Université des Sciences et de la Technologie d'Oran – Mohamed Boudiaf
Faculté de Génie Électrique – Département d'Électronique
Niveau: M2 – ESE / **Module:** Langage Java
E-mail: meriem.aarizou@univ-usto.dz
Dépôt Git: https://github.com/MerAARIZOU/Java_M2_ESE

TP n° 2: Classe et Objet

Modélisation des graphes en classe

Objectif

Utiliser le formalisme orienté objet pour modéliser un graphe orienté et valué. Reprendre les classes définies dans TP1 qui utilisent une matrice d'adjacence pour représenter un graphe orienté et valué, et reformuler les méthodes `succ`, `pred` et `dijkstra` pour une modélisation orientée objet.

Données

- Classe `tp2.graphe.Sommet` qui représente un sommet/nœud d'un graphe.
- Classe `tp2.graphe.Arete` qui représente l'arête/liaison valuée entre les différents sommets d'un graphe.
- Classe `tp2.graphe.Graphe` qui permet de modéliser un graphe valué et orienté. Celui-ci est constitué d'une liste de sommets et d'arêtes
- La classe `Ex1` qui contient un exemple d'utilisation de la classe `java.util.Scanner` pour lire un fichier texte et afficher son contenu ligne par ligne.

Énoncé

1. Créer un projet qui contient les classes `tp2.graphe.Sommet`, `tp2.graphe.Arete` et `tp2.graphe.Graphe`.
2. Dans le projet, définir la classe `Application` qui contient la méthode principale `main`
3. En utilisant la classe `Ex1`, écrire un programme dans la méthode principale qui permet de construire un graphe à partir du contenu du fichier `"graphe.txt"`.
4. Le fichier `"graphe.txt"` contient la description d'un graphe valué et orienté sous forme de matrice d'adjacence. Le contenu du fichier correspond au modèle suivant :
   (a) Le fichier est divisé en deux parties : les sommets et la matrice d'adjacence
   (b) La première ligne correspond aux sommets, elle définit les noms ordonnés, des sommets du graphe
   (c) les autres lignes correspondent aux valeurs de la matrice d'adjacence du graphe.
   (d) Chaque ligne et chaque colonne représentent un sommet.
   (e) Les indices des lignes et des colonnes correspondent à l'ordre défini dans la première ligne du fichier.
   (f) Les valeurs de la matrice représentent les valeurs des arêtes du graphe.
   (g) La valeur -1 indique qu'il n'existe pas de liaison entre sommet de la ligne et le sommet de la colonne.
   (h) Toutes les valeurs du fichier sont séparées par une `tabulation` ($\backslash t$)
   (i) Le graphe donné dans l'exercice 02 du TP1 est représenté par le fichier suivant :
   ```
   1  2  3  4  5
   -1  10  -1  5  -1
   -1  -1  1  2  -1
   -1  -1  -1  -1  4
   -1  3  9  -1  2
   7  -1  6  -1  -1
   ```
5. Chaque sommet du graphe doit être défini par un objet de la classe `tp2.graphe.Sommet`
6. Chaque arête du graphe doit être définie par un objet de la classe `tp2.graphe.Arete`, cet objet doit relier le sommet d'origine au sommet de destination.
7. Compléter la classe `tp2.graphe.Graphe` de manière à ce qu'un objet contient l'ensemble des sommets et arêtes précédemment définis.
8. Dans la classe `tp2.graphe.Sommet` définir les méthodes :
   — `succ` qui retourne la liste des successeurs d'un sommet
   — `pred` qui retourne la liste des prédécesseurs d'un sommet
   — `voisin` qui retourne la liste des voisins
9. Dans la classe `tp2.graphe.Sommet` définir la méthode `dijkstra` qui retourne la liste des plus courts chemins d'un sommet donné vers tous les autres sommets du graphe.

Compte rendu

Il est demandé de faire un compte rendu sur la modélisation des graphes orientés et valués en détaillant la différence entre les deux modes de programmation (procédurale et orienté objet). Le document doit être envoyé à l'adresse numérique au plus tard **08 Janvier 2020**. Le rapport doit répondre aux questions suivantes :

1. Expliquer le fonctionnement de la lecture des fichiers
2. Expliquer l'utilisation des blocs `try` et `catch` et la notion des exceptions. (Prendre l'exemple d'une lecture d'un fichier qui n'existe pas)
3. La `javaDoc` est un outil qui permet de créer de la documentation sous forme HTML depuis des commentaires dans du code Java. Cet outil a été utilisé pour décrire les deux classes `tp2.graphe.Sommet` et `tp2.graphe.Arete` ainsi que la classe `Ex1`. Utiliser l'exemple déjà fourni pour construire la documentation de la classe `tp2.graphe.Graphe` et compléter la documentation des deux classes `tp2.graphe.Sommet` et `tp2.graphe.Arete`
4. En utilisant la documentation officielle de JDK, générer par ligne de commandes les pages HTML de documentation des classes
5. Tracer le diagramme de classes de cet exercice
6. Quels étaient les avantages/inconvénients de la modélisation orientée objet par rapport à la programmation classique.