Première approche du langage Java

Une approche traditionnelle d'un langage impératif

Types primitifs

- entier
  - signés seulement
  - type byte (8 bits), short (16 bits), int (32 bits), long (64 bits)
- flottant
  - standard IEEE
  - type float(32 bits), double (64bits)
- booléen
  - type boolean (true,false)
- caractère
  - unicode,
    - type char (16 bits) UTF-16 http://www.unicode.org

Variables de type primitif

- type nom_de_la_variable;
- type nom1,nom2,nom3;
- type nom_de_la_variable = valeur;

- exemples :
  - int i;
  - int j = 0x55AA0000;
  - boolean succes = true;

- l'adresse d'une variable ne peut être déterminée

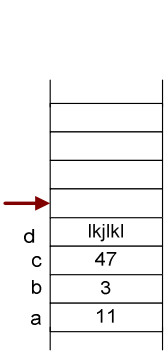
- Pour une méthode, le passage de paramètre est par valeur uniquement (par recopie dans la pile)

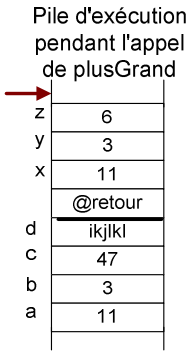
Application Java, un exemple

```
public class Application{
  public static void main( String[] args){
    int i = 5;
    i = i * 2;

    System.out.print(" i est égal à :");
    System.out.println( i);
  }
}
```

Pour une méthode, il y a passage de paramètre par valeur (et par recopie)

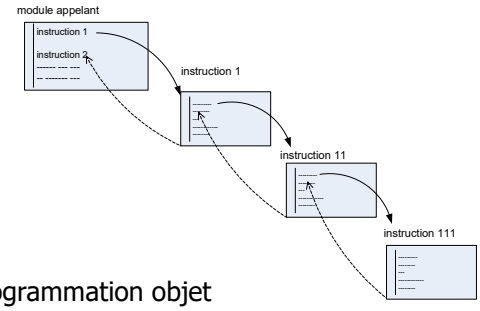
```
class MaClasse{
    // soit, une méthode retournant
    // le plus grand des paramètres
    static int plusGrand(int x, int y, int z)
    {
       ...
    }

    Public static void main( String [] args)
    {
        int a=11, b=3, c=47;
        int d = plusGrand(a,b,6);
        Sysout.println("resultat :" +d);
    }

}
```

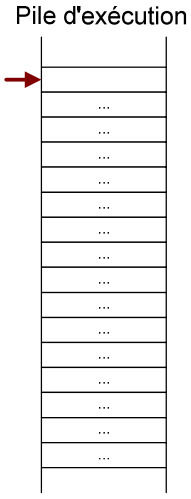




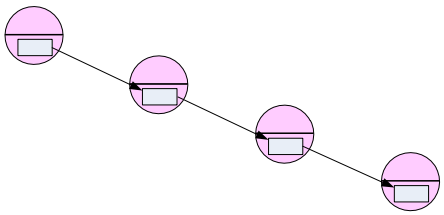
appels successifs et retours successif

- programmation avec méthodes de classe (static)





- programmation objet



Type entier et changement de type

- Automatique
  - si la taille du type destinataire est supérieure
  - byte a,b,c;
  - int d = a+b/c;
- Explicite
  - byte b = (byte)500;
  - b = (byte)(b * 2); // b * 2 promue en int
- par défaut la constante numérique est de type int, suffixe L pour obtenir une constante de type long 40L
- par défaut la constante flottante est de type double, suffixe F pour obtenir une constante de type float 40.0F

Conversions implicites

- Automatique
  - si la taille du type destinataire est supérieure
    - byte -> short,int,long,float,double
    - short -> int, long, float, double
    - char -> short, int, long, float, double
    - int -> long, float, double
    - long -> float, double
    - float -> double

La classe Conversions : Conversions.java

- public class Conversions{
  - public static void main(String args[]){
    - byte b;short s;char c;int i;long l;float f;double d;
    - b=(byte) 0; s = b; i = b; l = b; f = b; d = b;
    - i = s; l = s; f = s; d = s;
    - i = c; l = c; f = c; d = c;
    - l = i; f = i; d = i;
    - f = l; d = l;
    - d = f;
  - }
- }

Type caractère char

- Java utilise le codage Unicode UTF-16
  - représenté par 16 bits
- char c = '\u0020' à '\u007E' code ASCII (7bit)
- \u00AE ©
- \u00BD / (la barre de fraction ...)

- \u0000 à \u1FFF zone alphabets
  - ....
  - \u0370 à \u03FFF alphabet grec
  - .....

- http://www.unicode.org

Opérateurs

- Opérateurs arithmétiques
  - +, -, *, /, %, ++, +=, -=, /=, %=, -- Syntaxe C
- Opérateurs de comparaison (relationnels)
  - ==, !=, >, <, >=, <= Syntaxe C
- Opérateurs logiques (ou booléens)
  - ||, &&
  - &, |, !
- Opérateur ternaire (trois opérandes)
  - ? :
    - MaVariable = expr ? val1 : val2

Opérateurs booléens et court-circuits, exemple

```
1 public class Div0{
2   public static void main( String args[]){
3   int den = 0, num = 1;
4   boolean b;
5
6   System.out.println("den == " + den);
7
8   b = (den != 0 && num / den > 10);
9
10  b = (den != 0 & num / den > 10);
11  }
12}
```

```
Exception in thread "main" java.lang.ArithmeticException :
/ by zero at Div0.main(Div0.java:10)
```

Précédence des opérateurs

+	()	[]	.	
	++	--	~	!
	*	/	%	
	+	-		
	>>	>>>	<<	
	>	>=	<	<=
	==	!=		
	&			
	^			
	|			
	&&			
	||			
	?:			
-	=	op=		

- int a = 1, b = 1, c=2;
- int d = a + b * c; // ??? Que vaut d ???

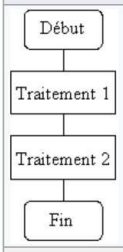
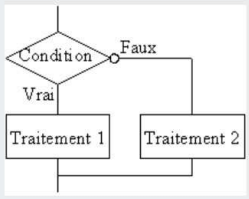
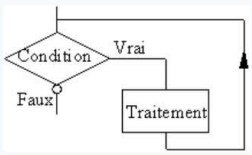
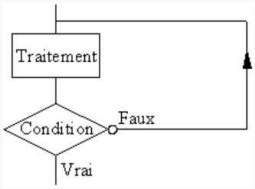
Portée des variables

- Variable locale d'une méthode
- Variable de classe, champs static, attribut static, static field
  - Mot réservé static
  - Variable de portée globale pour toutes les méthodes de la classes

Structure d'un programme

- Pseudo-code, ordinogramme, structures de contrôle
- Source: wikipedia

Les différentes structures de l'organigramme de programmation [modifier | modifier le code]

Séquence linéaire	Séquence alternative « si…alors…sinon »	Séquence répétitive « tant que…faire… »	Séquence répétitive « répéter…jusqu'à… »
			
Début • « Traitement 1 » • « Traitement 2 » Fin	Si « condition » • alors « Traitement 1 » • sinon « Traitement 2 » Fin si	Tant que « condition » • faire « traitement » Fin tant que	Répéter « traitement » jusqu'à « condition »

END