ABULIMITI Alafate

LI Yuanyuan

12/05/2017

Rapport Projet C++: Les Graphes

1. Préambule

Ce projet tuteuré nous allons travailler les graphes, L'objectif est de réaliser une libraire de classes et fonctions permettant de manipuler des graphes. La réalisation de ce projet va nécessiter l'utilisation de la méthodologie UML pour modéliser le logiciel a développer ,ainsi que le langage C++ pour la mise en oeuvre [1] , nous utilisons le VS 2012 pour C++.

Nous créons sept classes pour gérer des graphes: Cexception, Carc, Csommet, Cgraphe, CsommetOpération, CgrapheOpération, Cfichier. Cexception stocke les méthodes des exceptions, par exemple , erreur d'ouvre le fichier ... Carc ne stocke que un seul attribut : le numéro du sommet destination, et les méthodes relatives. Csommet stocke cinq attributs: le numéro du sommet, la liste des arcs partants du sommet et la liste des arcs arrivants au sommet, le numéro de Arrivant et le numéro de Partant, et les méthodes relatives. Cgraphe stocke deux attributs : le numéro de sommet et un tableau de Csommet et les méthodes relatives. CsommetOpération qui stocke les méthodes de opération de base hérite Csommet. CgrapheOpération qui stocke les méthodes de opération de base hérite Cgraphe. Cfichier stocke les méthodes qui présente un parseur et le méthode qui peut lire le fichier.

