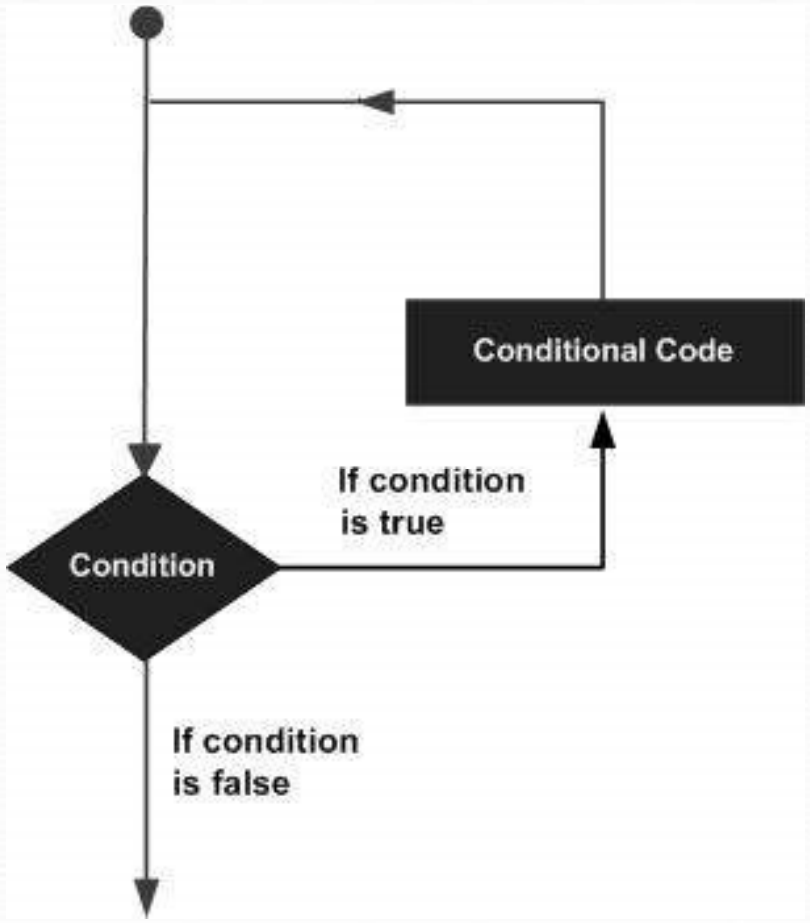
Experiment 8: Demonstrate for and while loops in R

Aim: To understand the working of loops in R

Experiment 8: loops in R

- There may be a situation when you need to execute a block of code several number of times. In general, statements are executed sequentially.

- A loop statement allows us to execute a statement or group of statements multiple times.

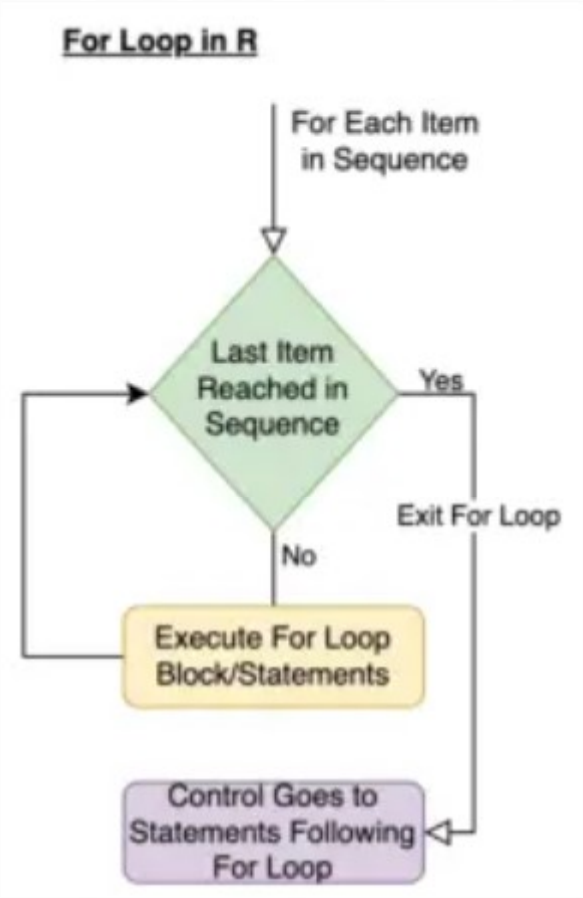


Experiment 8: loops in R

- R programming language provides the following kinds of loop to handle looping requirements.
  - **for**
  - **while**
  - ~~repeat~~

Experiment 8: loops in R : for

- Unlike traditional languages, in R loops, especially **for** loops are used to iterate over elements of a vector, list or data.frame.



