Initiation à la programmation

Module M1011 - DUT Informatique

Type, variable & fonction

yann.secq@univ-lille1.fr

Abdelghani ATAMENIA, Géry CASIER, Iovka BONEVA, Antoine NONGAILLARD

Lille iut

Université de Lille 1 SCIENCES ET TECHNOLOGIES

Promotion 2016

Menu

- Notion de type
- Notion de variable
- Notion d'affectation
- Notion de fonction
- Squelette d'un programme *iJava*
- Notion de trace d'exécution

Notion de type

- Informatique = Information + (traitement) automatique
- Qu'est-ce qu'une information ?
- Différentes natures d'information:
  - Nombre, texte, image, son …
- Pour l'instant: nombres, caractères et chaînes

Représenter des nombres

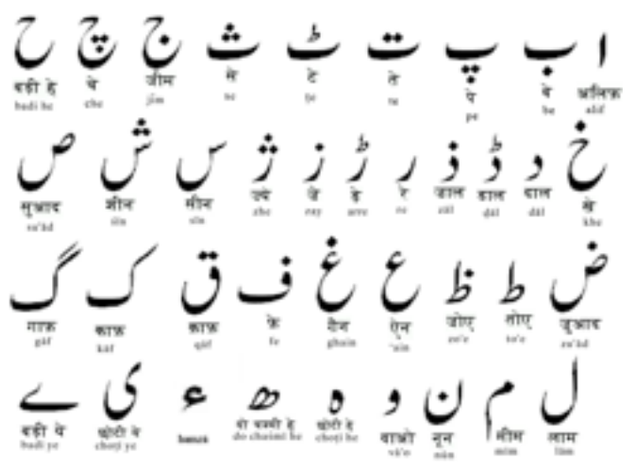
- Nombres entiers
  - `byte`, `short`, `int` **et** `long`
  - exemples: 0, -1, 32, -456 …
- Nombres décimaux
  - `float` **et** `double`
  - exemples: -0.543, 123214.123123, ...
- ATTENTION: `0 != 0.0`
- Opérateurs: `+`, `–`, `*`, `/`, `%`

Expressions arithmétiques

- Expression constituée de nombres et d'opérateurs
- Une expression arithmétique correspond à un nombre lorsqu'elle est évaluée
- Exemple: `42 + (2 / 3) * -17` = ?
- Abusez des parenthèses ...

Expression	Evaluation
3 + 5	
12 / (2+1)	
10 / 6	
10 % 6	

Représenter du texte



Аа Бб Вв Гг Дд Ее
Ёё Жж Зз Ии Йй
Кк Лл Мм Нн Оо
Пп Рр Сс Тт Уу Фф
Хх Цц Чч Шш Щщ
Ъъ Ыы Ьь Ээ Юю
Яя (Іі Ѳѳ Vv Ѣѣ)

夢	風	火	水	心	勇	美
Dream	Wind	Fire	Water	Heart	Courage	Beauty
雨	雪	宝	金	花	星	死
Rain	Snow	Treasure	Gold	Flower	Star	Death
猫	犬	日	月	愛	秋	夏
Cat	Dog	Sun	Moon	Love	Autumn	Summer
考	強	空	朝	夜	冬	春
Think	Strength	Sky	Morning	Night	Winter	Spring

Représenter du texte

- Notion d'alphabet et de mots
- Représentation de caractères et chaînes de caractères
- Caractères
  - `char` (**code** `ASCII`, **cf. codage**)
  - **ex:** `'a'`,`'Y'`,`'9'`,`' '`,`'%'` ...
- Chaîne de caractères
  - `String` (**ensemble de caractères**)
  - **ex:** `"Hello"`,`""`,`"Hamlet: \"Words, words, words\""`

Opérations sur les chaînes

- Sur les caractères:
  - égalité: `==`
  - addition: `+` (!), cf. code `ASCII`
- Sur les chaînes de caractères:
  - égalité: `boolean equals(String s1, String s2)`
  - longueur: `int length(String chaine)`
  - sous-chaîne: `String substring(String chaine, int idxD, int idxF)`
  - concaténation: `"Hello" + "World"` => `"HelloWorld"`

Opérations sur les chaînes

Expression	Evaluation
`equals("titi", "tutu")`	`false`
`equals("tutu", "tutu")`	`true`
`length("Hello")`	`5`
`substring("Bonjour", 0, 3)`	`"Bon"`

Variable et affectation

- Type = nature d'une information
- Variable = emplacement mémoire nommé ne pouvant contenir qu'un type donné d'information
- A sa création, une variable n'est pas initialisée
- Ex: `int age; String nom; char symbole, direction;`
- L'opération d'affectation permet d'initialiser ou modifier le contenu d'une variable

Instruction d'affectation

- L'affectation = stocke une valeur dans une variable
- Syntaxe: `<variable> = <valeur ou expression>;`
  - La partie droite est évaluée et le résultat est stocké dans la variable indiquée à gauche
  - Ex: `int age = 7; double prix = 3.0 * 1.196;`
  - ATTENTION: ne pas confondre = et ==

Notion de constante

- Parfois, on souhaite définir une valeur ne variant pas !
- Une constante est une variable dont la valeur ne peut changer
- **ex:** `final double TVA = 19.6;`
- Préfixer le type par `final` pour définir une constante
- Pour distinguer constantes et variables, les constantes sont définies en MAJUSCULES (convention)

Notion de fonction

- Fonction = séquence d'instructions réalisant une tâche
- Une fonction est caractérisée par ses entrées (paramètres) et par le type du résultat produit (valeur de retour)
- Ex: `int addition(int operande1, int operande2)`
- Ex: `void print(String message)`
- A l'exécution des valeurs sont transmises à la fonction et cette dernière retourne un (seul!) résultat
- Signature = type de retour + nom de la fonction + liste de paramètres (type et nom)

