TP 6 : Encore des Graphes

23 mai 2013

Question 1
Télécharger les données depuis le site `http://snap.stanford.edu/data/as.html`. Extraire le fichier `.txt` de l'archive.

Question 2
Transformer en liste d'adjacence. (Retirer les commentaires. interpréter les lignes comme des paires d'entiers)

Question 3
Récupérer les sommets. Récupérer les arêtes.

Question 4
Trouver le sommet de plus grand degré.

Question 5
Utiliser l'algorithme de Floyd-Warshall pour déterminer toutes les distances.

Question 6
Utiliser l'algorithme de Prim pour construire un arbre couvrant.

```
let dist be a |V| x |V| array of minimum distances initialized to infinity
for each vertex v: dist[v][v] <- 0
for each edge (u,v): dist[u][v] <- w(u,v)  // the weight of the edge (u,v)
for k from 1 to |V|
   for i from 1 to |V|
      for j from 1 to |V|
         if dist[i][k] + dist[k][j] < dist[i][j] then
            dist[i][j] <- dist[i][k] + dist[k][j]
```

```
Create a tree containing a single vertex, chosen arbitrarily from the graph
Create a set containing all the edges in the graph
Loop until every edge in the set connects two vertices in the tree
    Remove from the set an edge with minimum weight that connects a vertex in the tree with a vertex not in the
    Add that edge to the tree
```

```
def readfile(fname):
    with open(fname) as f: content = f.readlines()
    content = content[0:1000]
    notComment = [line for line in content if not line.startswith('#')]
    pairs = [line.split() for line in notComment]
    return pairs

def getVertex(edges):
    vertices = set()
    for (i, j) in edges:
        vertices.add(i); vertices.add(j)
    return vertices

def getEdges(edges):
    e = set()
    for (i, j) in edges:
        e.add((i,j)); e.add((j,i))
    return e
```