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\nPress ENTER To Exit \n\n";
$tmplvar = <STDIN>;
```

Review: Simple Perl Commands:
- print
- printf
- `<STDIN>`
- chop
- chomp

Review: Types of Variables
- Scalar variables
- Arrays
- Hashes

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

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

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

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

If statements

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

Comparison Operators:

<table>
<thead>
<tr>
<th>Comparison</th>
<th>Numeric</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal</td>
<td>==</td>
<td>eq</td>
</tr>
<tr>
<td>Not Equal</td>
<td>!=</td>
<td>ne</td>
</tr>
<tr>
<td>Less than</td>
<td>&lt;</td>
<td>lt</td>
</tr>
<tr>
<td>Greater than</td>
<td>&gt;</td>
<td>gt</td>
</tr>
<tr>
<td>Less or equal to</td>
<td>&lt;=</td>
<td>le</td>
</tr>
<tr>
<td>Greater or equal to</td>
<td>&gt;=</td>
<td>ge</td>
</tr>
</tbody>
</table>

if-else statements

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

Compare String Values

- Referring to their ASCII values
  - The capital letters are technically less than the lowercase letters
  - The higher the letter is in the alphabet, the higher the ASCII value.

<table>
<thead>
<tr>
<th>Character</th>
<th>Dec.</th>
<th>Hex.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>65</td>
<td>41</td>
</tr>
<tr>
<td>B</td>
<td>66</td>
<td>42</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>90</td>
<td>5A</td>
</tr>
<tr>
<td>a</td>
<td>97</td>
<td>61</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>z</td>
<td>122</td>
<td>7A</td>
</tr>
</tbody>
</table>

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

```php
$x = 0;
while ($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

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

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

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

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

Array & the for loop

```php
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.

```php
# 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

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

---

Good Habits for Producing Easily Readable Codes

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

---

Announcements:

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