2. Structure UML

2.1 Le diagramme de classes(figure 2.1)

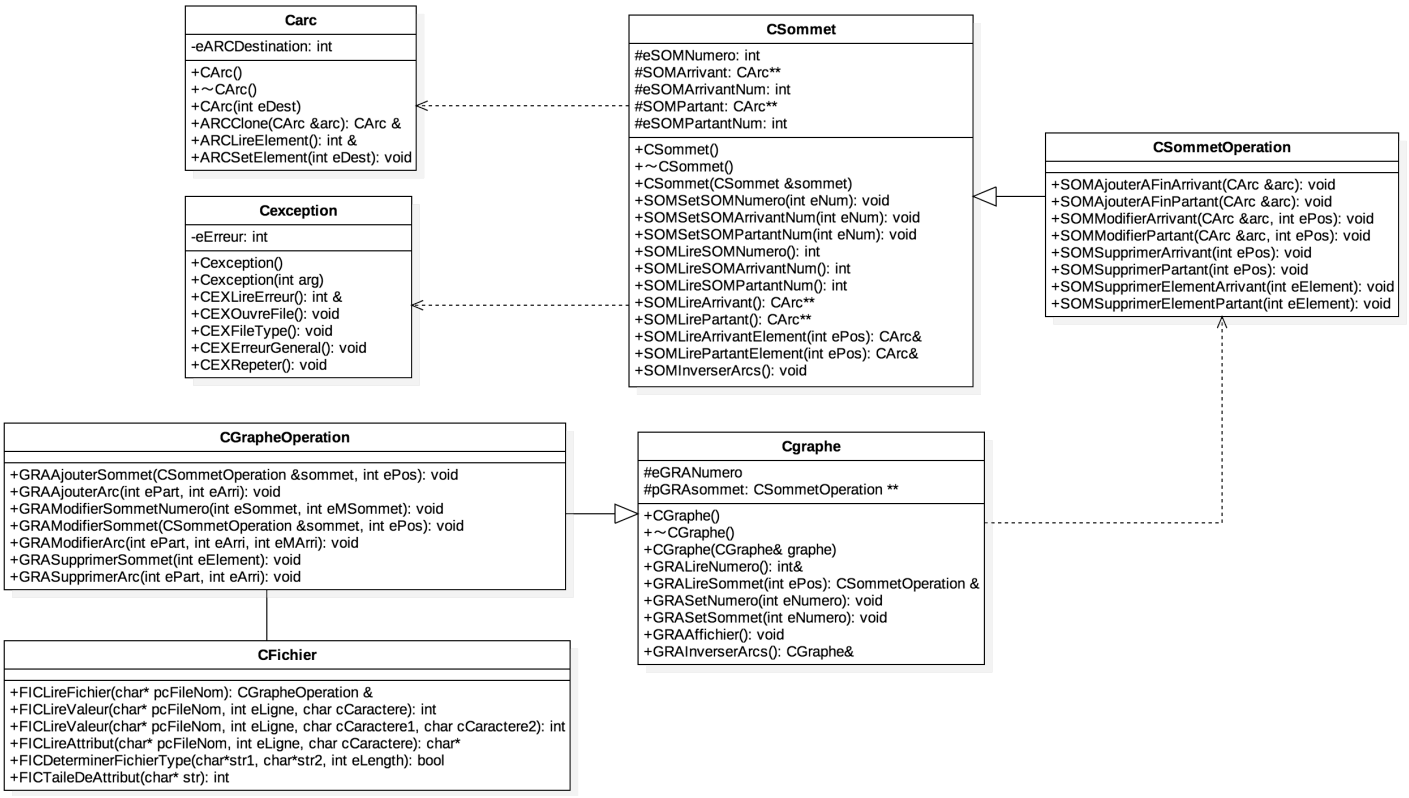

figure 2.1

3. L'Explication de diagramme

3.1 la classe <<CArc>>

Cette classe représente un arc.

3.1.1 attributs:

Dans cette classe, il n'y a qu'un seul attribut, la visibilité est <private>:
int eARCDestination: le numéro du sommet destination.

3.1.2 méthodes

Cette classe comporte cinq méthodes. La visibilité est <public>.

1) CArc ();
   -fonction: le constructeur.
2) CArc (int eDest);
   -fonction: le constructeur.
3) CArc& ARCClone(CArc &arc);
   - Les paramètres: un objet d'arc.
   - Fonction: cloner un arc.
   - Retour: un objet d'arc.
4) int &ARCLireElement();

- Fonction: lire l'attribut d'un objet.
- Retour: l'attribut eARCDestination.

5) void ARCSetElement(int eDest);
- Fonction: configurer l'attribut d'un objet.
- Retour: void.

3.2 la classe <<CSommet>>

Cette classe est pour sommet.

3.2.1 attributs

Dans cette classe, il y a cinq attributs. La visibilité est <protected>. Ils sont:

1) int eSOMNumero: le numéro d'un sommet.
2) CArc** SOMArrivant: la liste des arrivant du sommet d'arrive.
3) int eSOMArrivantNum: le nombre de la liste des arrivant du sommet d'arrive
4) CArc** SOMPartant: la liste des partant du sommet de départ.
5) int eSOMPartantNum: le nombre de la liste des partant du sommet de départ

3.2.2 méthodes

Cette classe comporte quatorze méthodes. La visibilité est <public>. Ils sont:

1) CSommet();
-fonction: le constructeur
2) ~CSommet();
-fonction: le destructeur.
3) CSommet(CSommet &sommet);
-fonction: le constructeur de recopie.
4) void SOMSetSOMNumero(int eNum);
- Les paramètres: un numero.
- Fonction: configurer l'attribut - eSOMNumero.
- Retour: void.
5) void SOMSetSOMArrivantNum(int eNum);
- Les paramètres: un numero.
- Fonction: configurer l'attribut - eSOMArrivantNum.
- Retour: void.
6) void SOMSetSOMPartantNum(int eNum);
- Les paramètres: un numero.
- Fonction: configurer l'attribut - eSOMPartantNum.
- Retour: void.
7) int SOMLireSOMNumero();
- Fonction: lire l'attribut - eSOMNumero.
- Retour: le valeur d'attribut - eSOMNumero.

8) int SOMLireSOMArrivantNum();
- Fonction: lire l'attribut - eSOMArrivantNum.
- Retour: le valeur d'attribut - eSOMArrivantNum.

9) int SOMLireSOMPartantNum();
- Fonction: lire l'attribut - eSOMPartantNum.
- Retour: le valeur d'attribut - eSOMPartantNum.

10) CArc** SOMLireArrivant();
- Fonction: lire l'attribut - SOMArrivant.
- Retour: le pointeur d'attribut - SOMArrivant.

