TD : Classes
Algorithmie
Master EEA 1ère année (2006)

1. Introduction

1.1. Edition du fichier

Afin d'écrire votre script C++, vous pouvez utiliser n'importe quel éditeur de texte, cependant xemacs ou nedit paraissent être un bon choix. On lance l'éditeur à partir d'une console (« nedit & »).
Le fichier aura une extension .cpp.

1.2. Compilation

Ouvrez un terminal et placez-vous dans le répertoire dans lequel vous avez enregistré votre script C. Afin de compiler votre script, utiliser la commande :
g++ nom_du_fichier.cpp -o nom_executable

Ce qui donne, par exemple, si votre script du exo1 se nomme exo1.cpp et que vous voulez que votre exécutable se nomme exo1:
g++ exo1.cpp -o exo1

Le compilateur génère alors un fichier exo1 que vous pouvez exécuter en utilisant la commande ./exo1

1.3. En tête de fichier

L'en-tête du fichier d'extension « .cpp » commencera par :

```
#include <iostream>
using namespace std;
```

La première ligne fait appel à la librairie relative aux fonctions d'entrées sorties.
La seconde ligne définit au compilateur que l'on va utiliser les librairies standard du langage C++.

2. Exercice 1

```
float value;
value = 10;

cout << "value = " << value << endl ;          (1)
cout << "&value = " << &value << endl ;       (2)
```

1. Que représente value ? Que va afficher la ligne 1 ?
2. Que représente &value ? Que va afficher la ligne 2 ?

3. Exercice 2

```
float *value;
```

1. Comment allouer la mémoire de value ?
2. Que représente &value ?
3. Que représente value ?
4. Que représente *value ?

4. Exercice 3

```
#include <iostream>
using namespace std;

struct point {
        int x;
        int y;
};

void init (int *a, int *b);

int main(int argc, char* argv[])
{
        struct point p, *pp;
        …
}

void init (int *a, int *b) {
        *a = 10 ;
        *b = 20 ;
}
```

1. Comment afficher la valeur de x de p ?
2. Comment afficher l'adresse mémoire dans laquelle est stockée la valeur de y ?
3. Comment allouer l'espace mémoire de pp ?
4. Comment afficher la valeur de x de pp ?
5. Comment afficher l'adresse mémoire de pp ?
6. Comment afficher l'adresse mémoire de y ?

5. Exercice 3

```
#include <iostream>
using namespace std;

class CPoint {
public:
        int x;
        int y;
        Cpoint();
        Cpoint(int a, int y);
};
```

```
int main(int argc, char* argv[])
{
        CPoint p, *pp;
        …
}

Cpoint::Cpoint (){
        x = 11 ;
        y = 21 ;
}
Cpoint::Cpoint (int a, int b) {
        x = a;
        y = b;
}
```

1. Comment afficher la valeur de x de p ?
2. Comment afficher l'adresse mémoire dans laquelle est stockée la valeur de y ?
3. Comment allouer l'espace mémoire de pp ?
4. Comment afficher la valeur de x de pp ?
5. Comment afficher l'adresse mémoire de pp ?
6. Comment afficher l'adresse mémoire de y ?
7. Qu'est ce qu'un constructeur par défaut ?

6. Exercice 4

Le but de cet exercice est de créer une classe CComplexe (z = a+ib). Cette classe va nous permettre de manipuler des nombres complexes.

1. Déclarer la classe CComplexe dans un fichier « complexe.h » constituée de ses paramètres a et b, ainsi que de son constructeur par défaut.
2. Créer un fichier « complexe.cpp » dans lequel vous implémentez le constructeur par défaut. On affichera à l'écran dans cette fonction la valeur d'un moins un des paramètres de la classe.
3. Créer un fichier « exo4.cpp » contenant la fonction main dans lequel on déclare un objet z de la classe CComplexe.
   La ligne de commande pour compiler et générer l'exécutable est :
   g++ exo4.cpp complexe.cpp –o exo4
4. Ajouter un constructeur permettant d'initialiser les valeurs de a et b (pouvant être différents de 0) en les passants comme paramètres.
5. Créez une méthode permettant de saisir au clavier un nombre complexe sous la forme partie imaginaire et partie réelle
6. Créer une méthode qui affiche à l'écran un nombre complexe sous la forme partie imaginaire et partie réelle.