

Introduction à l'informatique

Chapitre 2 - Bases de l'algorithmique

R. Groult, F. Levé

UFR des Sciences,
Université de Picardie Jules Verne, Amiens

mercredi 21 septembre 2011

Structure d'un algorithme

Algorithme

déclaration des variables

début

instructions

.....

fin

Instructions

Les **instructions** précisent les ordres donnés à la machine. Elles sont exécutées séquentiellement : dans l'ordre, l'une après l'autre.

- Dans de nombreux langages de programmation, toute instruction se termine par un point-virgule « ; ».
- En langage algorithmique :
 - il est fortement conseillé de le mettre à chaque fin d'instruction.
 - il est obligatoire de le mettre entre deux instructions situées sur une même ligne.

Variables

- Les variables stockent les données traitées par les instructions.
- Une variable est caractérisée par :
 - son nom
 - son type
 - sa valeur (variable au fur et à mesure des instructions)
- Un programme peut utiliser autant de variables que nécessaire. Elles sont stockées en mémoire à une certaine adresse.

Variables : nom, type

Un **nom de variable** (identificateur) :

- est composé de lettres, de chiffres et « _ »
- commence par une lettre
- est différent d'un mot-clé (mot réservé) du langage

Type de données :

- Détermine les opérations utilisables sur la variable
- Quelques types :
 - entier
 - réel
 - booléen
 - caractère
 - chaîne

Variables : déclaration de variable

Une variable doit être déclarée :

- une fois et une seule
- avant toute utilisation
- Syntaxe :

```
type nom_de_variable ;
```

Il est possible de déclarer plusieurs variables **de même type** sur une même ligne, en séparant les noms de variables par une virgule :

- Syntaxe :

```
type nom_de_variable_1, nom_de_variable_2 ;
```

Variables : affectation d'une variable

- Mécanisme permettant de modifier la valeur d'une variable
- Syntaxe :

`nomVariable ← expression ;`

- Fonctionnement :
 1. évaluation (de la valeur) de l'expression ;
pour cela, les variables sont remplacées par leur valeur
 2. la variable prend la valeur évaluée

Attention!

- Lors d'une déclaration de variable, aucune valeur n'est affectée à la valeur,
- Une variable doit être affectée avant d'être utilisée dans une expression.

Saisie/Affichage

Saisir au clavier et Afficher à l'écran

- lire** Affecte à une variable la valeur lue au clavier, c.à.d. une valeur saisie au clavier par l'utilisateur.
- écrire** Affiche (à l'écran) la valeur d'une variable, d'une constante, ...

Opérateurs

- Similaires aux opérateurs en mathématiques
- Permettent d'effectuer des opérations arithmétiques, booléennes, ou sur des chaînes.

Exemple

+ nombres : addition

+ chaînes : concaténation (ie. chaînes mises bout à bout)

- nombres : soustraction

* nombres : multiplication

/ nombres : division

div entiers : division entière

mod entiers : reste de la division entière (modulo)

==, <, <=, >, >= nombres : comparaison

et, ou, non booléen : « et », « ou », « non » logique

Expression

- Suite d'opérateurs et d'opérandes : variables, constantes (fonctions) décrivant un calcul à effectuer
- Interprétée (évaluée) selon des règles de précedence et d'associativité

Ordre d'évaluation :

MEFIEZ-VOUS ! de nombreux détails diffèrent selon les langages informatiques.

- évaluation des expressions entre parenthèses (et des appels de fonctions)
- priorité des opérateurs
 1. - (unaire)
 2. * / div mod
 3. + -(soustraction)
- priorité des opérateurs booléens : non, et, ou
- évaluation de gauche à droite si même priorité

Trace d'une exécution

Principe

- Méthode permettant de vérifier le fonctionnement d'une séquence d'instructions, d'un programme (ou d'une fonction).
- On écrit dans un tableau la trace temporelle d'une exécution : les instructions exécutées, les valeurs de variables modifiées,

Comment faire :

- Numérotter les lignes
- Faire un tableau
 - Prévoir une colonne pour les instructions
 - Prévoir une colonne pour chacune des variables
- Exécuter pas à pas la séquence d'instructions : pour chaque instruction, inscrire dans les colonnes correspondantes le numéro d'instruction et les modifications de valeurs de variables.

Type d'une expression

- Si l'on affecte à une variable une valeur qui n'est pas du type déclaré, une erreur peut se produire.

Règles de base :

| type1 | opérateur | type2 | résultat |
|--------|---------------|--------|----------|
| entier | + - * div mod | entier | entier |
| réel | + - * / | réel | réel |
| chaîne | + | chaîne | chaîne |

Attention : sous réserve de ne pas diviser par 0

Deux nouveaux opérateurs : div, mod

- **div** : fournit le quotient de la division entière entre deux entiers
- **mod** : fournit le reste de la division entière entre deux entiers

Conversion de type

- Quand nécessaire (calcul, passage de paramètres, ...), une conversion (**transtypage**) du type entier vers le type réel est opérée automatiquement (implicitement).
- De plus, en présence de l'opérateur de concaténation, les conversions suivantes sont opérées :
 - entier vers chaîne
 - réel vers chaîne
 - booléen vers chaîne

Forcer la conversion de réel en entier

- Syntaxe : `entier(expression_de_type_reel)`

Démarche d'écriture (conseils)

1. Pour commencer

- déterminer les **données** du problème à résoudre et leur type
- déterminer le type du **résultat**
- préciser en commentaire le rôle de l'algorithme (le problème résolu)
et le rôle des données
- écrire les « préconditions »
 - Cas général = domaine des valeurs pouvant être prises par les variable pour que le problème ait du sens
 - Cas particulier = restriction du problème par l'énoncé

2. Écrire le corps de l'algorithme de « début » à « fin »

- suite d'instructions résolvant le problème