Master ICONE 1ère année

TP Graphes

Réaliser par :
Mohammed BENAOU
Mohammed RASFA

1 Implémentation d'un graphe

1.1 Prise en main de graphviz

Exercice 1

1-visualisation du graphe à partir le fichier (test.dot) :

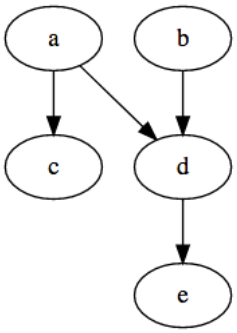

Figure 1 – Graphe

Exercice 2

- l'ensemble des états est Q = {a, BENAOU, c, d, e},
- l'alphabet d'entrée est = {0, 1,RASFA},

- l'état initial est I = a,
- le seul état final F = e,

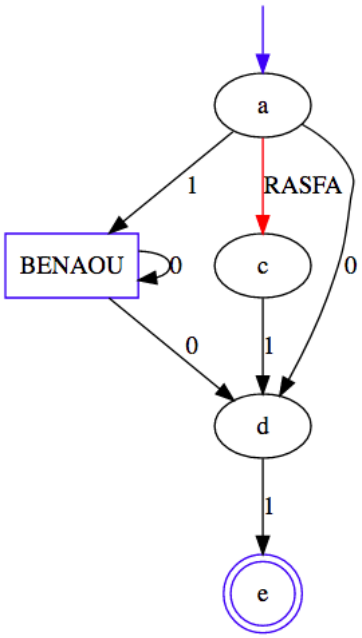

FIGURE 2 – Graphe orienté

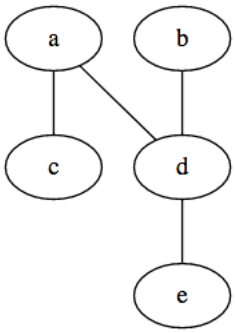

FIGURE 3 – Graphe non orienté

1.2 Classes DGraph et DAGraph en Java

Exercice 3

1. les variables d'instance d'un noeud sont : un identifiant,un contenu.

Le code source

```
1  public static void main(String[] args) {
2        // TODO code application logic here
3
4
5         LinkedList<Node> N = new LinkedList<Node> ();
6
7        Node n0=new Node("a");
8        Node n1=new Node("b");
9        Node n2=new Node("c");
10       Node n3=new Node("x");
11
12       N.add(new Node("a"));
13       N.add(n1);
14       N.add(n2);
15       TreeSet<Node> treeS = new TreeSet<Node> ();
16       treeS.add(n3);
17       treeS.add(n2);
18       treeS.add(n1);
19       treeS.add(n0);
20       System.out.println(N);
21       System.out.println(treeS);
22
23    }
```

Exercice 4

2– Les noeuds d'un graphe sont stockés dans le container Treeset.
Les arcs d'un graphe sont stockés dans des Treemap de Treeset.
3– Les variables d'instances d'un graphe :
TreeSet<N> nodes, TreeMap<N, TreeSet<Edge<N,U»> pred, TreeMap<N,TreeSet<Edge<N,U»> succ ;
Les variables d'instance d'un arc sont : • from, to, content
4– Le mécanisme de comparaison entre les arcs mis en oeuvre est la méthode compareTo pour comparer les identificateurs des arcs.
5– voir le code

Exercice 5

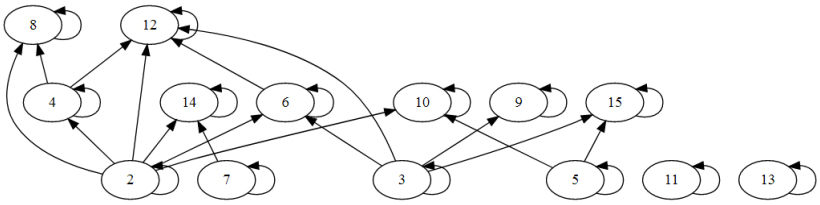

FIGURE 4 – Graphe

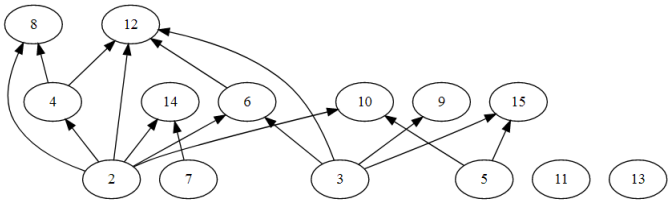

FIGURE 5 – Graphe

Liste des sources	2,7,3,5,11,13,4,14,9,6,15,10,8,12
Tri-topologique	2,7,3,5,11,13,4,14,9,6,15,10,8,12

