Inf7212 – Langage de programmation Perl

Vladimir Makarenkov et Alix Boc

UQAM

Automne 2012

Documentation recommandée

- Learning Perl, 5th Edition
  Tom Phoenix, Randal L. Schwartz and B.D. Foy
  Publisher: O'Reilly, 2008

- Cours d'introduction au langage Perl en français (DESS TIMH):
  http://www.med.univ-rennes1.fr/~poulique/cours/perl/perl_html/introperl02.html

- Cours du Perl plus avancé en français (François Dagorn et Olivier Salaun):
  http://perso.univ-rennes1.fr/francois.dagorn/perl

- Les diapos du cours seront disponibles sur mon site web:
  http://www.info2.uqam.ca/~makarenv/INF7212.html

Plan

- Introduction au langage Perl
- Structure d’un programme Perl
- Les variables
- Les tableaux
- Les structures de contrôles
- Les boucles
- Les entrées / sorties
- Les tableaux associatifs
- Exercices pour chacun des concepts vus

Introduction au langage PERL

P.E.R.L. signifie Practical Extraction and Report Language
(langage pratique d'extraction et d'édition)

- Créé en 1986 par Larry Wall, pour gérer un système de « News » entre deux réseaux.

- C'est un langage interprété :
  - pas de compilation
  - moins rapide qu'un programme compilé

- **Portable** : existe sous Unix, Linux, Windows, Mac (Amiga, Atari ...)

- **Gratuit** : disponible sur Internet

- **Simple** : quelques commandes permettent de faire ce que fait un programme de 500 lignes en C.

- **Robuste** : pas d'allocation de la mémoire à manipuler, chaînes, piles, pas de limite pour la taille des variables et des tableaux...

Exemples d'un programme Perl

```
1 #!/bin/perl
2
3 print "Bonjour le monde\n";
```

Commentaire obligatoire : indique l'interpréteur Perl

Commande print : afficher à l'écran

```
1   #!/bin/perl
2
3   open(FL,'< monFichier') || die "Problème d'ouverture : $! \n";
4
5   my $i = 0;
6
7   while (my $ligne = <FL>) {
8       $i++;
9    }
10
11  close FL;
12
13  print "Nombre de lignes : $i\n";
```

Ouverture d'un fichier en lecture

Détection d'erreur

Déclaration et initialisation du compteur

Pour chaque ligne lue….

Incrémentation du compteur

Fermeture du fichier

Affichage du contenu du compteur

Exécution d'un programme Perl

Les programmes en Perl ne sont pas compilés, ils sont simplement interprétés par l'interpréteur Perl.

L'usage veut que l'on nomme un programme Perl en utilisant l'extension **.pl**

Exemple :

Soit le fichier : bonjour.pl

```
1 #!/bin/perl
2
3 print "bonjour le monde\n";
```

Il est exécuté en faisant :

>perl bonjour.pl

```
PBG4:~/inf7212_cours09 dcarrey$ perl bonjour.pl
bonjour le monde
PBG4:~/inf7212_cours09 dcarrey$
```

Programme « Hello, world! »

Exemple 1

```
#!/usr/bin/perl
print "Hello, world!\n"; # affiche Hello, world!
```

Exemple 2

```
#!/usr/bin/perl
print # C’est un commentaire
"Hello, world!\n"
; # N’écrivez pas votre code Perl de cette façon
```

Exécution du programme

```
$ perl my_program.pl
```

#Sous Unix et Linux, on ajoute:

```
$ chmod a+x my_program.pl
```

ou

```
$ chmod 755 my_program.pl
```

Les types de données en Perl

1. **Les constantes**

5, -12345, 0.1415, 1.6E16, ‘cerise’, ‘a’, ‘les fruits du palmier sont les dattes’

1. **Les variables scalaires**

Les variables sont précédées du caractère $.

$i = 0; $c = ‘a’; $mon_fruit_prefere = ‘kiwi’;

$racine_carree_de_2 = 1.41421;

$chaine = ‘100 grammes de $mon_fruit_prefere’; # 100 grammes de $mon_fruit_prefere

$chaine = "100 grammes de $mon_fruit_prefere"; # 100 grammes de kiwi

Attention: Ne mettez pas d’espaces dans les noms des variables.

Un nom peut être aussi long qu’on le veut. Dans les dernières versions de Perl,

les noms de variables peuvent être accentués.

Constantes numériques

Exemples

```
1.25
255.000
255.0

7.25e45         # 7.25 fois 10 puissance 45 (un grand nombre)
-6.5e24         # -6.5 fois 10 puissance 24 (un grand nombre négatif)
-12e-24         # -12 fois 10 puissance -24 (un très petit nombre négatif)
-1.2E-23        # une autre façon de représenter E (lettre majuscule)

61298040283768
61_298_040_283_768

0377            # 377 octal, la même chose que 255 décimal
0xff            # FF hex, la même chose que 255 décimal
0b11111111      # aussi, la même chose que 255 décimal
```

Chaînes de caractères et opérations associées

Exemples

```
'barney‘          # le mot barney
' '               # la chaîne nulle (pas de caractères)
"barney"          # la même chose que 'barney'
"hello world\n"   # hello world, et une nouvelle ligne
'\'\\'            # guillemet simple suivi du backslash
```

Exemples d’opérations associées à des chaîne de caractères

```
"hello" . ' ' . "world"      # la même chose que 'hello world'

"fred" x 3                   # c’est "fredfredfred"
"barney" x (4+1)             # "barney" x 5 ou "barneybarneybarneybarneybarney"
 5 x 4                       # est en réalité "5" x 4, ce qui est "5555"

"Z" . 5 * 7                  # la même chose que "Z" . 35 ou "Z35"
```

La saisie au clavier

Perl offre des moyens relativement simples à utiliser pour communiquer avec la console. Nous avons déjà vu comment écrire avec la console.

Écriture :

```
print "mon message s'affiche dans la console";
```

La lecture s'effectue en appelant les symboles suivants : <>

Lecture :

```
$monNom = <>;
```

Pour être plus précis, on lit dans le STDIN (l'entrée standard) et on écrit dans le STDOUT (la sortie standard).

Écriture : print STDOUT "mon message s'affiche à la console";

Lecture : $monNom = <STDIN> ;

La saisie au clavier (2)

$madonna = <STDIN>; # on lit la ligne courante sur l'entrée standard

@madonna = <STDIN>; # on lit toutes les lignes restantes sur l'entrée standard

La lecture peut se faire depuis un fichier, par exemple.

La commande Ctrl-Z indique la fin du texte saisi au clavier sous DOS et Windows.

Chaque ligne est retournée comme un élément distinct du tableau.

Exemple:

```
@lines = <STDIN>;          # lire toutes les lignes sur STDIN
chomp(@lines);             # éliminer les sauts de lignes
```

ou tout simplement:

```
chomp(@lines = <STDIN>); # Lire les lignes sans les sauts de lignes
```

Exercices (1)

1. Écrire un programme qui calcule la circonférence d'un cercle avec le rayon de 12.5. La circonférence est 2π fois le rayon de la circonférence (la valeur de π est approximativement 3.141592654).

2. Modifiez le programme précédent pour permettre à l'utilisateur du programme de saisir la valeur du rayon.

3. Modifiez le programme précédent en sorte que si l'utilisateur saisit le nombre inférieur à 0, la valeur reportée sera 0 (plutôt que négative).

4. Écrire un programme qui lit les valeurs de 2 nombres et affiche à l'écran leur produit.

5. Écrire un programme qui lit une chaîne de caractères et un nombre et affiche, sur les lignes séparées, la chaine en question le nombre de fois indiqué par le nombre saisi. Si l'utilisateur saisit "Fred" and "3", la sortie sera donc:
   Fred
   Fred
   Fred

Coder ces programmes à la démo !!