Programme *iJava*

```
class Chute extends Program {
  void testVitesseChute() {
    // masse m lâchée de 10 mètre
    assertEquals(14.007, vitesseChute(10));
  }
  double vitesseChute(int hauteur) {
    final double G = 9.81; // m.s-2
    double vitesse = sqrt(2*G*hauteur);
    return vitesse;
  }
}
```

Trace d'exécution

- Programme = suite d'instructions
- Exécution = évaluation séquentielle d'une suite d'instructions
- Trace = visualisation de l'exécution d'un programme
- A terme, simulation d'exécution "de tête" :)

Trace du programme `Chute`

```
class Chute extends Program {
  void testVitesseChute() {
    // masse m lâchée de 10 mètre
    assertEquals(14.007, vitesseChute(10));
  }
  double vitesseChute(int hauteur) {
    final double G = 9.81; // m.s-2
    double vitesse = sqrt(2*G*hauteur);
    return vitesse;
  }
}
```

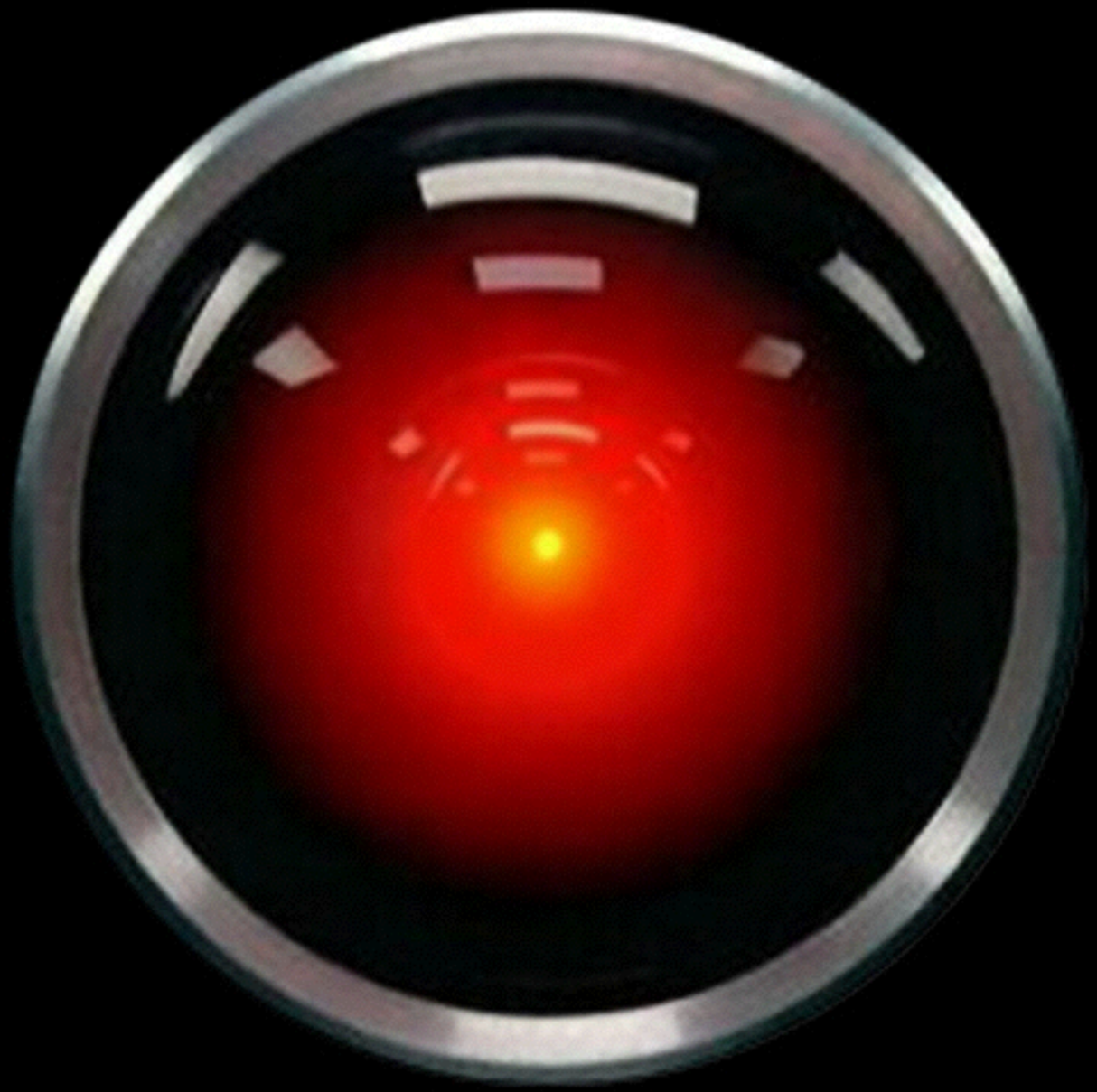
Entrée(s)	G	hauteur	vitesse	Sortie
10				
	9.81			
		10		
			14,007	
				14,007

Synthèse

- Notion de type
- Notion de variable
- Notion d'affectation
- Notion de fonction
- Exemple d'un programme iJava
- Notion de trace d'exécution

MOVIEWALLPAPERS.NET

?



if (<cond>) **{...} else {...}**

```
class Alternative extends Program {
  void algorithm() {
    boolean soleil = ...;
    int argent = ...;
    // supposons soleil et argent initialisés
    println(“début”);
    if (!soleil && argent > 10) {
      prendreMétro();
      argent = cinema(argent);
    } else {
      travaillerAlgo()
    }
    println(“fin”)
  }
}
```