



Programiranje I – RIN Računalništvo I – MA

Grafi

Vsebina

- Definicija
- Formalna definicija
- Kaj je graf?
- Zakaj graf?
- Graf kot podatkovna struktura v Javi
- Razredi grafa
- Implementacija

Definicija

- Množica elementov, ki so povezani s povezavami.
- Vsak element imenujemo vozlišče.
- Formalno je graf množica vozlišč z binarno relacijo sosednosti.

Formalna definicija

- Graf G je definiran kot par (V, E) , kjer V predstavlja množico vozlišč (Vertex), E pa množico robov med vozlišči (Edges).

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

- Če je graf neusmerjen, potem je relacija sosednosti simetrična (sosednost velja za obe vozlišči).

Kaj je graf?

- Teorija grafov je veja matematike, ki ima svoje začetke v drugi polovici 19. stoletja.
- Razvila se je iz potrebe po reševanju konkretnih problemov v znanosti in tehnologiji.
- Njen nagel razvoj v zadnjih desetletjih je povezan z vedno širšo uporabo na različnih znanstvenih področjih: arhitektura, biologija, gradbeništvo, kemija, računalništvo, sociologija,...

Kaj je graf?

- Graf je struktura v diskretni matematiki, s katero lahko ponazorimo omrežje (cest, železnic, spletne strani, kemijske molekule, zemljevide,...).
- Graf je sestavljen iz točk ali vozlišč (kraji, postaje, računalniki, atomi,...), ki so povezane s črtami, ki jih imenujemo povezave (ceste, proge, žice, vezi,...)

Graf kot podatkovna struktura v Javi

- Ogleдали si bomo “primitivno” implementacijo abstraktne podatkovne strukture graf
- Tako predstavljen graf je praktično neuporaben, saj nam iskanje povezanih vozlišč vzame preveč časa $O(n)$.
- Na koncu vaj si bomo ogleдали implementacijo, ki je veliko bolj uporabna: JGraphT

Predmeti grafa

- Oglejmo si kodo:
 - `Vozel.java`
 - `Povezava.java`
 - `Graf.java`
 - `StartGraf.java`

Graf

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

HashMap

- Podatkovna struktura,
- Hitro iskanje poljubnega elementa,
- Potrebujemo pri iskanju vozlišč
- Metodi:
 - `put(key, element)` //dodaj element
 - `element get(key)`//vrne element, ki ga opisuje ključ

Vozel

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

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

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

dodajPovezavo

```
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);
    }
}
```

izpisi

```
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);
    }
}
```

Uporaba

```
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();  
    }  
}
```


Zaključek

- Ogledali smo si:
 - Graf