Création d'une class

```
public class Class_exemple {

}
```

Création d'une méthode

```
public static void methode_exemple (String[] args) {

}
```

Variable

Type de variable :

Numérique : byte, short , int et long

Décimaux : float et double

Type caractère : char

Exemple :

int poids = 70;

float pi = 3.14;

Niveaux accès variable :

private : Les membres déclarés comme private sont accessibles uniquement à l'intérieur de la classe où ils sont déclarés. Ils ne sont pas accessibles en dehors de cette classe, même pas pour les classes héritées.

protected : Les membres déclarés comme protected sont accessibles dans la même classe, dans les classes du même package et dans les sous-classes (classes filles ou dérivées) même si elles sont situées dans un autre package.

public : Les membres déclarés comme public sont accessibles partout. Ils peuvent être utilisés par n'importe quelle classe.

static : Le mot-clé static est utilisé pour déclarer une variable ou une méthode qui appartient à la classe plutôt qu'à une instance spécifique de cette classe. Les variables et les méthodes statiques peuvent être accédées sans avoir besoin d'instancier la classe. Elles sont partagées par toutes les instances de la classe.

Exemple :

```
private int nombre;
```

Conversion de types :

```
double distance;

int conversion;

distance=2335.89;

conversion = (int)distance;
```

Constante :

```
final int TVA=21;
```

Opérateurs

Liste opérateurs unaires :

Opérateur	Action
-	Valeur négative
~	Complément à un
++	Incrémentation
--	Décrémentation
!	Négation

Opérateur d’affectation :

Operateur	Opération réaliser	Exemple	Résultat
+	Addition pour les valeurs numériques ou concaténation pour les chaînes	6+4	10
-	Soustraction	12-6	6
*	Multiplication	3*4	12

/	Division	25/3	8.333333
%	Modulo (reste de la division entirère)	25%3	1

Operateurs logiques :

Opérateur	Opération	Exemple	Résultat
&	Et logique	If ((test1) & (test2))	Vrai si test1 et test2 est vrai
|	Ou logique	If ((test1) | (test2))	Vrai si test1 ou test2 est vrai
^	Ou exclusive	If ((test1) ^ (test2))	Vrai si test1 ou test2 est vrai mais pas si les deux sont vrais simultanément
!	Négation	If (! Test)	Inverse le résultat du test
&&	Et logique	If((test1) || (test2))	Idem et logique mais test2 ne sera évalué que si test1 est vrai
||	Ou logique	If ((test1) || (test2))	Idem ou logique mais test2 ne sera évalué que si test1 est faux

Opérateurs de comparaison :

Opérateur	Opération réalisée	Exemple	Résultat
==	Egalité	2 == 5	False
!=	Inégalité	2 != 5	True
<	Inférieur	2 < 5	True
>	Supérieur	2 > 5	False
<=	Inférieur ou égal	2 <= 5	True
>=	Supérieur ou égal	2 >= 5	False
instanceof	Comparaison du type de variable avec le type indiqué	O1 instanceof Client	True si la variable O1 référence un objet créé à partir de la classe client ou d'une sous-classe

Operateurs binaires :

Opérateur	Opération réalisée	Exemple	Résultat
&	Et binaire	45 & 255	45
|	Ou binaire	99 | 46	111

^	Ou exclusif	99 ^ 46	77
>>	Décalage vers la droite (division par 2)	26>>1	13
<<	Décalage vers la gauche (multiplication par 2)	26<<1	52

Sortie

Effectuer une sortie :

```
System.out.println("Hello world");
```

Chaine de caractère

\t	tabulation
\b	BackSpace
\n	Saut de ligne
\r	Retour chariot
\f	Saut de page
\'	Simple quote
\"	Double quote
\\	Antislash

Création chaine de caractère :

```
String machaine;
```

Affectation chaine de caractère :

```
machaine = "abc"
```

Récupère un caractère par son indice :

machaine.charAt(3)

Récupérer le nombre d'élément d'une chaine de caractère :

machaine.length()

Récupérer une partie de la chaine de caractère :

machaine.substring(2,9)

Comparaison de chaine :

equales effectue une comparaison de la chaine en prennent en compte les majuscules et minuscule.

chaine1.equals(chaine2)

equalsIgnoreCase idem sans tenir compte des majuscules et minuscules .

chaine1.equalsIgnoreCase (chaine2)

Suppression des espaces de début et de fin :

machaine = machaine.trim();

Changer les cases :

Passer en minuscule :

machaine.toLowerCase()

Passer en majuscule :

machaine.toUpperCase()

Rechercher caractère dans une chaine :

machaine.indexOf("s") (renvoye la position du caractere dans la chaine)

Remplacement d’une chaine dans une chaine :

machaine.replace("hugo","jean");

Exemple :

```
String machaine = "je suis hugo";

String autrechaine = machaine.replace("hugo","jean");

System.out.println(autrechaine);
```

Tableau

Bibliothèque utiliser :

```
import java.util.Arrays;
```

Déclaration variable de type tableau :

```
int[]tableau;

int[][]tableau2;
```

Création du tableau et de la taille du tableau :

tableau = new int[10]; (creation d'un tableau de 10 element)

tableau = new int[10][20]; (creation d'un tableau a 2 dimension de 200 element)

Trier le tableau par ordre croissant :

```
int []chiffreAffaire =
{1234,1214,3214,6621,5202,1042,14512,6314,15732,6021,3523,5224};
Arrays.sort(chiffreAffaire);
```

Rechercher un élément dans un tableau :

Arrays.binarySearch(chiffreAffaire,1234)

Copie d'un tableau vers un autre tableau :

int[] copychiffreaffaire = Arrays.copyOf(chiffreAffaire,12);

Copie partielle d'un tableau vers un autre tableau :

int[] premiertrimsestre = Arrays.copyOfRange(chiffreAffaire,0,3);

Tableau dynamique Arraylist :

Date

Bibliothèque utiliser :

import java.time.LocalDate;

Affecter une date a une variable :

LocalDate ete;

ete= LocalDate.of(2023,06,21);

Récupération de la date machine :

LocalDateTime maintenant;

maintenant = LocalDateTime.now();

Transformation d’une LocalDateTime en LocalDate :

LocalDateTime maintenant;

```
Aujourdhui = LocalDate.from(maintenant) ;
```

Affectation d’une chaine de caractère dans une variable date :

```
LocalDate date;
date = LocalDate.parse("2001-01-12");
```

Condition

Strcuture if :

```
if(condition){
instruction ;
}
else if (Condition 2){
// Bloc d’instruction
}
else{
//Bloc d’instruction}
```

Structure switch :

exemple :

```
int jour = 3;
switch (jour){
    case 1:
        System.out.println("Le jour "+jour+"est le lundi");
        break;
    case 2:
        System.out.println("Le jour "+jour+"est le mardi");
        break;
```

```
        case 3:
            System.out.println("Le jour "+jour+"est le mercredi");
            break;
        case 4:
            System.out.println("Le jour "+jour+"est le jeudi");
            break;
        case 5:
            System.out.println("Le jour "+jour+"est le vendredi");
            break;
        case 6:
            System.out.println("Le jour "+jour+"est le samedi");
            break;
        case 7:
            System.out.println("Le jour "+jour+"est le dimanche");
            break;
      default:
              System.out.println("Le jour de la semaine n’existe pas")
    }
```

Boucle

while :

```
while(condition)
{
//Bloc instruction
}
```

do while :

```
do
{
```

//Bloc d’instruction

}

while(instruction)

for :

for(initialisation variable, condition, instruction d’itération)

{

//Bloc d’instrucution

}