KOUIDRI Yanis | JAVA | S2 | R&T 2020-2021

Une classe : public class <nomClasse> {
...
}

Pour tester une classe : public static void main (String[] args)
{
...
}

Les constantes : public static final <type> <NOM_EN_MAJ> = "valeur" ;

Les variables d'instance : private <type> <nom> ;

Exemple :
- Constante : public static final String DEF_NOM = "Inconnu" ;
- variable d'instance : private boolean majeur ;

Constructeur : public <nomClasse> (<type> <nom1>, <type> <nom2>){
<nom> = <nom1> ;
...
}

Constructeur par défaut : public <nomClasse>() {
<nom> = <NOM_EN_MAJ> ;
}

Accesseur en lecture (Pour toutes les variables d'instance) :

public <type> get<nom>(){
return <nom> ;
}

Accesseur en écriture (uniquement quand nécessaire) :
public void set<nom> (<type> <unNom>) {
<nom> = <unNom>

Un constructeur par défaut ou non.

Créer un objet : <nomClasse> <UnNom> = new <nomClasse>() ;

Utilisation de constantes : <nomClasse>.<NOM_EN_MAJ>

Utilisation accesseur en lecture (la seule chose que on écrit) :
<UnNom>.get<nom>() ;

Méthode toString() : public String toString() {
return <nom> + <nom> ;
}

utilisation : System.out.println(<Un Nom Objet>) ;

Autres : equals(chaine 1) pour comparer 2 chaînes de caractères.

Tableaux : NB est un constante déclaré de type int, "NB" ∈ IN

Déclaration : <type> <nomtab>[] ;

Allocation mémoire : <nomtab> = new <type>[NB] ;

Initialisation des éléments : for (int i=0 ; i<tab1.lenght ; i++) {
tab1[i] = 0 ;
}

Pour afficher tous les éléments d'un tableau :
Même chose que pour l'initialisation mais remplacer
tab1[i] = 0 ; par System.out.println(tab1[i]) ;

Définition :

Static : static signifie qu'il s'agit d'une variable de classe c-à-d d'une variable partagé par toutes les instances d'une classe.

final : final signifie que la variable ne changera pas de corps c'est une constante