LOOPING STATEMENTS

>> Looping statements allows users to execute a block of code/statement repeatedly based on some condition.

Types:

1. while loop
2. for loop
3. until loop

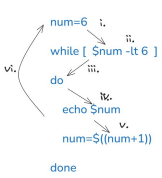
1. while loop:

>> While loop will execute the block of statement until the condition is 'True'.
>> Once the condition becomes 'False', it will stop the execution.
>> In while loop, we have to iterate over the values manually.

Syntax:

```
initialization
while [ condition ]
do
    #statement/code
    iterators (increment, decrement)
done
```

SCRIPT: WRITE A SCRIPT USING WHILE LOOP TO PRINT NUMBER FROM 1 - 5



o/p: 1 2 3 4 5

Iteration1
num=1 ✓
num=num+1
= 1+1 = 2

Iteration2
num=2 ✓
num=num+1
= 2+1 = 3

Iteration3
num=3 ✓
num=num+1
= 3+1 = 4

Iteration4
num=4 ✓
num=num+1
= 4+1 = 5

Iteration5
num=5 ✓
num=num+1
= 5+1 = 6

Iteration6
num=6 ✗

SCRIPT: WRITE A SCRIPT USING WHILE LOOP TO PRINT EVEN NUMBERS FROM 1 - 20

2. for loop:

>> For loop iterates over a list of items, executing a block of code for each item in the list.

Syntax:


```
                    1      2       3
for students in  Komal Lubdhesh Bhavana

do
   echo " Hello $students "
done
```

students -> Komal
Hello Komal

students -> Lubdhesh
Hello Lubdhesh

students -> Bhavana
Hello Bhavana

o/p: Hello Komal
Hello Lubdhesh
Hello Bhavana

SCRIPT: WRITE A SCRIPT USING FOR LOOP TO PRINT NUMBERS FROM 1-5

```
for num in 1 2 3 4 5
do
   echo $num
done
```

```
for num in {1..5}
do
   echo $num
donr
```

>> For loop with range

Initialization
Condition/Range
Iteration

Syntax:

```
for((initialization; condition/range; iteration))
do
   #statements/code
done
```

```
for((num=1;num<6;num++))
do
   echo $num
done
```

num++
↓ SAME AS
num = num+1