**PERL TUTORIAL**

Perl is a general-purpose programming language originally developed for text manipulation and now used for a wide range of tasks including system administration, web development, network programming, GUI development, CGI and more.

**What is PERL?**

* Perl is a stable, cross platform programming language.
* Perl stands for **Practical Extraction and Report Language**.
* Perl was created by Larry Wall.
* Perl is a programming language which can be used for a large variety of tasks. A typical simple use of Perl would be for extracting information from a text file and printing out a report or for converting a text file into another form.

**PERL Features**

* Perl takes the best features from other languages, such as C, awk, sed, sh, and BASIC, among others.
* Perls database integration interface ([DBI](http://dbi.perl.org/)) supports third-party databases including Oracle, Sybase,  [MySQL](http://www.mysql.com/) and others.
* Perl works with HTML, XML, and other mark-up languages.
* Perl supports both procedural and object-oriented programming.
* The Perl interpreter can be embedded into other systems.

## Is Perl Compiled or Interpreted?

Perl is implemented as an interpreted (not compiled) language. Traditional compilers convert programs into machine language. When you run a Perl program, it's first compiled into a byte code, which is then converted ( as the program runs) into machine instructions. So it is not quite the same as shells, which are "strictly" interpreted without an intermediate representation. Nor it is like most versions of C or C++, which are compiled directly into a machine dependent format.

# PERL Syntax Overview

* 1. **Perl statements end in a semi-colon:** print "Hello, world";
  2. **Comment Statement:** # This is a comment
  3. **White Space is irrelevant:** print "Hello, world";
  4. Double quotes or single quotes may be used around literal strings:

print "Hello, world";

print 'Hello, world';

**BUT THE IMPORTANT PONT IS THAT** - A String in-between Single quotes **( ‘ ‘ )** has value exactly the sequence of characters. In case of **( “ “ )** Substitution is occurred.

**Example:**

$i=10;

$s1=’winter for last $i months’;

$s2=”winter for last $i months”;

print $i;

print $s1;

print $s2;

**Output:**

10

winter for last $i months

winter for last 10 months

**Chomp () function in PERL:**

The chomp() function will remove (usually) any newline character from the end of a string. When reading user input from the standard input stream (STDIN) for instance, you get a newline character with each line of data. chomp() is really useful in this case because you do not need to write a regular expression and you do not need to worry about it removing needed characters.

**Normal Syntax:**

print "How old are you?";

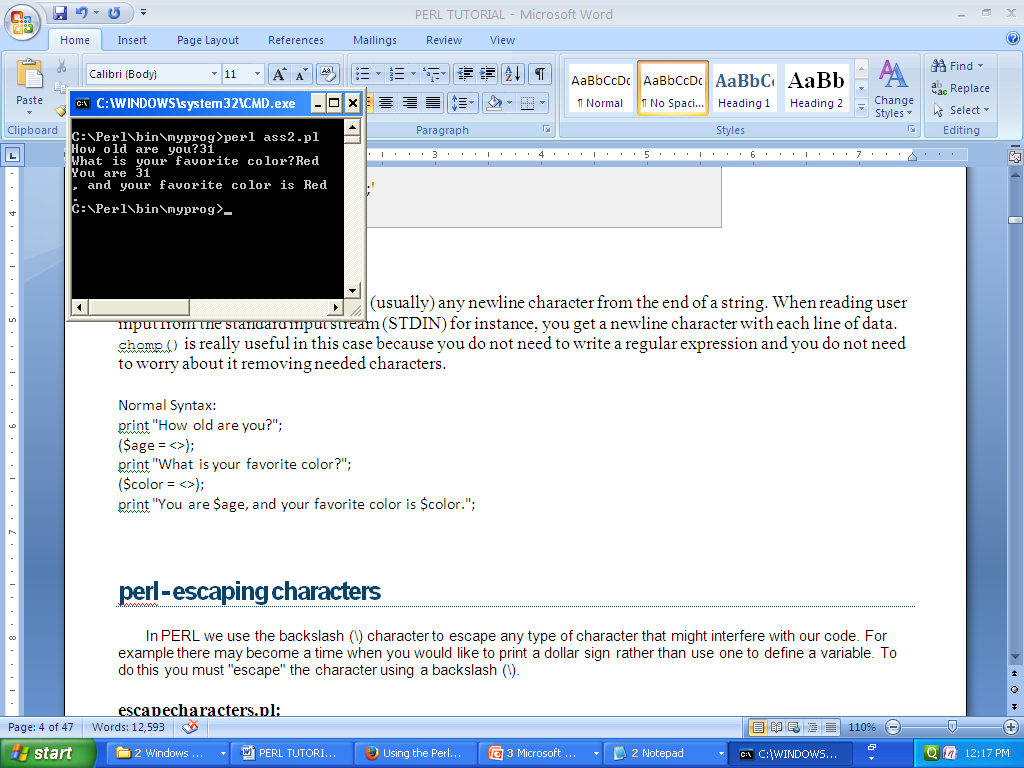
($age = <>);

print "What is your favorite color?";

($color = <>);

print "You are $age, and your favorite color is $color.";

