C++: TP 01 les classes

DAKKAR Borhen-eddine
Lycée le Corbusier
BTS SN

September 7, 2021

1 Objectifs du TP

Dans ce TP nous allons voir un concepte fondamental de la POO. Il s'agit des **classes**.

2 Logiciels à utiliser

Sous Linux les outils nécessaire à la compilation du C++ sont déjà présents, il s'agit de "gcc" et d'un éditeur de texte "gedit" par exemple.
Pour compiler un fichier source "main", il suffit d'utiliser la commande suivante:

```
g++  *.cpp -o main
```

Pour l'exécution de fichier ".exe", il suffit d'utiliser la commande suivante:

```
./main
```

3 Exercice 1:

Ecrire une classe appelée **Rectangle**. Cette classe contient cinq membres:

- Deux membres de données de type int:
  - longueur: privé;
  - largeur: privé;
- Deux fonctions membres avec accès public:
  - **Rectangle** (constructeur): il permet d'initialiser la longueur et la largeur.
  - **surface**: qui permet de calculer la surface d'un rectangle.
  - **périmètre**: qui permet de calculer le périmètre d'un rectangle.

Ecrire un programme **main()** pour tester le fonctionnement. Tester avec les valeurs $longueur = 3$ et $largeur = 4$.

4 Exercice 2:

Écrire une classe appelée **Forme** avec un constructeur qui donne la valeur à la **largeur (larg)** et à la **hauteur (haut)**. La classe **Forme** possède aussi une méthode **aire (char)** qui calcule l'aire d'une forme donnée. Dans le **main**, définissez un triangle et un rectangle de type **Forme**, puis appelez la méthode **aire** pour calculer leur aire.
Notez que l'aire d'un triangle est $= largeur * hauteur/2$ et l'aire d'un rectangle est $= largeur * hauteur$.

5 Exercice 3:

Réaliser une classe **point** permettant de manipuler un point d'un plan. On prévoira :

- Un constructeur recevant en arguments les coordonnées (float) d'un point;
- Une fonction membre **deplace** effectuant une translation définie par ses deux arguments (float);

- Une fonction membre affiche se contentant d'afficher les coordonnées cartésiennes du point.

Les coordonnées du point seront des membres données privés.
On écrira séparément :

- Un fichier source constituant la déclaration de la classe (.h);
- Un fichier source correspondant à sa définition (.cpp).

Écrire un programme d'essai **main** déclarant un point, l'affichant, le déplaçant $(dx = 3, dy = 2)$ et l'affichant à nouveau.

6 Exercice 4:

Modifier la classe **point** de l'exercice précédent en rajoutant deux fonctions membre publiques (nommées **abscisse** et **ordonnee**) fournissant en retour l'abscisse et l'ordonnée d'un point. Les deux fonctions doivent respecter le principe d'encapsulation des données.
Ecrire un programme d'essai qui fonctionne avec cette nouvelle classe.

7 Exercice 5:

L'objectif de cet exercice est de simuler le fonctionnement d'un ascenseur. La sortie requise du progaramme décrit l'étage actuel sur lequel l'ascenseur est stationné ou passé par. De plus, il faut fournir un bouton interne de l'ascenseur qui sera poussé pour demander un autre étage. L'ascenseur doit être identifié par un numéro, tel que l'ascenseur numéro 1 (pour permettre de placer d'autres ascenseurs supplémentaires en service, si nécessaire), et il peut circuler entre le 1^{er} et le $15^{ème}$ étage du bâtiment dans lequel il est situé.
Pour cette simulation, il faut prévoir:

- Trois membres privés:
  - **Num_Asc**: un entier indiquant le numéro de l'ascenseur;
  - **etage_act**: un enteir indiquant l'etage actuel;
  - **der_etage**: un entier indiquant le numéro du dernier étage;
- Deux membres public:

– Ascenseur: un constructeur;
– **void demande_ascenseur(int)**: un méthode qui permet l'appel de l'ascenseur;