

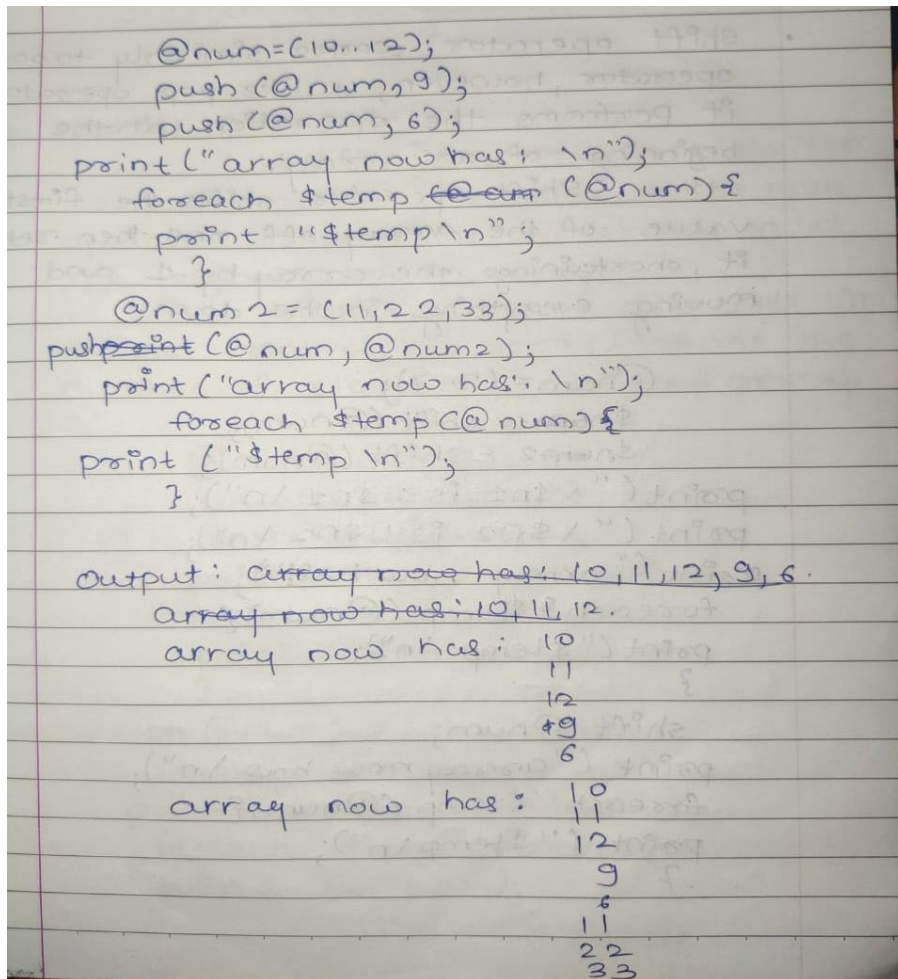
Perl Assignment No. 3

Q1. Write a short note on push and pop operator with example.

Ans:

- **Push Operator:**

Perl provides number of useful function to add and remove elements in an array. So pop operator removes an element from an array and push operator is used to add elements at the end of an array.



- **Push Operator:**

Pop operator removes last element from an array and returns it. Pop off's and returns the last value of an array.

```

@num=(3..7);
$num $n1=pop(@num);
$n2=pop(@num);
print("$n1 is: $n1\n");
print("$n2 is: $n2\n");
print("array now has:\n");
foreach $temp(@num){
    print "$temp\n";
}

pop @num;
print("array now has: \n");
for each $temp(@num){
    print("$temp\n");
}

```

Output:- \$n1 is: 7
 \$n2 is: 6
 array now has: ~~3,4,5~~ ³₅
 array now has: ~~3,4~~ ³₄

Q2. Write a short note on accessing array elements with example.

Ans: When accessing individual elements from an array, you must prefix the variable with a dollar sign (\$) and then append the element index within the square brackets after the name of the variable.

Array indices start from zero, so to access the first element you need to give 0 as indices. You can also give a negative index, in which case you select the element from the end, rather than the beginning, of the array.

```
#!/usr/bin/perl
```

```
@days = qw/Mon Tue Wed Thu Fri Sat Sun/;
```

```
print "$days[0]\n";
print "$days[1]\n";
print "$days[2]\n";
print "$days[6]\n";
print "$days[-1]\n";
print "$days[-7]\n";
```

This will produce the following result -

```
Mon
Tue
Wed
Sun
Sun
Mon
```

Q3. Write a short note on shift and unshift operator with example.

