

<b>Aide - mémoire Java</b>
----------------------------

### Principaux opérateurs

<b>Opérateur d'affectation</b>
<ul style="list-style-type: none"> <li>• = : affecte (le membre de droite au membre de gauche)</li> </ul>
<b>Opérateurs arithmétiques:</b>
<i>Les basiques :</i>
<ul style="list-style-type: none"> <li>• +</li> <li>• -</li> <li>• *</li> <li>• /</li> <li>• % (modulo)</li> </ul>
<i>Les 2-en-1</i>
<ul style="list-style-type: none"> <li>• ++ incrément (opérateur unaire postfixé : x++)</li> <li>• -- décrément (opérateur unaire postfixé : x--)</li> <li>• += : ajoute le membre de droite au membre de gauche</li> <li>• -= soustrait membre de droite au membre de gauche</li> <li>• *= multiplie le membre de gauche par le membre de droite</li> <li>• /= divise le membre de gauche par le membre de droite</li> </ul>
<b>Opérateurs logiques :</b>
<ul style="list-style-type: none"> <li>• &amp;&amp; (ET logique)</li> <li>•    (OU logique)</li> <li>• ! (NON logique)</li> <li>• ? : commutateur</li> </ul>
<b>Opérateur de concaténation de chaînes</b>
<ul style="list-style-type: none"> <li>• +</li> </ul>
<b>Opérateurs de comparaison :</b>
<ul style="list-style-type: none"> <li>• == égalité</li> <li>• != inégalité</li> <li>• &lt;</li> <li>• &gt;</li> <li>• &lt;=</li> <li>• &gt;=</li> </ul>
<b>Opérateurs sur digits :</b>
<ul style="list-style-type: none"> <li>• ! : inversion des digits (1 &lt;-&gt; 0)</li> <li>• &amp; : ET entre digits</li> <li>•   : OU entre digits (alt gr 6)</li> <li>• ^ : OU exclusif (XOR) entre digits (alt gr 9)</li> </ul>

## Comment faire pour ....

Créer une classe qui soit un programme	<pre>class nom_de_classe{     ...     public static void main(String[] nom)     {         ...     }     ... }</pre>
Interrompre un programme en cours d'exécution depuis la console	Ctrl + C
Afficher un message sur la console de sortie	System.out.println()
Changer le type d'une variable (ou d'une valeur) de type primitif	(nouveau_type_primitif) nom_de_variable ou (nouveau_type_primitif) valeur
Déclarer une variable de type primitif ou String	type nom_de_variable ( ex: <b>char</b> c, <b>int</b> i, <b>String</b> chaine..)
Déclarer un tableau	type_du_contenu[] nom_du_tableau (ex: <b>int</b> [] tablentier)
Créer un tableau de dimension n	nom_du_tableau = <b>new</b> type_du_contenu[n]
Accéder à la valeur de l'élément d'indice <i>i</i> d'un tableau	nom_du_tableau[i]
Accéder au nombre d'éléments d'un tableau	nom_du_tableau.length
Effectuer une séquence d'instructions si une condition est réalisée.	<pre>if (condition){     .... séquence d'instructions... }</pre>
Effectuer une séquence d'instructions si une condition est réalisée, et une autre séquence sinon.	<pre>if(condition){     ...séquence d'instructions } else{     ....autre séquence d'instructions }</pre>
Sélectionner une séquence d'instruction en fonction de la valeur d'un sélecteur (entier, caractère, chaîne)	<pre>switch (selecteur){ case valeur1 : ... séquence d'instructions... break; case valeur2: ..séquence d'instructions... break; .... default: ...séquence d'instructions ... break; }</pre>

Répéter une séquence d'instructions tant qu'une condition est vérifiée	<pre>while (condition){     ... séquence d'instructions à répéter... } ou bien do {     ...séquence d'instructions à répéter.. } while (condition);</pre>
Répéter une séquence d'instructions <i>n</i> fois.	<pre>for (int i=0;i&lt;n;i=i+1){     ...séquence d'instructions à répéter n     fois... }</pre>
Répéter une séquence d'instructions en utilisant un compteur	<pre>for (état initial;condition;incrémentaion){     ... séquence d'instructions à répéter...}</pre>
Accéder au nombre de caractères composant une chaîne de caractères.	<pre>nom_de_la_chaine.length()</pre>
Accéder au caractère d'indice <i>i</i> dans une chaîne de caractères.	<pre>nom_de_la_chaine.charAt(i)</pre>
Obtenir un nombre aléatoire compris entre 0 et 1.	<pre>Math.random()</pre>
Déclarer une méthode prenant des paramètres et n'ayant pas de valeur de retour	<pre>void nom_de_methode(type_du_parametre1 nom_du_parametre1, type_du_parametre2 nom_du_parametre2, ...) {     ...instructions }</pre>
Déclarer une méthode prenant des paramètres et ayant une valeur de retour.	<pre>type_de_retour nom_de_methode(type_du_parametre1 nom_du_parametre1, type_du_parametre2 nom_du_parametre2,...) {     ...instructions... return valeur_de_retour; }</pre>
Utiliser des méthodes et des constantes de classes contenues dans des bibliothèques non standard.	<pre>nom_de_bibliotheque.nom_de_classe.nom_de_methode( ) ou nom_de_bibliotheque.nom_de_classe.nom_de_variable</pre>
Importer une bibliothèque non standard	<pre>import nom_de_bibliotheque.*;</pre>

## Types primitifs et occupation mémoire :

identifiant	char	byte	short	int	long	float	double	boolean
nature	Caractère ou entier non signé entre 0 et $2^{16}-1$	Entier signé entre -128 et 127	Entier signé entre $-2^{15}$ et $2^{15}-1$	Entier signé entre $-2^{31}$ et $2^{31}-1$	Entier signé entre $-2^{63}$ et $2^{63}-1$	Réel en virgule flottante	Réel en virgule flottante double précision	Valeur logique (vrai ou faux, true ou false)
Taille en octets	2	1	2	4	8	4	8	