TP N° 2

Exercice 1

Classe Ident

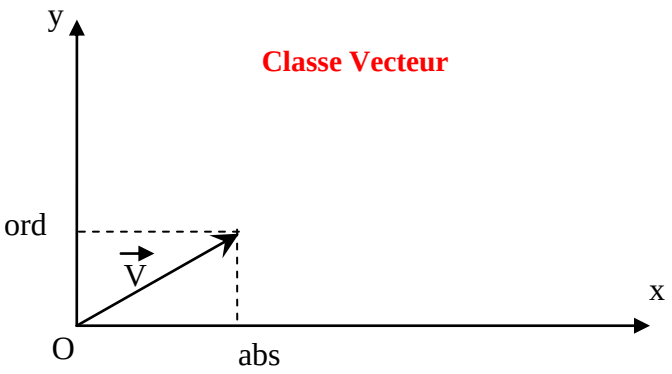
Réaliser une classe qui permet d'attribuer un **numéro** unique à chaque nouvel objet créé (1 au premier, 2 au suivant...). On dotera la classe d'un **constructeur** et d'une méthode **getIdentMax** qui retourne le **numéro du dernier objet** créé.

Classe TestIdent

Écrire un programme principal qui permet de créer 3 objets a, b et c et d'afficher :

```
numéro de a : 1
numéro de b : 2
dernier numéro : 2
numéro de c : 3
numéro de a : 1
```

Exercice 2



Classe Vecteur

Il s'agit de modéliser un vecteur $\vec{V}$ dont l'origine est en (0, 0) (un tel vecteur est donc caractérisé par deux nombres entiers **abs** et **ord**). En plus du constructeur avec paramètres, les opérations que l'on souhaite faire sur ce vecteur sont :

- Calculer sa longueur, par une méthode, nommée **longueur**(), et qui retourne cette longueur sous forme d'un double. Longueur de $\vec{V}$(abs, ord) est : **Math.sqrt(x * x** + y * y).
- Savoir si un vecteur est ou non plus petit qu'un autre vecteur donné, en comparant leurs longueurs. Ecrire pour cela une méthode nommée **plusPetitQue**(..) qui recevra en paramètre l'autre vecteur et qui retournera un booléen.
- Additionner à un vecteur un autre vecteur ; on écrira pour cela une méthode nommée **addition**(..) qui recevra en paramètre l'autre vecteur et qui ne retournera rien.
- Additionner deux vecteurs donnés ; on écrira pour cela une méthode **statique** nommée aussi **addition** (..) (surcharge) qui recevra en paramètres les deux vecteurs à additionner et qui retournera le résultat sous forme d'un objet Vecteur
- une méthode redéfinissant la méthode : public String **toString**() de la classe Object. Celle-ci décrira une instance de Vecteur sous la forme d'une chaîne de caractères (par exemple, le vecteur de coordonnées 1 et 2 pourra être décrit par la chaîne de caractères : "Vecteur (1, 2)"). La méthode retournera cette chaîne.
- Que retourne **toString()**, si elle n'a pas été redéfinie ?

Classe TestVecteur

- On créera une classe **TestVecteur** contenant la méthode main pour tester la classe Vecteur. Les coordonnées des objets vecteurs créés, seront données directement au constructeur de la classe Vecteur.

Un exemple d'exécution de ce programme peut être:

```
v1 : Vecteur (1, 2)
Longueur de vecteur (1, 2) : 2.23606797749979
v2 : Vecteur (5, -3)
Vecteur (1, 2) est plus petit que le Vecteur (5, -3)
v1 après addition de v2 : Vecteur (6, -1)
v3 = v1 +  v2 : Vecteur (11, -4)
```

N.B. :
Si v est une instance de la classe Vecteur, alors "System.out.println(v)" permet d'afficher ce que retourne la méthode toString().

Exercice 3

Classe : **Score**

1. Créez une classe qui représente le score d'un joueur. Cette classe, nommée **Score**, devra gérer trois données : le **nom** du joueur, **son score (sonScore)**, le **score maximal (scoreMax)** autorisé.
2. Ecrivez une méthode "**afficher ()**" de cette classe qui affiche le nom et le score du joueur.
3. Créez un **constructeur** de cette classe où le nom par défaut serait "inconnu", le score initial serait 0 et le score maximal serait 100.
4. Pour ne pas créer que des inconnus, écrivez un second constructeur (surcharge) qui fixera le nom du joueur. Eliminez la **redondance** de codes entre les 2 constructeurs.
5. Dans un programme on peut lire ou modifier le contenu du score du joueur. Pour permettre d'agir sur lui en tenant compte de ses limites (0 <=sonScore <= scoreMax) vous devez ajouter deux nouvelles méthodes : l'une de lecture "**lireScore()**" (est une fonction qui retourne le score) et l'autre de modification "**modifierScore(int sco)**". En plus ajouter une troisième nouvelle méthode "**ajouter(int points)**" permettant d'ajouter des points au score.

Classe : **ScoreEssai**

1. Pour les besoins d'un programme de jeu particulier, pour pouvoir gérer le score d'un joueur, et aussi le nombre d'essais qui ont été effectués pour obtenir ce score, créez une classe dérivée de la classe score (**ScoreEssai**) qui prend en compte le **nombre d'essais (nbEssai)**.
2. Dans cette nouvelle classe créer deux constructeurs en complétant les constructeurs de la classe mère (utilisation du mot clé "**super**").
3. Redéfinissez la méthode **modifierScore(int sco)** qui complète le code de celle de la classe Score (utilisation du mot clé "**super**") pour que chaque modification du score s'accompagne d'une augmentation du nombre d'essais.
4. Ajouter une méthode **lireNbEssais()** qui permet de retourner le contenu de l'attribut **nbEssai**.

Classe : **Jeu**

1. Ecrire la classe principale **Jeu** qui simule un jeu qui consiste à lancer un dé jusqu'au moment où on obtient un total supérieur ou égal à 21 points. Afficher le nombre d'essais.

N.B. :

Syntaxe de la boucle while est : while (condition) { … }

Double d; //Double est une classe "réelle". La méthode "d.intValue()" renvoie la partie entière de la
//valeur réelle de d.
..
d=new Double(6*Math.random()+1); // permet de créer un objet Double ayant une valeur aléatoire réelle
// entre 1 et 6.99.. .