Ans:

- **Shift Operator:**

As we have seen that push and pop operator remove add and remove the last element from an array. The shift and unshift operator performs operation at the beginning of the array.

Shift operator works similarly to pop operator, however, unlike pop operator it performs the operation at the beginning of array.

Shift operator shifts the first value of the array off and then returns it, shortening the array by 1 and moving everything in lower index.

```
@num = (3..7);
$num1 = shift (@num);
$num2 = shift (@num);
print "\$n1 is : $n1 \n";
print "\$n2 is : $n2 \n";
print "array now has \n";
foreach $temp (@num) {
    print "$temp \n";
}
shift @num;
print "array now has \n";
foreach $temp (@num) {
    print "$temp \n";
}
```

```
Output:- $n1 is : 3
         $n2 is : 4
         array now has:
                5
                6
                7
         array now has:
                6
                7
```

- **Unshift Operator:**

Unshift operator works similar to push operator, however, unlike push operator it performs the operation at the beginning of an array.

Unshift operator prepends list to the front of an array, returns the number of elements in the new array.

```
@num = (10..12);
unshift (@num, 9);
unshift (@num, 8);
print("array now has \n");
for each $temp (@num){
    print (" $temp \n");
}

per@num2 = (11, 22, 33);
unshift (@num, @num2);
print("array now has \n");
foreach $temp (@num){
    print (" $temp \n");
}
```

Output :-

```
array now has:
 8
 9
10
11
12
array now has:
11
22
33
 8
 9
10
11
12
```

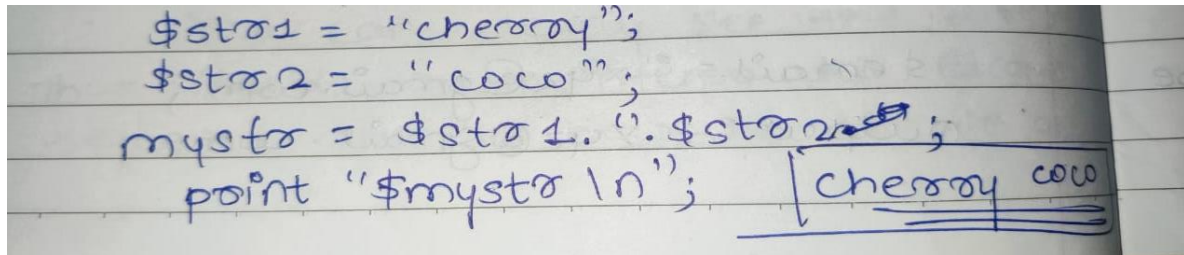

Q4. Write a short note on String operation on concatenation and length of the string with example.

Ans:

- **String operation on concatenation:**

Perl strings are concatenated with a Dot(.) symbol.

This operator takes two scalar variables as operands and combines them in a single scalar variable. Both scalars i.e. left and right will convert into a single string.



```

$str1 = "cherry";
$str2 = "coco";
my $str = $str1 . $str2;
print "$str\n";

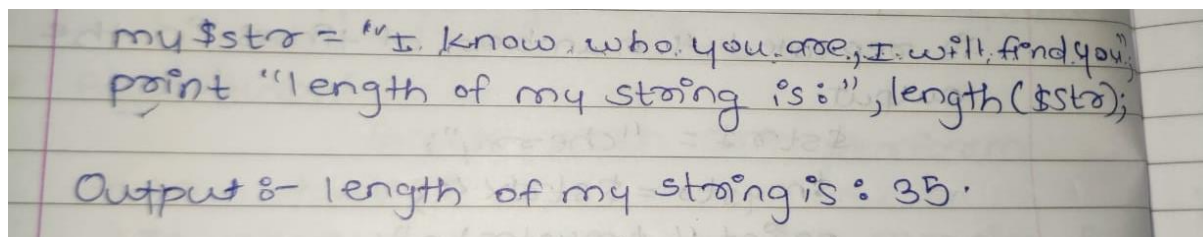
```

The output of the code is: cherry coco

- **Length of the string:**

To find the length of the string in Perl, we use 'length()' function.

