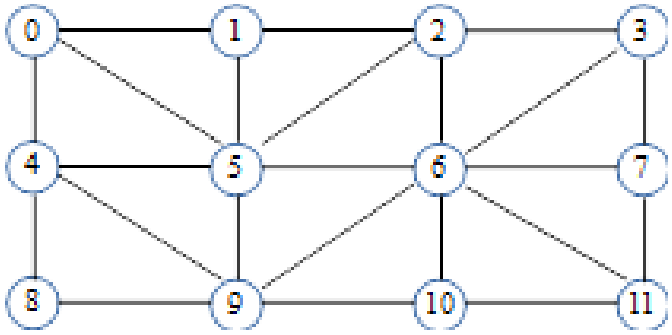
Université de Reims Champagne-Ardenne
Département de Mathématiques, Mécanique et Informatique

Info0501 – 2021-2022
Pierre Delisle

Travaux pratiques 2

Parcours de graphes

L'objectif de ce TP est de programmer les principaux algorithmes de parcours de graphes. Pour ce faire, nous utiliserons le graphe suivant comme exemple :



Ce graphe correspond au fichier **graphe2.txt** qui a été traité lors du TP 1.

Parcours en largeur

Afin de réaliser le parcours en largeur, il faut tout d'abord créer une structure permettant d'enregistrer le parcours.

1) Créez un tableau de sommets permettant, pour chaque sommet du graphe, de conserver sa couleur, sa distance du sommet d'origine du parcours et son père dans l'arbre de parcours.

Il faut ensuite créer une structure de file afin de gérer l'ensemble des nœuds durant le parcours.

2) Créez les fichiers **file.c** et **file.h** et implémentez-y une structure de file de sommets par tableau ainsi que les procédures de file suivantes :

- **initialiser_file** : initialise une file vide de capacité maximale fixe en allouant la mémoire nécessaire ;
- **detruire_file** : détruit une file en libérant ses ressources mémoire ;
- **file_vide** : retourne vrai si la file est vide, faux sinon ;
- **file_pleine** : retourne vrai si la file est pleine, faux sinon ;
- **enfiler** : ajoute un élément en queue de file ;
- **défiler** : enlève l'élément en tête de file s'il en existe un et retourne sa valeur.

Avant d'aller plus loin, n'oubliez pas d'écrire un main pour tester votre file !

3) Définissez la fonction **parcours_largeur** qui construit l'arbre de parcours en largeur à partir d'un sommet spécifique.

4) Définissez la fonction **afficher_chemin** qui, à partir d'une arborescence de parcours en largeur, donne le chemin le plus court entre l'origine du parcours et un sommet du graphe. Affichez ensuite le plus court chemin entre l'origine 0 et chaque sommet, et vérifiez que vous obtenez bien les chemins suivants :

```
0
0 1
0 5 2
0 5 6 3
0 4
0 5
0 5 6
0 5 6 7
0 4 8
0 5 9
0 5 9 10
0 5 6 11
```

Parcours en profondeur

5) Définissez la fonction **parcours_profondeur_recursif** qui construit une forêt de parcours en profondeur tout en calculant les dates de découverte et de fin de traitement de chaque sommet.

6) Définissez la procédure **afficher_parcours_profondeur** qui, à partir d'une arborescence de parcours en profondeur, affiche les dates de découverte et de fin de traitement de chaque sommet. Vérifiez que vous obtenez bien les résultats suivants :

```
Sommet : 0, date debut : 1, date fin : 24, pere : -1
Sommet : 1, date debut : 10, date fin : 11, pere : 2
Sommet : 2, date debut : 9, date fin : 12, pere : 3
Sommet : 3, date debut : 8, date fin : 13, pere : 6
Sommet : 4, date debut : 19, date fin : 20, pere : 8
Sommet : 5, date debut : 2, date fin : 23, pere : 0
Sommet : 6, date debut : 7, date fin : 14, pere : 7
Sommet : 7, date debut : 6, date fin : 15, pere : 11
Sommet : 8, date debut : 18, date fin : 21, pere : 9
Sommet : 9, date debut : 3, date fin : 22, pere : 5
Sommet : 10, date debut : 4, date fin : 17, pere : 9
Sommet : 11, date debut : 5, date fin : 16, pere : 10
```

Afin de réaliser un parcours en profondeur de façon itérative plutôt que récursive, on peut utiliser une pile.

7) Créez les fichiers **pile.c** et **pile.h** et implémentez-y une structure de pile de sommets par tableau ainsi que les procédures de pile suivantes :

- **initialiser_pile** : initialise une pile vide de capacité maximale fixe en allouant la mémoire nécessaire ;
- **detruire_pile** : détruit une pile en libérant ses ressources mémoire ;
- **pile_vide** : retourne vrai si la pile est vide, faux sinon ;
- **pile_pleine** : retourne vrai si la pile est pleine, faux sinon ;
- **sommet** : retourne la valeur de l'élément au sommet de la pile sans le dépiler ;
- **empiler** : ajoute un élément au sommet de la pile ;
- **dépiler** : enlève l'élément au sommet de la pile s'il en existe un et retourne sa valeur.

8) Définissez la fonction **parcours_profondeur_iteratif** qui réalise le parcours en profondeur de façon itérative en utilisant une pile.

9) Complétez votre main de sorte à afficher les résultats des parcours implémentés et vérifiez les résultats obtenus.