- **Syntax:**

```
for (var in sequence)
{
        statement(s)
}
```

Here, the sequence can be vector, array, list, matrix, data.frame e.t.c

Experiment 8: loops in R : for

Example1 on sequence:

```
for (x in 1:10) {
print(x)
}
```

Example 2 on list:

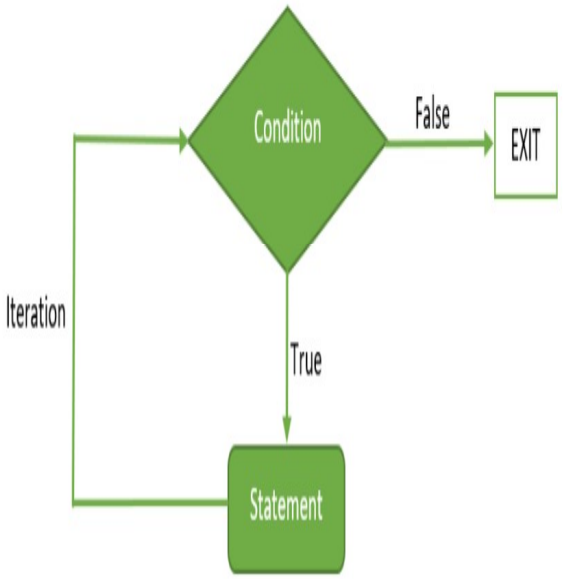
```
fruits=list("banana","apple","melon")
for(x in fruits){
print(x)
}
```

Example 3 on vector:

```
dice <- c(1, 2, 3, 4, 5, 6)
for (x in dice) {
  print(x)
}
```

Experiment 8: loops in R : while

- It is a type of control statement which will run a statement or a set of statements repeatedly unless the given condition becomes false.
- It is also an **entry controlled** loop, in this loop the test condition is tested first, then the body of the loop is executed, the loop body would not be executed if the test condition is false.



Syntax:

```
while ( condition )
{
statement
}
```

Experiment 8: loops in R : while

 # R program to demonstrate the use of while loop

```
i=1
while(i <= 5)
{
   print(i)
 i=i+1
}
```

Experiment 8: loops in R : while

 # R program to calculate factorial of a number

```
n=readline("Enter n")
f=1
i=1
while(i<=n)
{
        f=f*i
        i=i+1
}
print(f)
```

Experiment 8: loops in R : break, next

- Loop control statements change execution from its normal sequence.

- R supports the following Jump control statements.
  - break
  - next

Experiment 8: loops in R : break, next

break Statement:

- When the break statement is encountered inside a loop, the loop is immediately terminated.

- With the break statement, we can stop the loop even if the while condition is TRUE.

Experiment 8: loops in R : break, next

Example: Exit the loop if i is equal to 4.

```
 i=1
while(i<6) {
print(i)
i=i+1

if(i==4) {
break
}
}
```

Experiment 8: loops in R : break, next

Example: Prints the index of first occurrence of 50 in the given list of numbers.

```
 print("Enter The Numbers: ")
v=scan()
for(i in 1:length(v)){
        if(v[i]==50){
                print(i)
                break
                        }
        }
```

Experiment 8: loops in R : break, next

next Statement:

We can skip an current iteration without terminating the loop.

Example: Skip the value of 3

```
i=0
while(i<6) {
i=i+1
if(i==3)     {
next
}
print(i)
}
```

Experiment 8: loops in R : break, next

Example: print even numbers

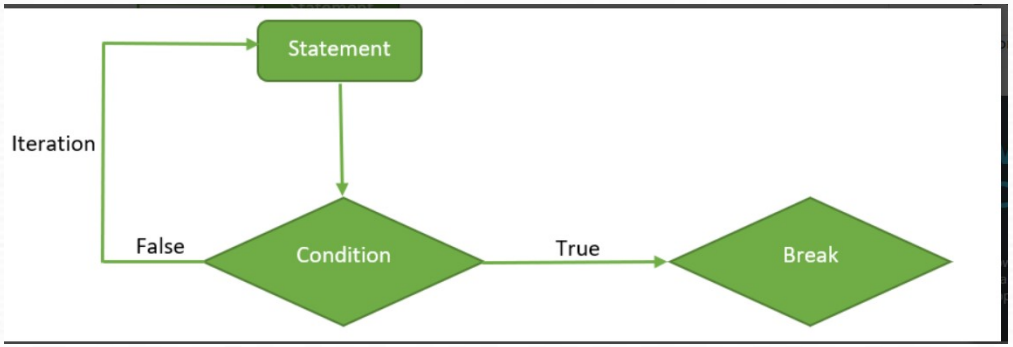
```
for(i in 1:10){
        if(i%%2){
                next
                }
        print(i)
}
```

Experiment 8: loops in R : repeat

- It is a simple loop that will run the same statement or a group of statements repeatedly until the stop condition has been encountered.
- Repeat loop does not have any condition to terminate the loop, a programmer must specifically place a condition within the loop's body and use the declaration of a break statement to terminate this loop.
- If no condition is present in the body of the repeat loop then it will iterate infinitely.

Experiment 8: loops in R : repeat

Syntax:




```
repeat
{
        statement
        if( condition )
        {
        break
        }
}
```

Experiment 8: loops in R : repeat

Example: Program to display a statement five times.

```
 i=0
 repeat
{
 print("Hello!")          #statement to be executed multiple times
 i = i + 1
 if (i == 5)
  {
   break
  }
}
```