Théorie des graphes

TP N°3 (1ère Génie INFO)

Mode de représentation d'un graphe

Construction d'un graphe

A.U 2022-2023

On se propose de programmer un ensemble de fonctions de base (structures, types et fonctions) pour la gestion d'un graphe (orienté). Le langage de programmation utilisé est le langage C.

Afin de simplifier le travail, nous ne considérons que des graphes composés de sommets dont les noms sont représentés par des chiffres.

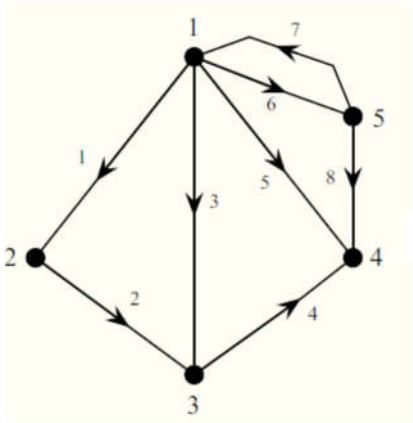

Figure 1

A. Les structures employées:

1) Définition des types de base pour la construction d'un graphe

Pour construire un graphe, on définit en C les éléments suivants :

a) Définition d'un arc

```
typedef struct arc
{
int  init; // sommet initial
int  final; // sommet final
float poids ; // cout de l'arc ou arête
}arc;
```


b) Définition d'un graphe

```
typedef struct graph
{ int nb_sommets; // nombre de sommets du graphe
  int nbarcs; // nombre d'arcs du graphe
int *liste_sommets; // liste des sommets du graphe
arc * arcs; // liste des arcs ou arêtes du graphe
}graph ;
```

c) Définition des constantes

Définir deux constantes **Nbarc**=10 (Nombre maximum d'arcs) et **Nbs** =6 (nombre maximum de sommets)

2) Création d'un fichier texte "graphe.txt"

Pour simplifier la saisie de données, on représente le graphe de la figure 1 (5 nœud et 8 arcs) par le fichier texte suivant:

```
5 8
1 2 1
1 3 3
1 4 5
1 5 6
2 3 2
3 4 4
5 4 8
5 1 7
```

B. Ecrire les fonctions suivantes :

1) Construction d'un graphe à partir d'éléments saisis

a) graph saisie_graphe () : permet de remplir un graphe

b) void affichegraphe(graphe g) : permet d'afficher pour chaque sommet la liste des sommets adjacents.

2) Construction d'un graphe à partir d'un fichier

a) graph creer_grapheV2(): permet de créer un graphe à partir d'un fichier texte "**graph.txt**".

b) void affichegraphe(graphe g) // appeler la fonction de 1)b)

3) Construction des matrices d'adjacence et d'incidence à partir du graphe g

a) void Rzero (int M[Nbs][Nbs] , int s) : permet de mettre à zéro une matrice carrée **M** d'ordre s.

b) void affichmat(int M[Nbs][Nbs], int s) : permet d'afficher une matrice carrée **M.**

c) **void Lsucc**(graphe g, int Msucc[Nbs][Nbs]) : permet de remplir la matrice de successeurs **Msucc**.

d) **void Lpred**(graphe g, int Mpred[Nbs][Nbs]): permet de remplir la matrice de prédécesseurs **Mpred**.

Exemple *: les matrices de successeurs et de prédécesseurs* du graphe de la figure 1 sont les suivantes :

Liste de successeurs

2	**3**	**4**	**5**	
3				
4				
1	**4**			

Liste de prédécesseurs

5				
1				
1	**2**			
1	**3**	**5**		
1				

e) **Remplir une matrice d'adjacence (avec deux méthodes)**

Considérons un graphe G = (X,A) comportant **s** sommets. La matrice d'adjacence de G est égale à la matrice U = (u_{ij}) de dimension s ×s telle que.

$$u_{ij} = \begin{cases} 1 & \text{si } (i,j) \in A \text{ (c'est-à-dire } (i,j) \text{ est une arête)} \\ 0 & \text{sinon} \end{cases}$$

Exemple : *La matrice d'adjacence du graphe de la* ***figure 1*** *est la suivante :*

$$U = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

- void **MatadjV1**(int Msucc[Nbs][Nbs],int Madj[Nbs][Nbs] , int s) : permet de remplir une matrice statique **Madj** à partir de la matrice des successeurs M**succ**.
- void **MatadjV2**(int Mpred[Nbs][Nbs], int Madj[Nbs][Nbs], int s) : permet de remplir une matrice statique **Madj** à partir de la matrice des prédécesseurs **Mpred**.

f) Remplir une matrice d'incidence.

Considérons un graphe orienté sans boucle G = (X,A) comportant **s** sommets et **a** arêtes. On appelle matrice d'incidence (aux arcs) de **G** la matrice **M** = (m_{ij}) de dimension s×a telle que :

$$m_{ij} = \begin{cases} 1 & \text{si } x_i \text{ est l'extrémité initiale de } a_j \\ -1 & \text{si } x_i \text{ est l'extrémité terminale de } a_j \\ 0 & \text{si } x_i \text{ n'est pas une extrémité de } a_j \end{cases}$$

La matrice d'incidence du graphe de la figure 1 s'écrit sous la forme suivante :

```
La matrcie d'incidence methode 1 est =
     1     1     1     1     0     0     0    -1
    -1     0     0     0     1     0     0     0
     0    -1     0     0    -1     1     0     0
     0     0    -1     0     0    -1    -1     0
     0     0     0    -1     0     0     1     1
```

- **void matincidence**(int Msucc[Nbs][Nbs], int **Mincid**[Nbs][Nbarc], int s) permet de remplir une matrice d'incidence **Mincid** à partir d'une matrice **Msucc**
- **void Affichmat2**(int M[Nbs][Nbarc], int s, int a) : permet d'afficher la matrice d'incidence.
- **void Rzero2** (int M[Nbs][Nbarc],int s,int a) : remise à zéro de la matrice d'incidence .

4) Construction de la matrice d'adjacence à partir d'un fichier

int MatadjV3(Madj[Nbs][Nbs]) : permet de remplir la matrice statique **Madj** à partir d'un fichier texte "**graph.txt**" et de retourner le nombre de sommets.