SIMRAN WATWANI   D7B-70   BATCH-C

Experiment 11

Aim: To implement Perl basics

Theory:

Perl is a programming language developed by Larry Wall, especially designed for text processing. It stands for Practical Extraction and Report Language. It runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. Perl is an interpreted language, which means that your code can be run as is, without a compilation stage that creates a non portable executable program.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, or Tcl, which are strictly interpreted without an intermediate representation.
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, and more.

Features of Perl:

1. Variables and their Related Operators

Perl provides three kinds of variables: scalars, arrays, and associative arrays. The discussion includes the designation of these three types and the basic operators provided by Perl for their manipulation.

2. Control Structures

Perl is an iterative language in which control flows from the first statement in the program to the last statement unless something interrupts. Some of the things that can interrupt this linear flow are conditional branches and loop structures. Perl offers approximately a dozen such constructs. Each of these basic constructs are described along with examples illustrating their use.

3. Functions

Functions are a fundamental part of most programming languages. They often behave like an operator, producing a change in the value of some variable or returning a value that can be assigned to a variable. They also control the flow of execution, transferring control from the point of invocation to the function definition block and back. Thus, they combine properties of the two preceding discussions. The discussion will cover both the designation of functions and their invocation and use.

4. Regular Expressions and Related Operators

Regular expressions are strings that can be recognized by a regular grammar, a restricted type of context-free grammar. Basically, they are strings that can be parsed left to right, without backtracking, and requiring only exact symbol matching, matching of a symbol by a category of symbols, or matching of a symbol by a specified number of sequential occurrences of a symbol or category.

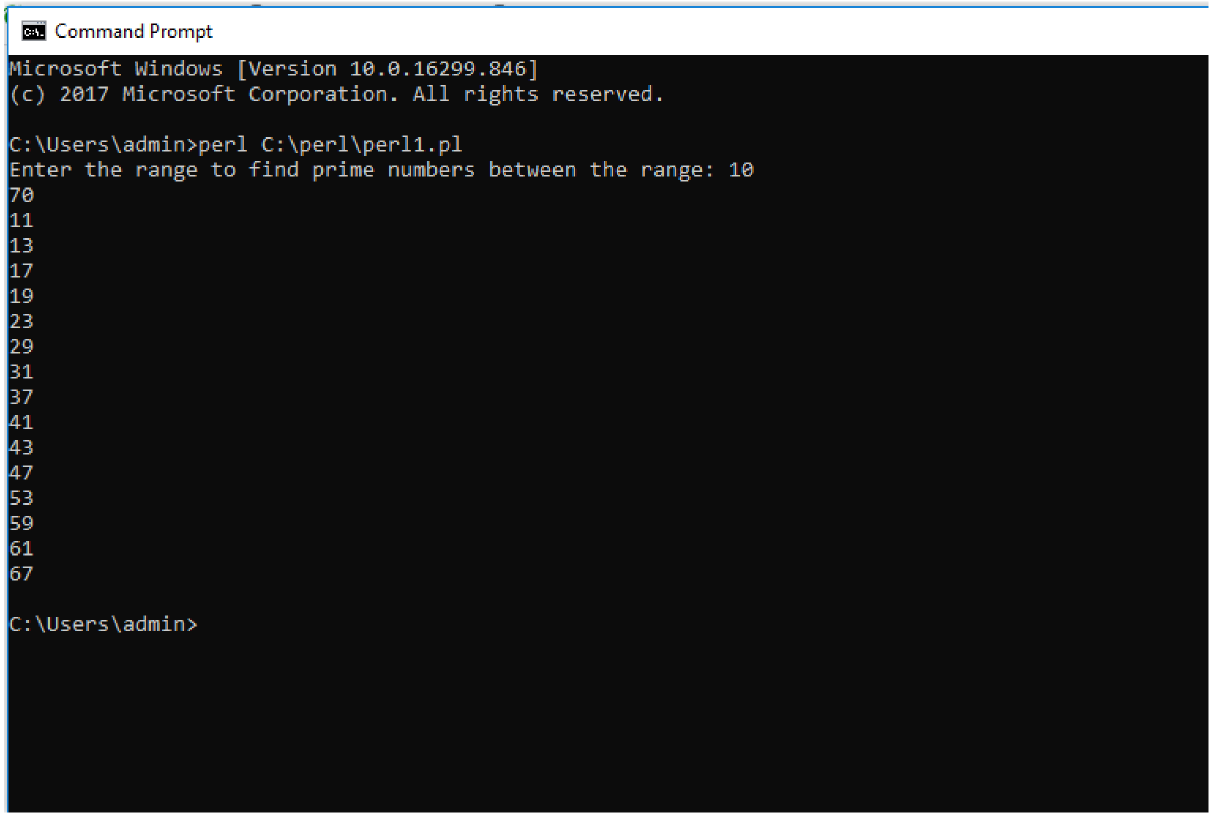
Perl provides a general mechanism for specifying regular expressions. It also provides several operators that manipulate strings based upon the evaluation of a regular expression.
The discussion will begin by describing the various mechanism for specifying patterns and then discuss expression-based operators.
5. Input/Output
Perl provides basic I/O for both the standard input (keyboard) and output (display) devices and for files in the UNIX file system. More sophisticated I/O is provided through the UNIX DBM library. These various I/O capabilities are discussed.
6. System Operators
Perl offers a number of operators that mimic or call UNIX system operators or analogous operators for other operating systems. The discussion here will be cast in the context of UNIX and will assume familiarity with basic UNIX facilities.
Perl system operators can be roughly divided into two large categories: file/ directory operators and process operators. Both types are discussed.

