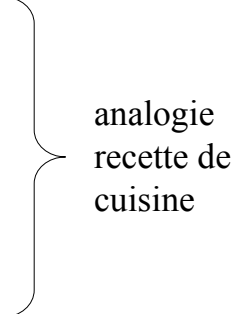


LIFI-Java 2004

Séance du Jeudi 9 sept.

Cours 1

La notion de langage

- Décrire une tâche à effectuer
 - programme
 - Écrire à un haut niveau
 - facile pour l'utilisateur
 - connaissance de la machine utile
 - “Traduire” pour la machine
 - compilation
 - Exécution
- 
- analogie
recette de
cuisine

Le Langage JAVA

- Langage Impératif Orienté Objet
 - manipulation de variables
- Langage Compilé
 - La machine virtuelle, pourquoi?
 - Le *bytecode*, fichier .java et .class
- Langage très répandu!!!
 - <http://java.sun.com>

Mon premier program

- Syntaxe du langage
 - commentaires
 - instructions/blocs d'instructions
 - point-virgule
 - accolades
 - mot-clés
 - valeurs

Mon premier program

- Notion de variables
 - on veut manipuler des valeurs
 - on leur donne des noms “parlants”
 - ça correspond à une “case” en mémoire
- Notion de type
 - que représente la valeur d’une variable?
 - quelles valeurs sont compatibles?

Variables

- Déclaration de variable
 - réserve une case mémoire
 - associe cette case à un nom parlant
- Affectation de variable
 - toute variable est initialisée
 - la valeur peut être changée
- Utilisation de la variable
 - exemple de l’affichage

Types de bases

- int
- float
- char
- String
- boolean
- tableaux

Expressions

- Permet de faire du calcul
- Exemple
- Précédence

Instructions (1/3)

- Comment afficher les nombres de 1 à 100?
 - approche naïve
 - fastidieux
 - pas évolutif!
- Il faut un moyen de faire des “boucles”
 - syntaxe de la boucle `for`
 - utilisation d’une variable “compteur” locale

Instructions (2/3)

- Comment n'afficher que les nombres pairs?
 - astuce mathématique (`println(2*i)`)
- Il faut pouvoir indiquer des “conditions”
 - syntaxe des tests `if..then..else`
 - les opérateurs de test
 - opérateur (`test?instruction1:instruction2`)
- Les boucles `for` utilisent un test!

Instructions (3/3)

- Variante de `for`: boucles `while` et `do..while`
 - montrer l'équivalence `for/while`
 - intérêt de `do..while`
- Variante de `if`, le `switch`
 - écriture simplifiée pour les test successifs
 - attention au `break`!

Exécuter mon programme

- Le fichier “source” porte un nom précis
 - le nom de la classe
 - une classe par fichier
 - par convention, extension `.java`
- Compilation avec `javac`
 - fabrique un fichier `.class`
- Exécution avec `java`

Mon deuxième programme

- Crible d'Eratosthene (276-194 AVJC)
 - But: trouver les nombres premiers
 - Idée: rayer les multiples
- Implémentation en Java
 - Un tableau de 1 à 100 de `boolean`
 - Initialiser à `true`
 - Faire une série de boucles pour mettre à `false` les multiples de 1,2,3,etc...
 - Afficher le tableau



Retour sur les tableaux

- Les tableaux ont une taille fixe
 - indiquées à la compilation (statique)
 - choisie à l'exécution (dynamique)
 - l'opérateur `new`
 - paramétrable par une variable!
- Il existe des tableaux à taille variable
 - redimensionnables dynamiquement
 - classe `Vector`
 - prochain cours!

Mon troisième programme

- Modifier `Erathosten.class` pour prendre la taille du tableau en paramètre.
- Indications:
 - `String args[]` contient la ligne de commande
 - On peut transformer une `String` en `int`:

```
String s = "123";  
int i = Integer.parseInt(s);
```