This function counts the number of characters in string including whitespaces and return it.



```

my $str = "I know who you are, I will find you";
print "length of my string is : ", length($str);

```

Output :- length of my string is : 35.

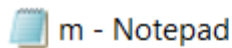
Q5. Write a multidimension program for output:

1 0 1

0 1 1

1 0 0

Ans:



File Edit Format View Help

```
@array = (
[1, 0, 0],
[0, 1, 1],
[1, 0, 0]
);
for($i=0;$i<3; $i++){
for($j=0; $j<3; $j++){
print"$array[$i][$j]";
}
print"\n";
}
```

 Command Prompt

```
Microsoft Windows [Version 10.0.19045.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\manis>cd pp

C:\Users\manis\pp>perl m.pl
100
011
100

C:\Users\manis\pp>
```

Q6. How will you create Hash in Perl and get elements from Hash?

Ans: A hash is a group of key-value pairs. The keys are unique strings and values are scalar values. Hashes are declared using my keyword. The variable name starts with a (%) sign. Hashes are like arrays but there are two differences between them. First arrays are ordered but hashes are unordered. Second, hash elements are accessed using its value while array elements are accessed using its index value.

To access single element of hash, (\$) sign is used before the variable name. And then key element is written inside {} braces.

No repeating keys are allowed in hashes which makes the key values unique inside a hash. Every key has its single value.

```
my %capitals = (
```

```
"India" => "New Delhi",
```

```

"South Korea" => "Seoul",
"USA" => "Washington, D.C.",
"Australia" => "Canberra"
);
print "$capitals{'India'}\n";
print "$capitals{'South Korea'}\n";
print "$capitals{'USA'}\n";
print "$capitals{'Australia'}\n";

```

Output:-

```

New Delhi
Seoul
Washington, D.C.
Canberra

```

Q7. How will you get all keys and all values from Hash?

Ans: Hashes are indexed using \$key and \$value variables. All the hash values will be printed using a while loop. As the while loop runs, values of each of these variables will be printed.

```

my %capitals = (
"India" => "New Delhi",
"South Korea" => "Seoul",
"USA" => "Washington, D.C.",
"Australia" => "Canberra"
);
While(($key,$value)=each(%capitals)){
Print $key. ", " . $value. "\n";
}

```

Output:-

```

Australia, Canberra
India, New Delhi
USA, Washington, D.C.
South Korea, Seoul

```

Q8. How will you get size of Hash?

Ans: The number of key/value pairs is known as the size of hash. To get the size, the first user has to create an array of keys or values and then he can get the size of the array.

Syntax:

```
print scalar keys % hash_variable_name;
```

Ex.

```
%rateof = ('Mango' => 64, 'Apple' => 54, 'Grapes' => 44, 'Strawberry'=>23);
```

```
# creating array of keys
```

```
@keys = keys %rateof;
```

```
$size = @keys;
```

```
print "Hash size using Keys is: $size\n";
```

```
# creating hash of values
```

```
@values = values %rateof;
```

```
$size = @values;
```

```
print "Hash size using Values is: $size\n";
```

Output:-

```
Hash size using Keys is: 4
Hash size using Values is: 4
```

Q9.How will you add and remove elements to Hash?

Ans: User can easily add a new pair of key/values into a hash using simple assignment operator.

Example 1: Addition of element in hash

```
%rateof = ('Mango' => 64, 'Apple' => 54, 'Grapes' => 44, 'Strawberry'=>23);
```

```
@keys = keys %rateof;
```



```

$size = @keys;
print "SIZE OF HASH BEFORE ADDING: is $size\n";
$rateof{'Potato'} = 20;
@keys= keys %rateof;
$size = @keys;
print "SIZE OF HASH AFTER ADDING: is $size\n";

```

Output:-

```

SIZE OF HASH BEFORE ADDING:  is 4
SIZE OF HASH AFTER ADDING:  is 5

```

Example 2: Removing an element from hash using delete function.

```

%rateof = ('Mango' => 64, 'Apple' => 54, 'Grapes' => 44, 'Strawberry'=>23);
@keys= keys %rateof;
$size = @keys;
print "SIZE OF HASH BEFORE DELETING: $size\n";
delete $rateof{'Mango'};
@keys= keys %rateof;
$size = @keys;
print "SIZE OF HASH AFTER DELETING: $size\n";

```

Output:-

```

SIZE OF HASH BEFORE DELETING: 4
SIZE OF HASH AFTER DELETING: 3

```