# 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;

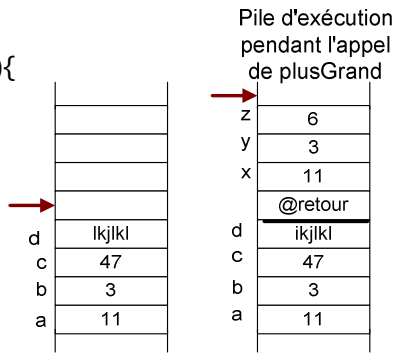
- l'adresse d'une variable ne peut être déterminée

- le passage de paramètre est par valeur uniquement (par recopie dans la pile)



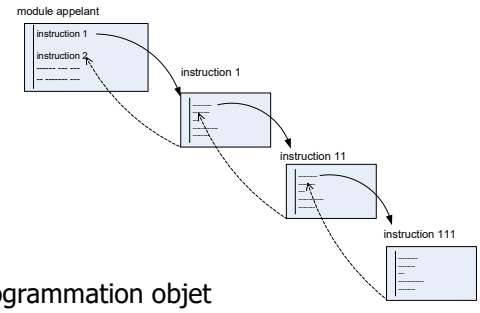
# Pour une méthode, il y a passage de paramètre par valeur (et par recopie)

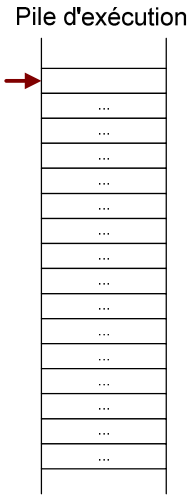
```
class MaClasse{
    // soit, une méthode retournant
    // le plus grand des paramètres
    static int plusGrand(int x, int y, int z){
       ...
    }

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

}
```



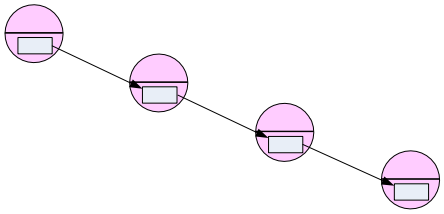
# appels successifs et retours successif

- 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



## 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);
    }
}
```

# 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
- \u0020 à \u007E code ASCII
- \u00AE ©
- \u00BD / (la barre de fraction ...)
- \u0000 à \u1FFF zone alphabets
  - ....
  - \u0370 à \u03FFF alphabet grec
  - .....
- http://www.unicode.org

# Opérateurs

- Arithmétiques
  - +, -, *, /, %, ++, +=, -=, /=, %=, --  Syntaxe C
- Relationnels
  - ==, !=, >, <, >=, <=  Syntaxe C
- Booléens (ou logique)
  - ||, &&
  - &, |, ==, !=, !
- Ternaire
  - ? :
    - MaVariable = expr ? val1 : val2



# 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



# Les tableaux de primitifs dimension 1 et 2

# Le tableau de type primitif

- Déclarations de tableaux
  - int [] mois; // mois est affecté à null ....
  - ou int [] mois = new int[12];
  - ou int [] mois={31,28,31,30,31,30,31,31,30,31,30,31};
  - deux syntaxes autorisées : int mois[] ou int [] mois; la seconde est préférée !
- Déclarations de tableaux de dimension 2
  - double [][] m= new double [4][4];
- Accès aux éléments
  - le premier élément est indexé en 0
  - vérification à l'exécution des bornes, levée d'exception
- Les tableaux sont des objets



# Passage de tableau en paramètres

- rappel: en java, passage de paramètre par valeur uniquement
- tableau en paramètre
  - la variable de type tableau est une référence
  - le passage par valeur de Java ne peut que transmettre la référence sur le tableau
- static void transformer(int [] tab){
  - tab[0] = 8; // --> t[0] == 8;
  - ...
- }
- int[] t = {2,4,3,1,5};
- transformer(t);