11)CArc** SOMLirePartant();
- Fonction: lire l'attribut - SOMPartant.
- Retour: le pointeur d'attribut - SOMPartant.

12) CArc& SOMLireArrivantElement(int ePos);
- Les paramètres: la position qui vous voulez obtenir dans la liste des arrivant du sommet d'arrive.
- Fonction: lire la position spécifique dans la liste des arrivant du sommet d'arrive.
- Retour: la référence d'un arc.

13) CArc& SOMLirePartantElement(int ePos);
- Les paramètres: la position qui vous voulez obtenir dans la liste des partant du sommet de départ.
- Fonction: lire la position spécifique dans la liste des partant du sommet de départ.
- Retour: la référence d'un arc.

14) void SOMInverserArcs();
- Fonction: échanger la liste des partant du sommet de départ et la liste des arrivant du sommet d'arrive.
- Retour: void.

3.3 la classe <<CSommetOperation>>

Cette classe hérite la classe <<CSommet>>. Cette classe est pour les opérations de la classe <<CSommet>>

3.3.1 méthodes

Cette classe comporte huit méthodes. La visibilité est <public>. Ils sont:

1) void SOMAjouterAFinArrivant(CArc &arc);
- Les paramètres: la référence d'un arc que vous voulez ajouter a la fin de la liste des arrivant du sommet d'arrive.
- Fonction: ajouter un arc à la fin de la liste des arrivant du sommet d'arrive.
- Retour: void.

2) void SOMAjouterAFinPartant(CArc &arc);

- Les paramètres: la référence d'un arc que vous voulez ajouter a la fin de la liste des partant du sommet de départ.

- Fonction: ajouter un arc à la fin de la liste des partant du sommet de depart.

- Retour: void.

3) void SOMModifierArrivant(CArc &arc,int ePos);

- Les paramètres: la référence d'un arc; la position spécifique.

- Fonction: modifier un arc qui est dans la position spécifique de la liste des arrivant du sommet d'arrive.

- Retour: void.

4) void SOMModifierPartant(CArc &arc,int ePos);

- Les paramètres: la référence d'un arc; la position spécifique.

- Fonction: modifier un arc qui est dans la position spécifique de la liste des partant du sommet de départ.

- Retour: void.

5) void SOMSupprimerArrivant(int ePos);

- Les paramètres: la position spécifique.

- Fonction: supprimer un arc qui est dans la position spécifique de la liste des arrivant du sommet d'arrive.

- Retour: void.

6) void SOMSupprimerPartant(int ePos);

- Les paramètres: la position spécifique.

- Fonction: supprimer un arc qui est dans la position spécifique de la liste des partant du sommet de départ.

- Retour: void.

7) void SOMSupprimerElementArrivant(int eElement);

- Les paramètres: l'attribut - eARCDestination d'arc qui est dans la position spécifique de la liste des arrivant du sommet d'arrive..

- Fonction: supprimer un arc (dans la liste des arrivant du sommet d'arrive) dont l'attribut - eARCDestination est <<eElement>>.

- Retour: void.

8) void SOMSupprimerElementPartant(int eElement);

- Les paramètres: l'attribut - eARCDestination d'arc qui est dans la position spécifique de la liste des partant du sommet de départ.

- Fonction: supprimer un arc (dans la liste des partant du sommet de départ) dont l'attribut - eARCDestination est <<eElement>>.

- Retour: void.

-

3.4 la classe <<CGraphe>>

Cette classe est pour graphe.

3.4.1 attribut

Il y a deux attributs dans cette classe. La visibilité est <protected>. Ils sont:
1) int eGRANumero: le nombre des sommets qui sont dans un graphe.
2) CSommetOperation **pGRAsommet: le tableau de sommet (la classe CSommetOperation) et on peut faire les opérations sur ce tableau.

