HERITAGE

1. Exercice : Héritage en java

Ecrire un programme java appelée **GestionExamen** qui permet d'afficher les 3 phrases ci-dessous :

Je passe l'examen des matières générales
Je passe l'examen d'informatique
Je passe l'examen des matières générales

PROCEDURE

- Déclarez la classe ***Examen***, dans sa définition on déclare la fonction ***passerExamen( )*** qui ne retourne **rien** et encapsulé **public**, dans la définition de la fonction **passerExamen**(),

Affichez : ***Je passe l'examen des matières générales***

- Déclarez la classe ***ExamenInfo***, dans sa définition on déclare la fonction ***passerInfo( )*** qui ne retourne **rien** et encapsulé **public**; dans sa définition, affichez:

Je passe l'examen d'informatique

Cette classe **ExamenInfo** *hérite* la classe **Examen**

- N'oubliez pas de déclarer et de définir la classe principale, appelée **GestionExamen** qui va appeler toutes les fonctions des autres classes via l'instance de classe.

2. Exercice : Héritage en java

Déclarer la classe **Animal**, dans sa définition, déclarer et définir la fonction **marcher()** qui ne retournent rien et qui affiche : **Je marche**.

Déclarer et définir la classe **Chien** qui hérite de la classe **Animal** et qui affiche dans la fonction **afficheChien()**: **Je cours**

Déclarer et définir la classe **Chat**, qui hérite de la classe **Animal** et qui affiche : ***Je miaule***

1. ***Afficher***

Je marche
Je cours
Je marche
Je miaule
Je marche

SURCHARGE

- Une surcharge consiste à implémenter (déclarer et définir) une fonction plusieurs fois *au sein d'une même classe* en échangeant les nombres d'***arguments.***

  On peut changer le type retour, mais ça n'a aucune incidence sur la définition

- On les différencie juste par le nombre d'argument(surcharge), le système saura les différencier selon le nombre d'argument.

- Lors de l'appel de la fonction, java va proposer la fonction selon le nombre d'arguments.

Remarque :
La surcharge peut se faire aussi bien dans une classe classique que dans une classe fille

3. Exercice : surcharge de la somme()

Ecrire un programme java : **GestionCalcul**, qui permet de calculer la somme.

Dans la classe **Calcul**, on déclare et définit 2 fois la méthodes **somme**() :

- La méthode **somme()** avec 2 arguments (entiers) qui retourne la valeur de la somme
- La méthode **somme()** avec 3 arguments (entiers) qui retourne la valeur de la somme avec une phrase

```
String   res = "la somme est"+(a+b+c) ;
```

POLYMORPHISME → Héritage

Le polymorphisme résulte exclusivement de l'héritage, et c'est la possibilité de redéfinir dans la classe **fille** une méthode qui existait déjà dans la classe **mère**, dans ce cas, on dit qu'on a fait un ***override*** de la méthode de la classe mère, c'est-à-dire qu'on a ***annulé***, **inhibé** de la méthode de la ***mère***.

4. Exercice : polymorphisme

Ecrire un programme java qui contient :

1. a. la classe **Personne**, contenant la méthode **affiche**(), cette méthode permet d'afficher la phrase :

   C'est la classe mère

   b. La classe **Employe** hérite de la classe **Personne**, on redéfinit la méthode **affiche**(), dans sa définition, on affiche :

   C'est la classe fille

2. On souhaite appeler la méthode de la mère qui affiche la phrase de la mère :

   C'est la classe mère

   Malgré que cette méthode soit déjà redéfinie dans la classe fille.

   Pour ce faire, appeler dans la classe fille, la méthode non écrasée de la mère grâce au mot clé **super** :

   super.nomDeLamethode()

RESULTAT :

C'est la classe mère
C'est la classe fille
C'est la classe mère