

## Les types entiers

Type	Taille (octets)	Valeur minimale	Valeur maximale
byte	1	-128 (Byte.MIN_VALUE)	127 (Byte.MAX_VALUE)
short	2	-32 768 (Short.MIN_VALUE)	32 767 (Short.MAX_VALUE)
int	4	-2 147 483 648 (Integer.MIN_VALUE)	2 147 483 647 (Integer.MAX_VALUE)
long	8	-9 223 372 036 854 775 808 (Long.MIN_VALUE)	9 223 372 036 854 775 807 (Long.MAX_VALUE)

23

## Les types flottants

Type	Taille (octets)	Précision (chiffres significatifs)	Valeur absolue minimale	Valeur absolue maximale
float	4	7	-1.40239846E-45 (Float.MIN_VALUE)	3.40282347E38 (Float.MAX_VALUE)
double	8	15	4.9406564584124654E-324 (Double.MIN_VALUE)	1.797693134862316E308 (Double.MAX_VALUE)

24

## Le type caractère

char      2 octets      code universel Unicode

25

## Le type booléen

boolean      valeur logique true ou false

Exemple d'initialisation :      boolean condition ;  
                                         condition = n > p ;

26

## Initialisation de variable

```
int n = 5 ;  
int n ; n = 5 ;  
int duo = 2 * n ;
```

En Java, une variable doit être initialisée avant d'être utilisée  
(sinon erreur de compilation).

27

## Les constantes

- final int N = 20 ;
- final int N = 2 \* p ;
- final int N ;  
N = Clavier.lireInt () ; // Première affectation : OK  
N = N + 1 ; // 2ème affectation : erreur de compilation...

28

## IV. Les opérateurs

29

### Opérateurs arithmétiques

+	addition
-	soustraction
*	multiplication
/	division
%	modulo, reste de la division sur entiers ou flottants

30

### Priorités des opérateurs (en ordre décroissant)

* / %	de gauche à droite →
+ -	de gauche à droite →

Pour changer la priorité : insérer des parenthèses...

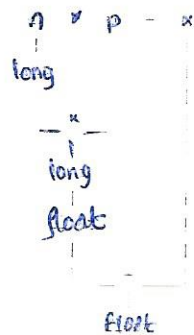
31

- Il y a conversion implicite dans l'évaluation des expressions arithmétiques ;
- Elle tient compte de la priorité des opérateurs ;
- Peut se faire à condition de ne pas dénaturer la valeur initiale :

int → long → float → double

32

Exemple :  $n * p + x;$  (avec n int, p long et x float)



Conversion de n en long  
multiplication par p  
le résultat est de type long  
il est converti en float  
pour être additionné à x  
ce qui fournit un résultat de type float

33

## Opérateurs relationnels

- < inférieur
- <= inférieur ou égal
- > supérieur
- >= supérieur ou égal
- == égal
- != différent

34

## Opérateurs logiques

!	négation
&	et
^	ou exclusif
	ou inclusif
&&	et avec court-circuit
	ou inclusif avec court-circuit

35

## Fonctionnement des opérateurs de court-circuit (&& et ||)

- La 2ème condition n'est évaluée que si c'est indispensable pour évaluer la condition complète ...

36

## Opérateurs d'incrémentation

`++`      pré ou post-incrémentation

`--`      pré ou post-décrémentation

- ✓ Opérateur à gauche de son opérande : on incrémente ou décrément d'abord ;
- ✓ Opérateur à droite de son opérande, on incrémente ou décrément après ;

37

## Exemples

```
a = ++ i ;            //    i = i + 1 ;    a = i ;
```

```
a = i ++ ;           //    a = i ;        i = i + 1 ;
```

```
tab[i++] = Clavier.lireInt() ;
```

```
                     // garnit tab[i] puis incrémente i...
```

```
i ++ ;
```

```
++ i ;            // sont 2 instructions équivalentes...
```

38



## Opérateurs d'affectation élargie

Java permet de condenser les affectations.

### Exemples :

<code>a = a + b ;</code>	devient	<code>a += b ;</code>
<code>n = n * v ;</code>	devient	<code>n *= v ;</code>

39

## L'opérateur de CAST

Si a et b sont de type float, et n de type entier :

`n = a / b ;`

donne une erreur de compilation...

"possible loss of precision - found float - required int"

40

Pour forcer la conversion de l'expression en entier,  
on utilise un opérateur nommé cast.

```
n = (int) (a / b) ;
```

Que pensez-vous de ceci ?

```
n = (int) a / b ;
```

Nouvelle erreur de compilation ...

41

Remarque :

le cast (la conversion) est possible également  
vers les autres types de données numériques...

