Série 2

Exercice 1 :

On considère une entité ***Adresse*** décrite par un numéro, un nom de rue, un code postale et une ville.

a) Donnez une implémentation de l'entité ***Adresse***, en donnant la possibilité de retourner ou de modifier n'importe quel champ de l'entité par des méthodes définies dans la classe, prévoir une méthode d'affichage.

b) Créer une entité ***Citoyen*** décrite par un nom, prénom et adresse. Pour les méthodes, on demande de définir uniquement le constructeur et une méthode d'affichage.

c) Ecrire un programme qui crée un tableau ***C*** de 10 citoyens et affiche ensuite tous les citoyens (nom, prénom, adresse) habitant dans une même ville ***V*** donnée.

Exercice 2 :

On considère une entité ***Repertoire*** décrite par un nom, un nombre de fichiers contenus, une taille en nombre d'octets et une date de création du répertoire.

a) Donnez la description d'une entité ***Date*** en incluant une méthode ***superieur*** qui vérifie si une date est supérieure à une autre.

b) Donnez la description de l'entité ***Repertoire***, en donnant la possibilité de retourner ou modifier n'importe quel champ de l'entité par des méthodes définies dans la classe, prévoir une méthode d'affichage.

c) Ecrire un programme qui crée un tableau ***R*** de 20 objets ***Repertoire*** et affiche tous les répertoires vides créés avant une date ***D*** donnée.

Exercice 3 :

On considère une entité ***Message*** décrite par un identifiant, un expéditeur, un destinataire, un contenu (chaines de caractères) et une date d'envoi.

a) Donnez la description d'une entité ***Date*** en incluant une méthode **egal** qui vérifie l'égalité entre deux dates.

b) Donnez la description de l'entité ***Message***, en donnant la possibilité de retourner ou modifier n'importe quel champ de l'entité par des méthodes définies dans la classe, prévoir une méthode d'affichage.

c) Ecrire le programme qui crée un tableau ***M*** de 20 messages et affiche tous les messages dont la date d'envoi est égale à une date ***D*** donnée.

Exercice 4 :

Définir une classe ***Pile*** décrite par un tableau T d'entiers et sommet qui désigne la position de l'élément se trouvant au sommet. Définir les méthodes suivantes :

- affiche : afficher des éléments de la pile,
- empiler : rajouter un élément dans la pile,
- depiler : supprimer un élément de la pile,
- pileVide : retourne 1 si la pile est vides 0 sinon,
- pilePleine : retourne 1 si la pile est pleine 0 sinon,
- sommetpile : retourne la valeur qui se trouve au sommet.

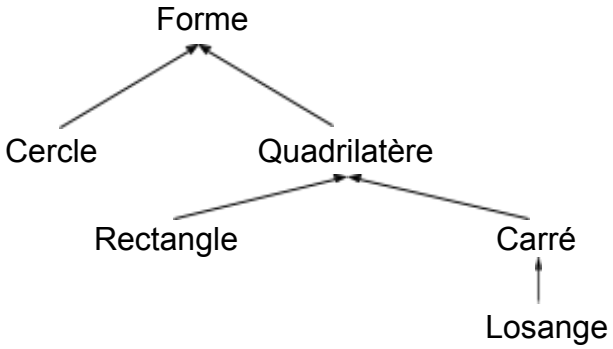
Ecrire un programme qui étant donné une pile d'entiers détermine la valeur maximale et la supprime de la pile. (supprimer aussi toutes las valeurs égales à la valeur maximale).

Exercice 5 : (Examen 2009-2010)

Partie I :

1. Définir une classe ***Point*** pour représenter les coordonnées d'un points caractérisées par deux nombres flottants x, y, mémorisés dans deux variables d'instance privées (de type double). La classe Point comportera les méthodes d'instance suivantes :
   - un constructeur recevant en argument les coordonnées d'un point,
   - les accesseurs qui renvoient/modifient les coordonnées du point,
   - ***String toString();*** qui transforme un point en une chaîne de caractères de la forme **"(x,y)"**, en vue de son affichage,
   - ***void affiche();*** qui affiche un point.