3.4.2 méthodes

Cette classe comporte neuf méthodes. La visibilité est <public>. Ils sont:
1) CGraphe();
-fonction: le constructeur
2) ~CGraphe();
-fonction: le constructeur de recopie.
3) CGraphe (CGraphe& graphe);
-fonction: le destructeur
4) int &GRALireNumero();
- Fonction: lire le nombre des sommets dans un graphe.
- Retour: le nombre des sommets dans un graphe.
5) CSommetOperation &GRALireSommet(int ePos);
- Les paramètres: la position spécifique dans le tableau des sommets.
- Fonction: lire le sommet qui est dans la position spécifique dans le tableau des sommets.
- Retour: le sommet qui est dans la position spécifique dans le tableau des sommets.
6) void GRASetNumero(int eNumero);
- Les paramètres: un numero.
- Fonction: modifier le nombre de sommets.
- Retour: void.
7) void GRASetsommet(int eNumero);
- Les paramètres: le nombre de sommets.
- Fonction: créer un tableau des sommet du graphe.
- Retour: void.
8) void GRAAffichier();
- Fonction: affichier un graphe.
- Retour: void.
9) CGraphe& GRAInverserArcs();
- Fonction: inverser tous les arcs du graphe.
- Retour: un nouveau graphe.

3.5 la classe CGrapheOperation

Cette classe hérite la classe <<CGraphe>>. Cette classe est pour faire opérations sur un graphe. Cette classe comporte sept méthodes. La visibilité est <public>. Ils sont:

1) void GRAAjouterSommet(CSommetOperation &sommet,int ePos);
- Les paramètres: la référence d'un sommet; la position spécifique.
- Fonction: ajouter un sommet dans un graphe..
- Retour: void.

2) void GRAAjouterArc(int ePart, int eArri);
- Les paramètres: le numéro de départ; le numéro d'arrive.
- Fonction: ajouter un arc dans un graphe..
- Retour: void.

3) void GRAModifierSommetNumero(int eSommet,int eMSommet);
- Les paramètres: le numéro de sommet que vous voulez modifier; le numero de sommet
- Fonction: modifier le numéro d'un sommet de "eSommet" à "eMSommet".
- Retour: void.

4) void GRAModifierSommet(CSommetOperation &sommet,int ePos);
- Les paramètres: la référence d'un sommet; la position spécifique..
- Fonction: modifier le sommet du graphe.
- Retour: void.

5) void GRAModifierArc(int ePart, int eArri, int eMArri);
- Les paramètres: le numéro de départ; le numéro d'arrive; le numéro d'arrive après modifier.
- Fonction:modifier le arc "ePart ---> eArri", après il devient "ePart ---> eMArri".
- Retour: void.

6) void GRASupprimerSommet(int eElement);
- Les paramètres: le numéro de sommet.
- Fonction: supprimer un sommet du graphe.
- Retour: void.

7) void GRASupprimerArc(int ePart, int eArri);
- Les paramètres: le numéro de départ; le numéro d'arrive.
- Fonction: supprimer un arc du graphe.
- Retour: void.

3.6 la classe CFichier

Cette classe est pour lire un fichier. Cette classe comporte six methodes. La visibilité est <public>. Ils sont:

1) CGrapheOperation &FICLireFichier(char* pcFileNom);
- Les paramètres: le nom du fichier.
- Fonction: lire un fichier pour obtenir un graphe.
- Retour: la référence du graphe.

2) int FICLireValeur(char* pcFileNom, int eLigne, char cCaractere);
- Les paramètres: le nom du fichier; le numéro de la ligne; un caractere.

- Fonction: lire la ligne spécifique jusqu'a le caractere "cCaractere", pour obtenir le valeur d'attribut.

- Retour: le valeur.

3) int FICLireValeur(char*pcFileNom,int eLigne,char cCaractere1, char cCaractere2);

- Les paramètres: le nom du fichier; le numéro de la ligne; deux caractere.

- Fonction: lire la ligne spécifique, entre le caractere "cCaractere1" et "cCaractere2", pour obtenir le valeur d'attribut.

- Retour: le valeur.

4) char* FICLireAttribut(char* pcFileNom, int eLigne, char cCaractere);

- Les paramètres: le nom du fichier; le numéro de la ligne; un caractere.

- Fonction: lire la ligne spécifique jusqu'à le caractere "cCaractere", pour obtenir l'attribut.