TABLE 1 – table de tri topologique

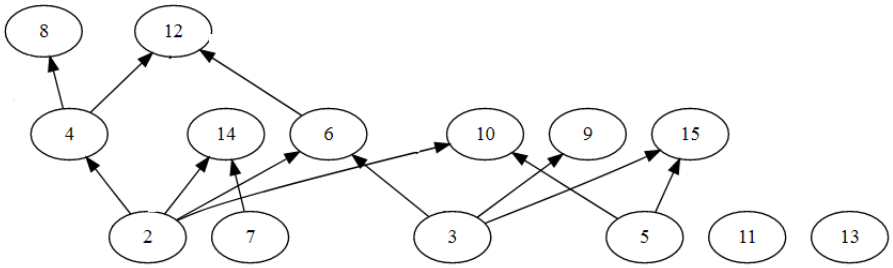

FIGURE 6 – La résuction transitive

Exercice 6

on a ajouté l'attribut couleur dans la fonction toDot voila le code modifié ci-dessus :

```
1  public void toDot (String filename) {
2      try {
3      String color = "red";
4      FileOutputStream fich = new FileOutputStream(filename);
5      DataOutputStream out = new DataOutputStream(fich);
6      out.writeBytes("digraph G {\n");
7          out.writeBytes("Graph [rankdir=BT]\n");
8      StringBuffer nodes = new StringBuffer();
9      String edges = "";
10     for (Node from : this.getNodes())
```

```
      nodes.append(from.toDot()+ "\n");
        for (Edge ed : this.getEdges())
            edges+=ed.toDot().append("[color="+color+"]\n");
    out.writeBytes(nodes+edges);
    out.writeBytes("}");
    out.close();
     }
    catch (Exception e) { e.printStackTrace(); }
  }
```

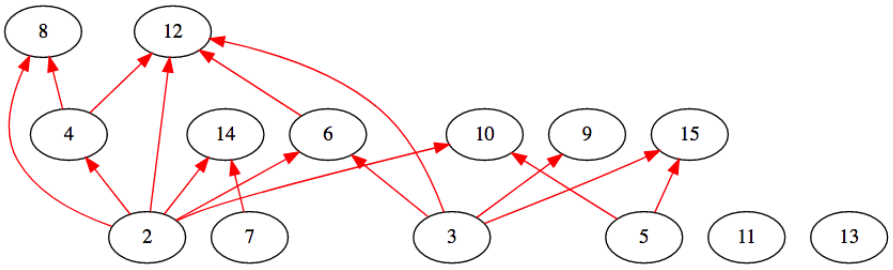

FIGURE 7 – Graphe java avec la couleur

2 Classe DiGraph en Python

Exercice 7

Le code source Voila la classe main avec laquelle on a pu exécuter et après visualiser le graphe :

```
import DiGraph

G=DiGraph.DiGraph()
G.add_nodes_from(range(1,10))

for i in range(1,10):
   G.add_edge(i,i+1)
   G.add_edge(i,i+2)

print G


file = open('G.dot','w')
file.write(G.to_dot())
file.close()

print G.sinks()
print G.topological_sort()

H=G.div_graph()

```

```
22 print H
```

Exercice 8

1−. voici le graphe du fichier G.dot,

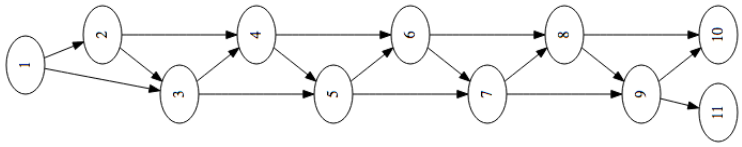

FIGURE 8 – Graphe Python

Liste des sources	1,2,3,4,5,6,7,8,9,10,11
Tri-topologique	1,2,3,4,5,6,7,8,9,10,11

TABLE 2 – table de tri topologique

Le tableau est à priopri trié [1]

1. http ://www.webgraphviz.com/