



BridgeLabz

Employability Delivered

Programming
Constructs –
for Loop
Repetitions

3. Repetition Statement

A repetition construct causes a group of one or more program statements to be invoked repeatedly until some end condition is met.

3. Repetition Statement Types

1. **Fixed count loops** - repeat a predefined number of times.

for ((...)) do done

2. **Variable count loops** - repeat an unspecified number of times.

while [...] do done

for Loop Statement

```
#!/bin/bash -x
for (( counter=1; counter<=5; counter++ ))
do
    echo -n "$counter "
done
printf "\n"
```

for Loop & Execution Statement

```
#!/bin/bash -x
for (( counter=1; counter<=5; counter++ ))
do
    echo -n "$counter "
done
printf "\n"
```

```
Narayans-MacBook-Pro:TerminalCommands narayan$ ./forloopV1.sh
+ (( counter=1 ))
+ (( counter<=5 ))
+ echo -n '1 '
1 + (( counter++ ))
+ (( counter<=5 ))
+ echo -n '2 '
2 + (( counter++ ))
+ (( counter<=5 ))
+ echo -n '3 '
3 + (( counter++ ))
+ (( counter<=5 ))
+ echo -n '4 '
4 + (( counter++ ))
+ (( counter<=5 ))
+ echo -n '5 '
5 + (( counter++ ))
+ (( counter<=5 ))
+ printf '\n'
```



UC 5

Calculating Wages for a Month

Calculating Wages for a Month

```
#!/bin/bash -x

isPartTime=1;
isFullTime=2;
totalSalary=0;
empRatePerHr=20;
numWorkingDays=20;

for (( day=1; day<=$numWorkingDays; day++ ))
do
    empCheck=$((RANDOM%3));
    case $empCheck in
        $isFullTime)
            empHrs=8
            ;;
        $isPartTime)
            empHrs=4
            ;;
        *)
            empHrs=0
            ;;
    esac

    salary=$(( $empHrs * $empRatePerHr ));
    totalSalary=$(( $totalSalary + $salary ));
done
empWageFor.sh (END)
```

Repetition Practice Problems with for loop

1. Write a program that takes a command-line argument n and prints a table of the powers of 2 that are less than or equal to 2^n .
2. Write a program that takes a command-line argument n and prints the n th harmonic number. Harmonic Number is of the form
$$H_n = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n}$$
3. Write a program that takes a input and determines if the number is a prime.
4. Extend the program to take a range of number as input and output the Prime Numbers in that range.
5. Write a program that computes a factorial of a number taken as input.
5 Factorial – $5! = 1 * 2 * 3 * 4 * 5$
6. Write a program to compute Factors of a number N using prime factorization method.
Logic -> Traverse till $i*i \leq N$ instead of $i \leq N$ for efficiency.
O/P -> Print the prime factors of number N .



BridgeLabz

Employability Delivered

Thank
You