
}
```

Результат выполнения программы:

```
Ввод ребер графа:
1 1 2
1 10 5
1 2 3
2 6 5
3 7 5
2 3 4
4 11 5
3 4 4
0
Исходный граф:
1->2
1->5
1->3
2->5
3->5
2->4
4->5
3->4
0
Результат:
1->2
1->3
2->4
2->5
```

Исходный граф:



Результат: