

CS100: Introduction to Computer Science

Lecture 16: Programming Language: Perl (III) - flow control

Review: A Simple Program (example1.pl)

```
#!/c:/Perl/bin/perl  
print "Hello Perl!\n\n";  
  
print "This is my first program in PERL!\n\n";  
  
# the follow two lines are used to keep the output of the  
# program stays.  
print "\n\n Press ENTER To Exit !\n";  
$tmpvar = <STDIN>;
```

Review: Simple Perl Commands:

- n **print**
- n **printf**
- n **<STDIN>**
- n **chop**
- n **chomp**

Review: Types of Variables

- n Scalar variables
- n Arrays
- n Hashes

Why Computers?

- n Earlier examples:
 - q Statements are executed in order from top to bottom
 - q Every statement is executed only once.
- n Perform repetitive tasks
 - q A set of instructions can be executed over and over again
- n Make quick decisions
 - q Execute one set of instruction if certain condition is true.

Statements and Blocks

- n **Statements:** basic units of execution in Perl. A statement represents one instruction.
 - q Terminates with a semicolon
 - q Can span multiple lines
 - q `$total_annual_sales = $first_quarter_total + $second_quarter_total + $third_quarter_total + $fourth_quarter_total;`
- n **Blocks:** A block is a set of statements enclosed in curly braces.

Program Control

n Conditional Execution

- q Execute one set of instructions if a certain condition is true, another set of instructions otherwise.
- q **if** statement in Perl

n Loops

- q Repeat a block of code either a fixed number of times or until a certain condition is met.
- q The **for** loop and the **while** loop in Perl.

If statements

```
$abs_x = $x;  
if ($abs_x < 0)  
{  
    $abs_x = $abs_x *(-1);  
}  
print ("The absolute value of x is $abs_x \n");
```

if-else statements

```
if ($x > 0)  
{  
    print ("x is positive\n");  
}  
else  
{  
    print ("x is either zero or negative\n");  
}
```

Comparison Operators:

Comparison	Numeric	String
Equal	==	eq
Not Equal	!=	ne
Less than	<	lt
Greater than	>	gt
Less or equal to	<=	le
Greater or equal to	>=	ge

Compare String Values

n Referring to their ASCII values

- q The capital letters are technically less than the lowercase letters
- q The higher the letter is in the alphabet, the higher the ASCII value.

Character	Dec.	Hex.
A	65	41
B	66	42
...		
Z	90	5A
a	97	61
...		
z	122	7A

Exercise 1: Write a program to output the maximum given three numbers.

A while loop

```
$x = 0;
while ($x < 5)
{
    print ("The value of x is $x\n");
    $x++;
}
```

```
The value of x is 0
The value of x is 1
The value of x is 2
The value of x is 3
The value of x is 4
```

A for loop

```
for ($x = 0; $x < 5; $x++)
{
    print ("The value of x is $x\n");
}
```

- n The execution of a for loop
 - q 1. The first part of the for statement ($x = 0$) is executed immediately when the for statement is reached.
 - q 2. Next the second part ($x < 5$) is tested; if the condition is not met, then the rest of the for statement is skipped.
 - q 3. If the condition in the second part is true, then the conditional code is executed.
 - q 4. The last part ($x++$) is executed.
 - q 5. Go to step 2 and continue.

Exercise 2: Display the following graph on the screen (learn to use the for loop or while loop.)

```
#####
#####
#####
#####
#####
```

The foreach loop

```
foreach $value (7, 3, -3, 5, 2)
{
    print ("The value of x is $value\n");
}
```

- n The loop will execute five times, once for each value in the list.
- n For each iteration, the next value in the list will be assigned to the variable.

Array & the for loop

```
for ($i=0; $i < 1000; $i++)
{
    print ("The next number is $numbers[$i]\n");
}
```

Exercise 3: Write a program that prompts the user for 10 numbers and then prints their average.

Code for Exercise 3.

```
#prompt the user for all 10 numbers and store them in the array numbers
for ($i=0; $i <10; $i++)
{
    print ("Enter a number ($i) :\n");
    $numbers[$i] = <STDIN>;
}

#sum up all values in the array numbers
$sum = 0;
for ($i=0; $i <10; $i++)
{
    $sum += $numbers[$i];
}

#Divide the sum by 10 and get the average, then print it
$average = $sum / 10.0;
print ("The average of all the 10 numbers is $average\n");
```

Readability

- n Long and complex programs consisting thousands of lines.
- n Share your codes with others
 - q A group of peoples work on the same project
 - q Modify codes written by other programmers.

Good Habits for Producing Easily Readable Codes

- n Structured Programming
 - q Appropriate use of newlines
 - q Consistently indenting of code blocks, each nested block can be indented further than the previous block
- n Comments
 - q Notes in your code
 - q Initiating with the # character
- n Descriptive variable names

```
#####
# Factorial Program #
# John C. Programmer; July 2001 #
# This program prints the integers from 1 to 10 #
# and the factorial of each to the screen #
#####

# for each number from 1 to 10
for ($number = 1; $number <=10; $number++)
{
    #print the number to the screen:
    print ("Number : $number!\n");

    #initialize the factorial to 1
    $factorial = 1;
    #compute the factorial by multiplying with 2, 3, .. $number
    for ($i = 2; $i <= $number; $i++)
    {
        $factorial *= $i;
    }

    #now the factorial is computed, print it to the screen and go on to
    #the next number.
    print ("Factorial: $factorial!\n");
}
#####
```

Announcements:

- n Lab 4 this week
- n Next lecture: Wednesday, April 11th