Perl File naming Convention:
Perl file name can contain numbers, symbols, letters and underscore (_), however spaces are not allowed in the file name. for e.g. hello_world.pl is a valid file name but hello world.pl is an invalid file name.
Perl file can be saved with .pl or .PL extension.

Conclusion:
Thus, we have successfully completed and executed prime number and factorial programs in
Perl script and learned the basics of Perl programming.
The errors encountered during execution are:
1) print(“The number is”,$n);
Solution: print“The number is $n”;
2) Missing $ sign: The program runs in an infinite loop when $ sign is not added before a variable Solution: $age

CODE :
1.PRIME SERIES
```
print "Enter the range to find prime numbers between the range: ";
$s = <>;
$e = <>;
for($i=$s; $i<=$e; $i++){
$prime=0;
for($j=2; $j<=$i-1; $j++){
```

```
if($i % $j==0){
$prime=1;
break;
}
}
if ($prime == 0){
print "$i";
print "\n"; } }
```

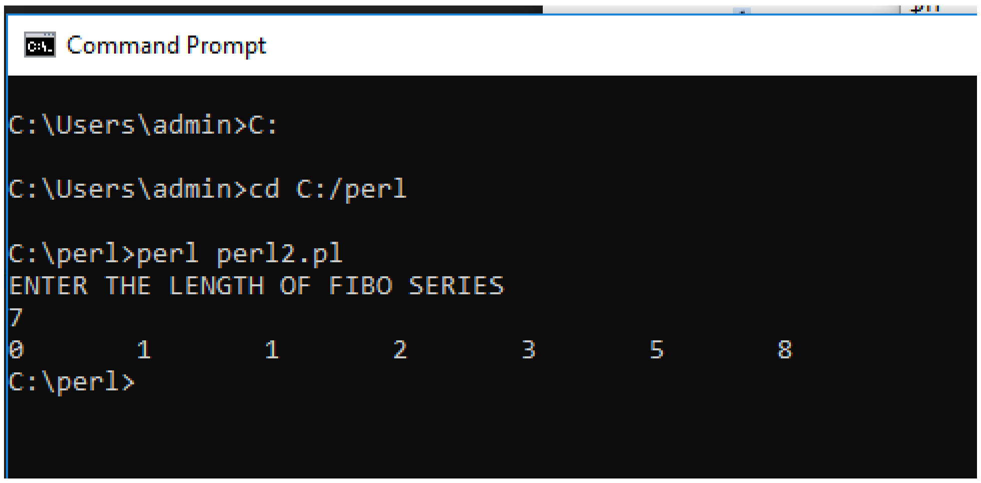


```
Microsoft Windows [Version 10.0.16299.846]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\admin>perl C:\perl\perl1.pl
Enter the range to find prime numbers between the range: 10
70
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67

C:\Users\admin>
```

2.FIBONACCI SERIES :

```
$count=0;
print "ENTER THE LENGTH OF FIBO SERIES\n";
$n= <>;

$a=0;
$b=1;
if($n==1)
{
print"0";
}
if($n==2)
{
```

```
print"0\t1";
}
if($n>2)
{
print"0\t1\t";
for($i=3;$i<=$n;$i++)
{
$c=$a+$b;
print"$c\t";
$a=$b;
$b=$c; } }
```



Command Prompt

```
C:\Users\admin>C:

C:\Users\admin>cd C:/perl

C:\perl>perl perl2.pl
ENTER THE LENGTH OF FIBO SERIES
7
0       1       1       2       3       5       8
C:\perl>
```