**Aim:**

To execute basic Perl programs

**Description:**

Perl is a family of two high-level, general-purpose, interpreted, dynamic programming languages. "Perl" refers to Perl 5, but from 2000 to 2019 it also referred to its redesigned "sister language", Perl 6, before the latter's name was officially changed to Raku in October 2019.

A Perl script can be created inside of any normal simple-text editor program. There are several programs available for every type of platform. There are many programs designed for programmers available for download on the web.

Perl uses the backslash (\) character to escape any type of character that might interfere with our code. Let's take one example where we want to print double quote

**QUESTION 1: Perform arithmetic operations using perl**

**PROCEDURE:**

1. Get two number inputs
2. Perform addition operation and display
3. Perform subtraction operation and display
4. Perform multiplication operation and display
5. Perform division operation and display
6. Perform modulus operation and display

**Output:**

**print "enter 2 numbers\n";**

**$a = <STDIN>;**

**$b = <STDIN>;**

**$c = $a + $b;**

**print 'Value of $a + $b = ' . $c . "\n";**

**$c = $a - $b;**

**print 'Value of $a - $b = ' . $c . "\n";**

**$c = $a \* $b;**

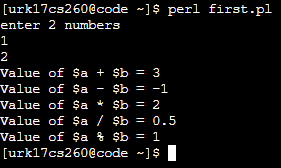
**print 'Value of $a \* $b = ' . $c . "\n";**

**$c = $a / $b;**

**print 'Value of $a / $b = ' . $c . "\n";**

**$c = $a % $b;**

**print 'Value of $a % $b = ' . $c. "\n";**



**QUESTION 2: Demonstrate all the escape sequences using print and say statement.**

**PROCEDURE:**

1. Print text on new line
2. Print text with @
3. Print text with $
4. Print text with double quote
5. Print text with qq
6. Print text with q

**Output:**

1. \n newline
2. Include @
3. $ sign
4. Double quotes “”
5. Double qq

We can replace the double quotes that we use to enclose a string with the double q operator.

The advantage of doing this is that we need not to worry about using escape sequences for double quotes(“) and brackets.

See I have not used the escape sequences for the double quotes and brackets.

1. Single q operator – q

Single q operator works like single quotes. The special characters present inside it does not interpolate.

print("Displaying text \n in new line by using \\n. \n");

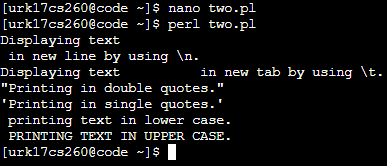
print("Displaying text \t in new tab by using \\t. \n");

print("\"Printing in double quotes.\"\n");

print("\'Printing in single quotes.\'\n");

print("\L printing text in lower case.\n");

print("\U PRINTING TEXT IN UPPER CASE.\n");



**QUESTION 3: Online shopping application**

**PROCEDURE:**

1. Get option from user
2. Get quantity from user
3. Go to respective case and calculate cost (item cost\*quantity)
4. Display total cost

**Output:**

**print("Enter the item:\n 1.Air conditioner(40k) \n 2.telivision(30k) \n 3.Refrigerator(20k) \n 4.Air cooler(15k) \n");**

**$c = <STDIN>;**

**print("Enter quantity\n");**

**$a = <STDIN>;**

**if ($c ==1)**

**{**

**$cost = $a \* 40000;**

**print("cost of Air conditioner: Rs $cost\n");**

**}**

**elsif ($c ==2)**

**{**

**$cost = $a \* 30000;**

**print("cost of telivision: Rs $cost\n");**

**}**

**elsif($c ==3)**

**{**

**$cost = $a \* 20000;**

**print("cost of Refrigerator: Rs $cost\n");**

**}**

**elsif($c ==4)**

**{**

**$cost = $a \* 15000;**

**print("cost of Air cooler: Rs $cost\n");**

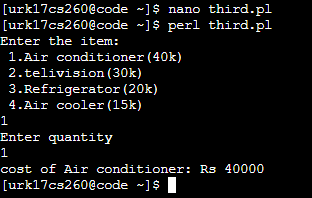
**}**

**else**

**{**

**print("Invalid option");**

**}**



**QUESTION 4: Demonstrate arithmetic assignment operators in perl**

**PROCEDURE:**

1. Get two values from user
2. Use increment operator (+=) and display the value
3. Use decrement operator (-=) and display the value
4. Use multiplication operator (\*=) and display the value
5. Use division operator (/=) and display the value
6. Use modulo operator (%=) and display the value
7. Use power operator (\*\*=) and display the value

**Output:**

**print("Enter a value\n");**

**$a = <STDIN>;**

**print("Enter Increment value\n");**

**$b = <STDIN>;**

**$b += $a;**

**print("Using increment assignment operator $b\n");**

**print("Enter decrement value\n");**

**$b = <STDIN>;**

**$b -= $a;**

**print("Using decrement assignment operator $b\n");**

**print("Enter Multiply value\n");**

**$b = <STDIN>;**

**$b \*= $a;**

**print("Using multiplication assignment operator $b\n");**

**print("Enter division value\n");**

**$b = <STDIN>;**

**$b /= $a;**

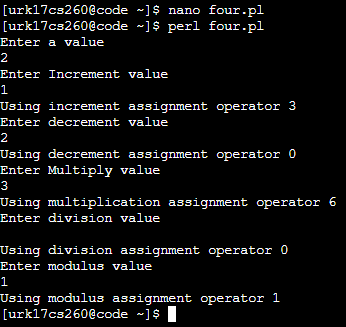
**print("Using division assignment operator $b\n");**

**print("Enter modulus value\n");**

**$b = <STDIN>;**

**$b %= $a;**

**print("Using modulus assignment operator $b\n");**

****

Video Link:

RESULT:

Basic perl programs are successfully executed.