2. Définir les classes qui respectent la hiérarchie suivante:



- ***classe Forme :*** Formes géométriques.

Attributs : Le point centre de la forme
Deux Constructeurs : L'un initialise le centre à 0, l'autre initialise le centre à (x,y)
Méthodes : - *String toString() ;* retourne le point (x,y) en vue de son affichage
- *void afficheForme() ;* affiche une forme ainsi "Le centre de cette forme est : (x,y)"

- *double surface() { return 0 ;* } // On retourne 0 car on ne sait // pas de quelle forme il s'agit
- *double périmètre() { return 0 ;* } // Idem

- ***classe Cercle :*** Sous classe de la classe Forme.

Attributs : Un cercle est identifié par son centre (Point) et son rayon
Un constructeur : Avec arguments
Accesseur : Ceux du rayon
Méthodes :
- *double surface() ;*
- *double périmètre() ;*
- *boolean estDansCercle(Point p) ;* vérifie si p est ou non à l'intérieur du cercle, (x,y) est dans le cercle de rayon R ssi $x^2 + y^2 \leq\leq R^2$
- *void afficheForme() ;* affiche "Cercle : Le centre de cette forme est (x,y)"

- ***classe Quadrilatère*** : Sous classe de la classe Forme.

Attributs : Un quadrilatère est identifié par 4 points a, b, c, d.
Constructeur : Avec arguments
Méthodes : - *static Point milieu(Point p1, Point p2) ;*
- *static double distance(Point p1, Point p2) ;*
- *boolean parallélogramme(double epsilon) ;* vérifie si un quadrilatère est un parallélogramme
- *void afficheForme() ;* affiche "Quadrilatère : Le centre de cette forme est (x,y)"

- ***classe Rectangle*** : sous classe de la classe Quadrilatère.

Attributs : Un rectangle est identifié par 4 points et 2 cotés, longueur et largeur.
Constructeur : Avec arguments
Accesseurs : Ceux de longueur et largeur
Méthodes : - *double surface() ;*
- *double périmètre();*
- *void afficheForme() ;* affiche "Rectangle : Le centre de cette forme est (x,y)"

- ***classe Carré*** : sous classe de la classe Quadrilatère.

Attributs : Un carré est identifié par 4 points et un coté.
Constructeur : Avec arguments
Accesseurs : Ceux du coté
Méthodes : - *double surface() ;*
- *double périmètre();*
- *void afficheForme() ;* affiche "Carré : Le centre de cette forme est (x,y)"

- ***classe Losange*** : Sous classe de la classe Carré.

Attributs : Un losange est identifié par 4 points et un coté.
Constructeur : Avec arguments
Méthodes : - *double surface() ;*
- *double périmètre();*
- *void afficheForme() ;* affiche "Losange : Le centre de cette forme est (x,y)"

Ecrire le programme qui :

a) Étant donné un tableau TF de N Formes (N<=100), affiche pour chaque Forme s'il s'agit d'un Cercle, Rectangle, Carré, Losange puis affiche leur surface et leur périmètre.
b) A votre avis quelle surface et périmètre affichera votre programme si la forme est un Quadrilatère ?
c) Étant donné un tableau TQ de M Quadrilatères (M<=100), affiche pour chaque quadrilatère s'il s'agit d'un rectangle, carré, losange ou aucun des trois c'est-à-dire un quadrilatère quelconque en effectuant les tests nécessaires.

Partie II :

Si on décide d'écrire les méthodes ***milieu*** et ***distance*** dans la classe ***Point*** (au lieu de la classe *Quadrilatère*) quelles seront les modifications à faire ? Réécrire les classes concernées.

Partie III :

Si dans la classe ***Forme*** on ne met que les prototypes des méthodes ***surface()*** et ***périmètre()*** quels sont les modifications à faire dans votre programme ? Réécrire les classes concernées.