**Output:**



**Using Chomp():**

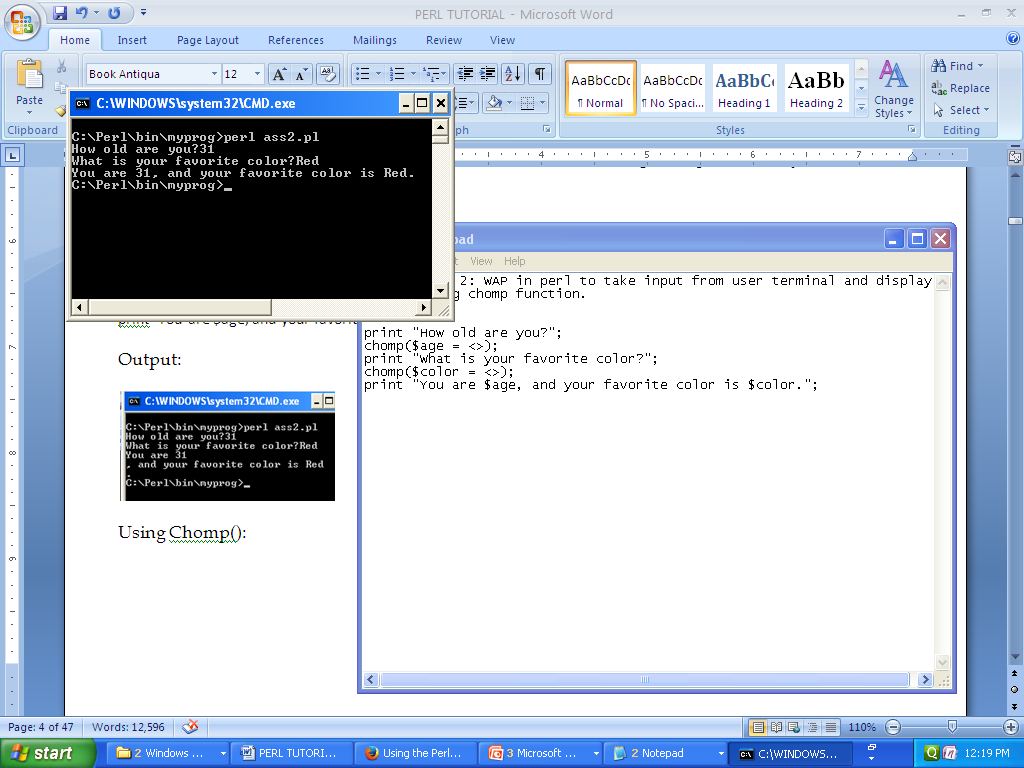
print "How old are you?";

chomp($age = <>);

print "What is your favorite color?";

chomp($color = <>);

print "You are $age, and your favorite color is $color.";



# PERL Variable Types

Perl has three built in variable types:

* **Scalar ( $ )**
* **Array ( @ )**
* **Hash ( % )**

**scalar =>** {

description => "single item",

sigil => '$',

},

**array =>** {

description => "ordered list of items",

sigil => '@',

},

**hash =>** {

description => "key/value pairs",

sigil => '%',

},

**Scalar Variables:**

A scaler variable is represented by doller sign ($).

A scalar represents a single value as follows:

|  |
| --- |
| my $animal = "camel"; my $answer = 42; |

Here **my** is a keyword.

A scalar values can be strings, integers or floating point numbers, and Perl will automatically convert between them as required. There is no need to pre-declare your variable types. Scalar values can be used in various ways:

|  |
| --- |
| $age = 25; 🡪integer  $name = “Anupam” 🡪String  $Salary = 1445.50 🡪 Floating |

**Array Variables:**

An array is a variable that stors an ordered list of Scalar variables. It is represented through “@” Symbol.

@ages = (25,30,40);

@name = (“Ram”, “Hari”, “Madhu”);

print “ages[0] = $ages[0]”; print “ages[1] = $ages[1]”; print “ages[2] = $ages[2]”;

**Output:** ages[0]=25 ages[1]=30 ages[2]=40

**Hash Variables:**

Hash variables are represented through “%” symbol.

A hash is a set of key/value pairs. To refer a single element of a hash, you will use the hash variable name followed by the “key” associated with the value in brackets.

%data = (‘John’,45,’Lisa’,30,’Kumar’,40);

Print “\data{‘John’}=$data{‘John’}\n”;

Print “\data{‘Lisa’}=$data{‘Lisa’}\n”;

Print “\data{‘Kumar’}=$data{‘Kumar’}\n”;

**SOME BASIC PARL PROGRAM:**

# Assignment 1: Write a perl script to take input from the user

#such as name,Roll,Department,Stream and diplay it with proper syntax.

NOTE: (#) it is **Comment statement**.

* Perl statements end in a **semi-colon**(;)

print"hello\n world";

* **variable declearation**

$name=anupam;

* Variable print

**print"\n$name";**

* **<STDIN>** stands for standard input.
* It can be abbreviated by using simple <>.

**print "\nHow old are you?";**

**$age = <>;**

**print "WOW! You are $age years old!";**

**#Assignment : WAP in perl to take input from user terminal and display it by using chomp function.**

print "How old are you?";

chomp ($age = <>);

print "What is your favorite color?";

chomp ($color = <>);

print "You are $age, and your favorite color is $color.";