

Convention d'écriture

- Le nom des classes (et des constructeurs) commence par une **majuscule** ;
- Le nom des méthodes, des variables ou des instances commence par une **minuscule** ;
- Les mots réservés sont obligatoirement en **minuscules** ;
- Les constantes sont généralement en **majuscules**.

Types primitifs

type java	type de codage	nb bits	min et max	défaut
boolean	true/false	1		false
char	Unicode	16	\u0000 à \uFFFF	\u0000
byte	entier signé	8	-128 à 127	0
short	entier signé	16	-32 768 à 32 767	0
int	entier signé	32	-2 147 483 648 à +2 147 483 647	0
long	entier signé	64	-9 223 372 036 854 775 808 à 9 223 372 036 854 775 807	0
float	flottant IEEE 754	32	+/-1.4e-45 à +/- 3.4028235 e+38	0.0f
double	flottant IEEE 754	64	+/-4.9e-324 à +/-1.7976931348263157 e+308	0.0d

Priorité des opérateurs

Les opérateurs sont classés suivant l'ordre des priorités décroissantes.

Les opérateurs d'une ligne ont la même priorité, tous les opérateurs de même priorité sont évalués de la gauche vers la droite sauf les opérateurs d'affectation.

opérateurs postfixés	[] . expr++ expr--
opérateurs unaires	++expr --expr +expr -expr ~ !
création ou cast	new (type) expr
opérateurs multiplicatifs	* / %
opérateurs additifs	+ -
décalages	<< >> >>>
opérateurs relationnels	< > <= >=
opérateurs d'égalité	== !=
et bit à bit	&
ou exclusif bit à bit	^
ou (inclusif) bit à bit	
et logique	&&
ou logique	
opérateur conditionnel	? :
affectations	= += -= *= /= %= &= ^= = <<= >>= >>>=

Annexe 2

Quelques classes et méthodes

La classe ES (non standard)

```
import outilsLi230.ES;      ou      import outilsLi230.*;
```

La classe ES contient entre autres les méthodes suivantes :

<code>static void ecrire (String s)</code>	Écrit la chaîne s à l'écran
<code>static void ecrireln(String s)</code>	Écrit la chaîne s à l'écran avec passage à la ligne (RC)
<code>static void ecrireln ()</code>	Fait un passage à la ligne (RC)
<code>static int lireEntierln ()</code>	Retourne la valeur d'un "int" lu
<code>static float lireFlottantln ()</code>	Retourne la valeur d'un "float" lu
<code>static double lireDoubleln ()</code>	Retourne la valeur d'un "double" lu
<code>static String lireLigne ()</code>	Retourne la ligne de caractères qui a été lu
<code>static void dormir (int n)</code>	Arrête l'exécution du programme pendant n millisecondes.
<code>static void valider (String s)</code>	Affiche une fenêtre contenant le texte s, la fenêtre se refermant quand on clique sur le bouton "OK".

La documentation Java est en ligne sur le site de l'ARI : <http://www-ari.ufr-info-p6.jussieu.fr/>

Onglet : *Outils*, item : *Documentation*, partie : *Documentation jdk 1.4.2*

La classe Math

Remarque : la classe Math contient 2 attributs de classe qui sont PI et E.

Elle contient entre autres les méthodes statiques suivantes :

