Programming I

Graphs

9th lecture

Overview

- Definition
- Formal definition
- What is a graph?
- Why graph?
- Graph as a data structure in Java
- Graph classes
- Implementation

Definition

- A set of elements that are associated with connections.
- Each element is called a node.
- Formally, graph is a set of nodes with binary adjacency relation.

Formal definition

- Graph G is defined as a pair (V,E), where V is a set of vertex (vozlišča), E is a set of edges (robovi).

$$E \subseteq \{(u,v) \mid u,v \in V\}$$

  - If the graph is undirected, then the adjacency relation is symmetric (adjacency applies to both nodes).

What is a graph?

- Graph theory is a branch of mathematics which had its beginnings in the second half of the 19th century.
- It developed from the need to solve real problems in science and technology.
- Its rapid development in recent decades is linked to an increasingly wider use in various scientific fields: architecture, biology, civil engineering, chemistry, computer science, sociology, ...

What is a graph?

- Graph is a structure in discrete mathematics, which can represent networks (roads, railways, sites, chemical molecules, maps, ...).
- Graph consists of points or nodes (places, stations, computers, atoms, ...), which are connected by lines, called links, edges, connections (road, track, wire ties, ...).

Graph as a data structure in Java

- You will see a "primitive" implementation of the abstract data structure graph.
- A graph represented like this is practically useless, as the search for related nodes takes too much time O(n).
- At the end of the exercise we will look at the implementation, which is much more useful: JGraphT.

Objects of the graph

- Source code:
  - **`Vozel.java`**
  - **`Povezava.java`**
  - **`Graf.java`**
  - **`StartGraf.java`**

Graph

```
public class Graf {
  private HashMap vozli = new HashMap();
  private Vector povezave = new Vector();
  //methods
}
```

HashMap

- Data structure,
- Quickly search for any item,
- Used to store and (quickly) search for nodes
- Methods:

```
put(key, element)  //add element
element get(key)//return element
```

Node

```
public class Vozel {
  String ime = "";
  public Vozel(String ii){
    ime = ii;
  }
}
```

Edge – connection (Povezava)

```
public class Povezava {
  private Vozel u, v;
  public Povezava(Vozel uu, Vozel vv){
    u = uu;
    v = vv;
  }
  public Vozel zacetek(){
    return(u);
  }
  public Vozel konec(){
    return(v);
  }
}
```

dodajVozel – add node(vertex)

```
public boolean dodajVozel(String i){
  if(vozli.get(i) == null){
    vozli.put(i, newVozel(i));
    return(true);
  }
  else{
    return(false);
  }
}
```

dodajPovezavo – add edge (connection)

```
public boolean dodajPovezavo(String u, String v){
  Vozel tmp1, tmp2;
  tmp1 = (Vozel)vozli.get(u);
  tmp2 = (Vozel)vozli.get(v);
  if((tmp1 != null) && (tmp2 != null)){
    povezave.addElement(new Povezava(tmp1, tmp2));
    return(true);
  }
  else{
    return(false);
  }
}
```

Print

```
public void izpisi(){
  Vozel tmp1, tmp2;
  Povezava pTmp;
  for(int i = 0; i < povezave.size(); i++ ){
    pTmp = (Povezava)povezave.elementAt(i);
    tmp1 = pTmp.zacetek();
    tmp2 = pTmp.konec();
    System.out.println(tmp1.ime + "--" + tmp2.ime);
  }
}
```

Usage

```
public class StartGraf {
  public static void main(String[] args) {
    Graf gg = new Graf();
    gg.dodajVozel("Lucija");
    gg.dodajVozel("Piran");
    gg.dodajVozel("Izola");
    gg.dodajVozel("Koper");
    gg.dodajVozel("Dekani");
    gg.dodajPovezavo("Lucija", "Piran");
    gg.dodajPovezavo("Piran", "Izola");
    gg.izpisi();
    gg.dodajPovezavo("Izola", "Koper");
    gg.dodajPovezavo("Koper", "Dekani");
    gg.izpisi();
  }
}
```

JGraphT

- A graph library
- Address:
  - http://jgrapht.sourceforge.net/
- Demo:
  - http://jgrapht.sourceforge.net/visualizations.html

JGraphT

- Let's make an example!
  - Prepared source code **`HelloJGraphT.java`**
- Compile:
  - **`javac -classpath jgrapht-jdk1.5.jar HelloJGraphT.java`**
- Start:
  - **`java -classpath jgrapht-jdk1.5.jar;. HelloJGraphT`**

Conclusion

- We have just learned a few things about:
  - Graphs