TP N8 : Révision C++

Exercice 1 :

Soit la classe complexe pour gérer les nombres complexes :

Attributs:

Re la partie réelle de type double

Img la partie imaginaire de type double

Méthodes:

- Constructeur avec arguments (valeurs d'initialisation par default)
- Fonction d'affichage
- Fonction qui retourne le module sachant que :

$$|z| = \sqrt{a^2 + b^2}$$

- Fonction qui retourne le conjugué sachant que :

$$\bar{z} = a - bi$$

Surcharge des opérateurs suivants :

- "+" : (complexe + complexe), en utilisant des fonctions membre.
- "+" pour (complexe + double) et "+" pour (double + complexe), en utilisant des fonctions membre.
- De même pour "*", "-" , en utilisant des fonctions amies.
- Créer un programme de test main ().

Exercice 2:

L'objectif de cet exercice est de gérer les notes des étudiants d'une institution à l'aide d'une classe C++ Etudiant définie par :

Les attributs suivants :

- matricule: l'identifiant de l'étudiant (auto incrémenté)
- nom: nom d'un étudiant
- nbrNotes: le nombre de notes de l'étudiant
- *tabNotes: tableau contenant les notes d'un étudiant (allocation dynamique).

Les méthodes suivantes :

1. Un constructeur d'initialisation
2. Un constructeur avec arguments
3. Un destructeur ~Etudiant ()
4. Un constructeur de recopie Etudiant (const Etudiant &)
5. void saisie () : permettant la saisie des note d'un étudiant
6. void affichage () : permettant l'affichage des informations d'un étudiant
7. float moyenne () : retourne comme résultat la moyenne des notes de l'étudiant.
8. int admis () : retourne comme résultat la valeur 1, si un étudiant est admis et la valeur 0, sinon. Un étudiant est considéré comme étant admis lorsque la moyenne de ses notes est supérieure ou égale à 10.
9. bool comparer(): qui compare la moyenne des deux étudiants, retourne comme résultat la valeur true, si deux étudiants ont la même moyenne et la valeur false, sinon.
10. Créer un programme de test main () qui contient :
    - Création d'un objet Etudiant E avec 3 notes
    - Copie de l'étudiant E dans E1 et afficher l'étudiant E1
    - Création d'un objet Etudiant E2 avec 2 notes
    - Calculer la moyenne de chaque étudiant et vérifier si les deux étudiants sont admis
    - Comparer les deux étudiants

Exercice 3:

On souhaite créer une application pour la gestion des factures d'un magasin pour cela on vous demande d'écrire 3 classes de cette application.

Dans ce magasin un article est identifié par :

- code (int)
- désignation (string)
- prix (double)
- catégorie (string): ne peut prendre que les valeurs « Informatique » ou « Bureautique »

1. Ecrire la classe Article :

a. Ecrire les attributs (ils doivent être visibles dans les classes filles de la classe article).

b. Ecrire le Constructeur par défaut et d'initialisation.

c. Ecrire la méthode virtuelle getPrix() pour retourner le prix de l'article.

d. Ecrire la méthode setPrix(double): pour changer le prix de l'article.

e. Ecrire la méthode toString() qui renvoie toutes les propriétés séparées par un point-virgule.

f. Ecrire la méthode Equals() :2 articles sont égaux s'ils ont les mêmes propriétés.

Un article en solde comprend une information additionnelle :

- Remise : pourcentage de réduction sur le prix d'origine

2. Ecrire la classe ArticleEnSolde.

a. Ecrire les Attributs.

b. Ecrire les Constructeurs par défaut et d'initialisation.

c. redéfinir la méthode getPrix(), afin de tenir compte du solde.

Le magasin établit des factures numérotées automatiquement.

Une facture est identifiée alors par :

- Numéro de la facture (automatiquement incrémenté).
- Date facture.

- Collection d'achats.

On considère une classe achat ayant comme attributs l'article acheté et la quantité achetée (il n'est pas demandé d'écrire la classe Achat).

```
class Achat {

int numéro_achat;

 Article article_acheté;

quantité;

 public:

 Achat(int num, Article art, int q)

      {numeroachat = num;

      article_achete = art;

         quantite = q;}

};
```

3. Ecrire la classe Facture.

a. Ecrire les attributs .

b. Ecrire le Constructeur d'initialisation.

c. Ecrire la méthode void Ajouter (Achat a): permet d'ajouter un achat à la collectionachats, vérifier la non existence du même achat dans la collection, dans le cas contraire la méthode doit afficher un message d'erreur.

d. Ecrire la **Méthode Double Montant_facture():** retourne le montant total de la facture.

e. Ecrire **la Méthode Tostring()**qui permet d' afficher le numéro et la date de facture avec la liste des articles achetées de la façon suivante :

Numéro facture …… Date facture 15/05/2022

Liste des achats

Désignation prix(en DH) quantité prix Total

……

Montant de la facture :…….