<code>static int abs (int i)</code> <i>idem pour float, double, long</i>	Retourne la valeur absolue de l'entier i (respectivement du flottant, du double et de l'entier long)
<code>static int max (int a, int b)</code> <i>idem pour float, double, long</i>	Retourne le plus grand des 2 entiers (respectivement des flottants, doubles et entiers longs)
<code>static int min (int a, int b)</code> <i>idem pour float, double, long</i>	Retourne le plus petit des 2 entiers (respectivement des flottants, doubles et entiers longs)
<code>static double pow (double a, double b)</code>	Retourne la valeur a puissance b
<code>static double random ()</code>	Retourne un nombre aléatoire compris entre 0 et 1 (1 exclus)
<code>static int round (float f)</code>	Retourne la valeur entière la plus proche du flottant f
<code>static long round (double d)</code>	Retourne la valeur entière la plus proche du double d
<code>static double sin (double d)</code> <code>static double cos (double d)</code> <code>static double tan (double d)</code> <i>idem avec asin, acos, atan</i>	Retourne le sinus de l'angle d (respectivement le cosinus et la tangente)
<code>static double sqrt (double d)</code>	Retourne la racine carrée du nombre d

La classe String

La classe String contient entre autres les méthodes suivantes :

<code>int length ()</code>	Retourne la longueur de la chaîne
<code>boolean equals (Object o)</code>	Compare la chaîne à l'objet o
<code>int compareTo (String s)</code>	Retourne 0 si les chaînes sont identiques
<code>String concat (String s)</code>	Concatène la chaîne s à la fin de la chaîne
<code>int indexOf (String s)</code>	Retourne la position dans la chaîne de la première occurrence de la chaîne s (retourne -1 sinon)
<code>int indexOf (String s, int i)</code>	Retourne la position dans la chaîne, à partir de i, de la première occurrence de la chaîne s (retourne -1 sinon)
<code>String replace (char oc, char nc)</code>	Retourne la chaîne dans laquelle toutes les occurrences du caractère oc ont été remplacées par le caractère nc
<code>char charAt (int i)</code>	Retourne le caractère à la position i
<code>String substring (int d, int f)</code>	Retourne la sous-chaîne de caractères commençant à la position d et se terminant à la position f
<code>String toLowerCase ()</code>	Retourne la chaîne convertie en minuscules
<code>String toUpperCase ()</code>	Retourne la chaîne convertie en majuscules
<code>static String valueOf (int i)</code> <code>static String valueOf (long l)</code> <code>static String valueOf (double d)</code> <code>static String valueOf (float f)</code>	Retourne la chaîne de caractères représentant l'entier i (respectivement le long l, le double d, et le flottant f)

La classe Vector

La classe Vector contient entre autres les méthodes suivantes :

<code>void add (int i, Object e)</code>	Insère l'élément e à la position i dans le vecteur
<code>boolean add (Object e)</code>	Insère l'élément e à la fin du vecteur
<code>boolean addAll (Collection c)</code>	Insère tous les éléments de la collection c à la fin du vecteur
<code>boolean addElement (Object e)</code>	Insère l'élément e à la fin du vecteur et incrémente la capacité du vecteur de 1 si nécessaire
<code>boolean contains (Object e)</code>	Teste si l'élément e appartient au vecteur
<code>boolean containsAll (Collection c)</code>	Teste si toute la collection c est contenue dans le vecteur
<code>void clear ()</code>	Retire tous les éléments du vecteur
<code>Object elementAt (int i)</code> <code>Object get (int i)</code>	Retourne l'élément à la position i du vecteur
<code>int indexOf (Object e)</code>	Retourne la position de la première occurrence de l'élément e dans le vecteur (retourne -1 sinon)
<code>int indexOf (Object e, int i)</code>	Retourne la position de la première occurrence de l'élément e dans le vecteur à partir de la position i (retourne -1 sinon)
<code>Object remove (int i)</code>	Retourne et supprime l'élément à la position i du vecteur
<code>void removeElementAt (int i)</code>	Supprime l'élément à la position i du vecteur
<code>void setElementAt (Object e, int i)</code>	Rangé l'élément e à la position i du vecteur
<code>Object set (Object e, int i)</code>	Rangé l'élément e à la position i du vecteur et retourne l'élément précédemment rangé à cette position
<code>int size ()</code>	Retourne le nombre d'éléments actuellement dans le vecteur
<code>int capacity ()</code>	Retourne la capacité courante du vecteur