- Retour: l'attribut.

5) bool FICDeterminerFichierType(char*str1, char*str2, int eLength);

- Les paramètres: deux "string"; la taille que vous voulez comparer.

- Fonction: comparer les deux "string" sur quelque tailles pour déterminer le contenu de fichier.

- Retour: la taille d'attribut.

6) int FICTaileDeAttribut(char* str);

- Les paramètres: un pointeur d'attribut.

- Fonction: obtenir la taille d'attribut.

- Retour: la taille d'attribut.

3.7 La classe Cexception

Nous utilisons cette classe pour gérer les exception.

3.7.1 attributs:

Cette classe comporte un attribut. La visibilité est <private>.

1). int eErreur: cet attribut comporte le numérotation d'exceptions. Chacun représente
les différents types d'erreur.

3.7.2 méthodes

Cette classe comporte sept méthodes. La visibilité est <public>.

1) Cexception();

-fonction: le constructeur

2) Cexception(int arg);

-fonction: le constructeur.

3) int &CEXLireErreur();

- Fonction: lire le valeur d'attribut - eErreur

- Retour: le valeur d'attribut - eErreur.

4) void CEXOuvreFile();

- Fonction: Si le fichier ne peut pas ouvrir, il va throw une exception et afficher le raison d'exception.
- Retour: void

5) void CEXFileType();

- Fonction: si le contenu de fichier n'est pas correct, il va throw une exception et afficher le raison d'exception. Par exemple, les informations de graphe ne sont pas assez .
- Retour: void

6) void CEXErreurGeneral();

- Fonction: si il ne trouve pas du sommet, il va throw une exception et afficher le raison d' exception.
- Retour: void

7) void CEXRepeter();

- Fonction: si le sommet déjà existe, il va throw une exception et afficher le raison d'exception.
- Retour: void

4. Résultat

Nous utilisons la matrice pour afficher le graphe, la première ligne présente la destination et la première colonne présente le début. Si Début = 5, Fin = 8, la coordonnée correspondante est 1(figure 4.0.1).

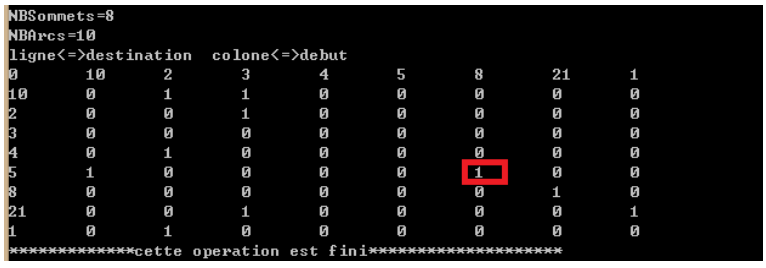

figure4.0.1

4.1 Les exceptions

1) Si le chemin de fichier est faux ,il y a une exception.(figure4.1.1)

figure4.1.1

2) Si le numéro de sommet et le numéro de arc est incorrect, il y a une exception.(figure4.1.2)

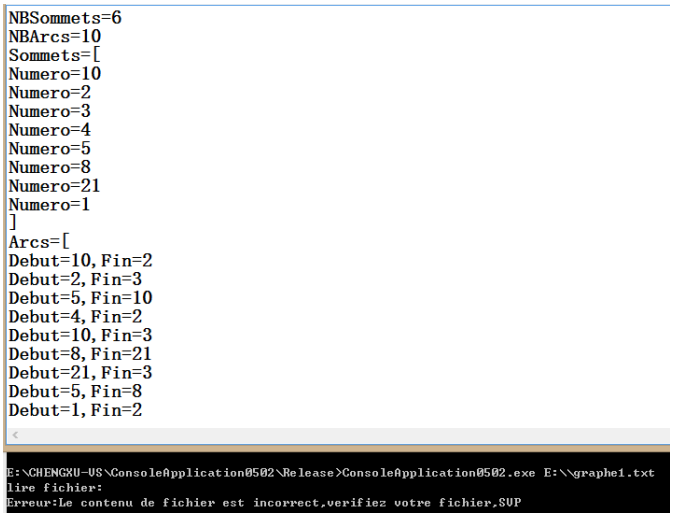

figure4.1.2

NBSommets est 6, mais il y a 8 sommets dans ce fichier.