42

## Exercice

Instructions valides ???

```
int a = 1 , b = 2 ;  
float f = 5.5f ;  
double d = 8.4 ;
```

```
f = a + b ;  
a = b * 2.5 ;  
a = (int) d + f ;  
d = a * f ;  
f = a + (int) d ;
```

3)

## V. Les instructions de contrôle

## Exercice

Instructions valides ???

```
int a = 1 , b = 2 ;  
float f = 5.5f ;  
double d = 8.4 ;
```

```
f = a + b ;           // oui  
a = b * 2.5 ;         // non (expression : double)  
a = (int) d + f ;     // non (expression : float)  
d = a * f ;           // oui  
f = a + (int) d ;     // oui
```

43

## V. Les instructions de contrôle

44

---

## L'alternative simple

### Instruction "if"

```
if (condition)
    { instr1 ; instr2 ; ... }
else
    { instr3 ; instr4 ; ... }
```

## V. Les instructions de contrôle

47

### Les alternatives

#### Instruction "if"

```
if (condition)
    { instr1 ; instr2 ; ... }
else
    { instr3 ; instr4 ; ... }
```

48

## Les alternatives

### Instruction "switch"

```
switch (expression)
{
    case const-1 : instr1 ; instr2 ; ... ; break ;
    case const-2 : instr3 ; instr4 ; ... ; break ;
    ...
    default : instr5 ; instr6 ; ... ;
}
```

49

## Les alternatives

### Exemple

```
switch (mois)
{
    case 1 :      case 2 :      case 3 :
    case 4 :      case 5 :
    case 6 : System.out.println ("Premier semestre") ; break ;
    case 7 : case 8 : System.out.println ("Vacances") ; break ;
    default : System.out.println ("Fin de l'année") ;
}
```

50

## Les répétitives

### Instruction "do ... while"

```
do
{
    série d'instructions ;
}
while (condition) ;
```

On exécute 1 fois la série d'instructions avant de vérifier la condition !!!

51

## Les répétitives

### Instruction "while"

```
while (condition)
{
    série d'instructions ;
}
```

On vérifie la condition avant d'entrer dans la répétitive !!!

52



## Les répétitives

### Instruction "for"

```
for ( initialisations ; condition ; incrémentations )  
{  
    série-d'instructions ;  
}
```

53

## Les répétitives

### Exemple 1

```
int i ;  
for ( i = 1 ; i < 4 ; i++ )  
    System.out.println ( "Passage " + i ) ;  
    System.out.print ( "I vaut " + i ) ;
```

*Que donnera l'affichage ?*

54

## Les répétitives

### Exemple 2

```
int i ;  
for ( i = 6 ; i > 0 ; i -= 2 )  
    { System.out.print ( "Valeur " ) ;  
      System.out.println ( i ) ; }
```

*Que donnera l'affichage ?*

55

## Exercices

Afficher les années bissextiles d'aujourd'hui jusqu'en 2500.

Remarque : une année est bissextile si elle est multiple de 4 mais pas multiple de 100.

Résoudre cet exercice

- 1) avec for ;
- 2) avec while ;
- 3) avec do... while ;

56

## Les répétitives

### Instruction "break"

- Break peut être utilisée dans une répétitive pour sortir prématurément ;
- Exemple : Calculer la somme de maximum 10 nombres entrés au clavier ou s'arrêter avant si on entre 0...

57

## Les répétitives

```
int i, n, som ;
for ( i=1, som=0 ; i <= 10 ; i++ )
{
    n = Clavier.lireInt() ;
    if ( n == 0 ) break ;           // Ce break nous branche
    som += n ;
}
System.out.print( "Somme des " + (i-1) + " nb = " + som ) ; // ici
```

58

## Les répétitives

### Instruction "break" avec étiquette

- Dans le cas de répétitives imbriquées, break fait sortir de la boucle la + interne ;
- Pour sortir de + d'un niveau, on utilisera le break avec étiquette :

59

## Les répétitives

### Instruction "break" avec étiquette

```
boucle : while (...)  
    {  
        for (...)  
        { ...  
            ... break boucle ; // Ce break nous branche  
        }  
        ...  
    }  
    ... // ← ici cette fois...
```

60

## Les répétitives

### Instruction "continue"

continue permet de passer prématurément au tour de boucle suivant ;

61

## Les répétitives

### Exemple

```
int i ;  
for ( i = 1 ; i <= 5 ; i++ )  
{   System.out.println (i) ;  
    if ( i < 4 ) continue ;  
    System.out.println ("Fin " + i) ;  
}  
System.out.println ("Fin de boucle") ;
```

Qu'affichera cette séquence d'instructions ?

62

## Les répétitives

### Solution

1  
2  
3  
4  
Fin 4  
5  
Fin 5  
Fin de boucle

63

## Les répétitives

### Instruction "continue" avec étiquette

- Dans le cas de répétitives imbriquées, continue concerne la boucle la + interne ;
- Pour descendre de + d'un niveau il y a le continue avec étiquette :

64

## Les répétitives

### Instruction "continue" avec étiquette

```
boucle : while (...)  
    {  
        for (...)  
        { ...  
            ... continue boucle ; // Ce continue fait poursuivre  
        }  
        ...  
    } // ← la boucle ici...  
    ...
```

65

Fin de la partie théorique du cours...

### Consigne pour le laboratoire :

La classe *Scanner* sera utilisée régulièrement...

66

Conseil : la stocker dans un dossier au choix et mentionner le chemin du dossier dans une variable d'environnement appelée 'Classpath'...

Marche à suivre :

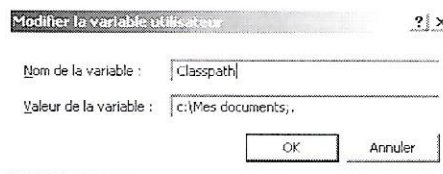
Paramètres / Panneau de configuration / Système / Avancé / Variables d'environnement ...

67

## Démonstration...



Nouvelle variable 'Classpath' avec le chemin du dossier qui contient la classe Clavier et le point qui désigne le répertoire courant...



68