3) Si il y a deux même numéros de sommet, il y a une exception.(figure4.1.3)

figure4.1.3

3) Si le numéro d'Arc ne existe pas dans le Sommet, il y a une exception. (figure4.1.4)

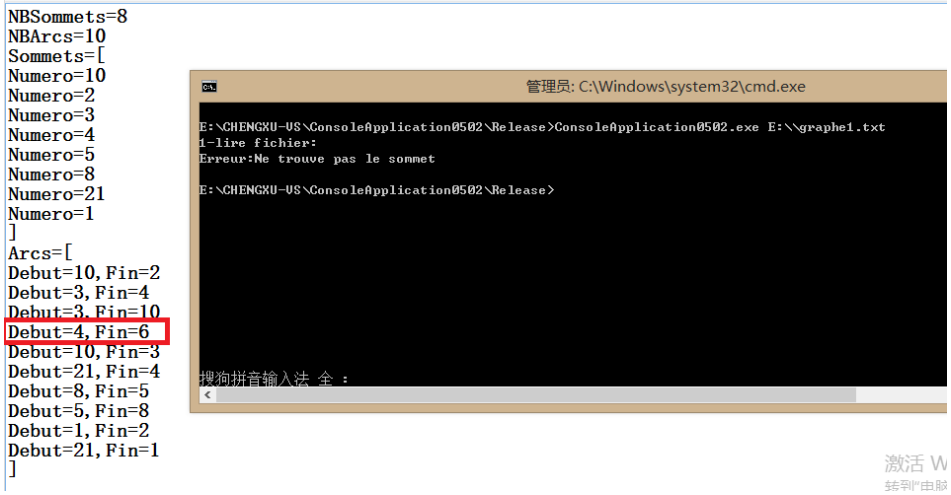

figure4.1.4

4.2 Test

Nous entrons le fichier de <release>et le fichier <.exe> par la commande ,et puis nous le passons de paramètres un fichier par le chemin de fichier.

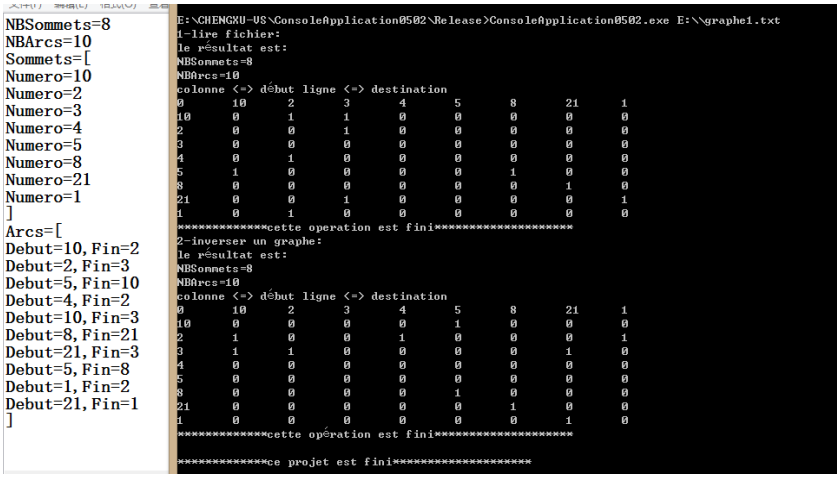

figure4.2.1

Après la presse de <enter>, il affiche le graphe originale et le graphe inversé automatiquement. (figure4.2.1)

Nous aussi testons plusieurs fonctions qui sont <ajouter/modifier/supprimer un sommet au graphe> et <ajouter/modifier/supprimer un arc au graphe>. Ils fonctionnent.

5. Référence

[1] MR Vincent T'kindt, Le document de 《Projet C++ : Les graphes》,Ecole Polytechnique de l'Université de Tours , 04/2017.