Foundations of R Software

Lecture 23

Basics of Calculations

::::

Loops – while and repeat

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Control structures in R :

Control statements,

Functions,

Loops.

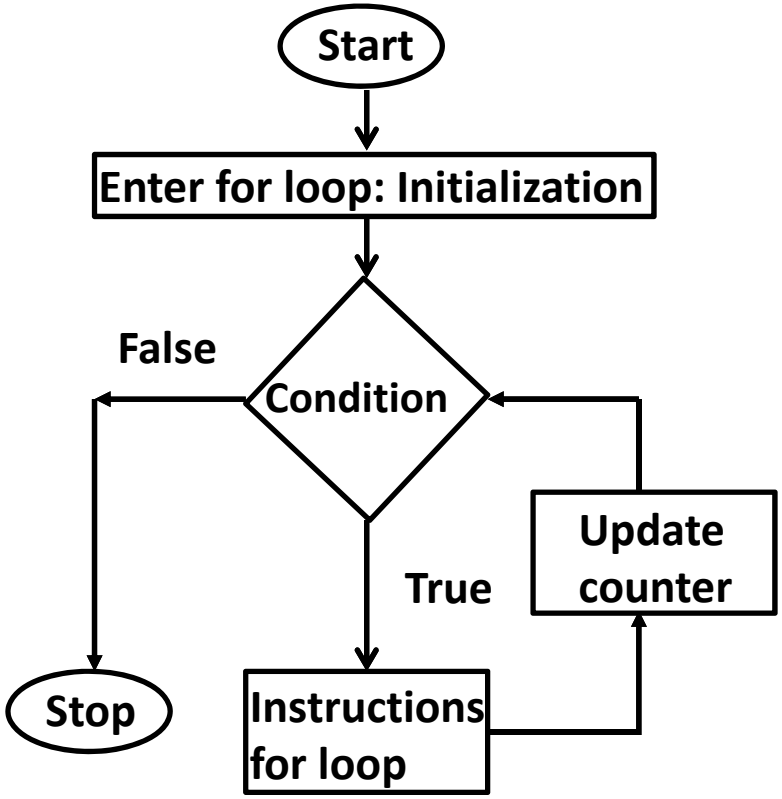
Loops

Repetitive commands are executed by loops

- **for loop**
- **while loop**
- **repeat loop**

1. The `for` loop

If the number of repetitions is known in advance (e.g. if all commands have to be executed for all cases *i* = 1,2,...,*n* in the data), a `for()` loop can be used.



1. The `for` loop

Syntax

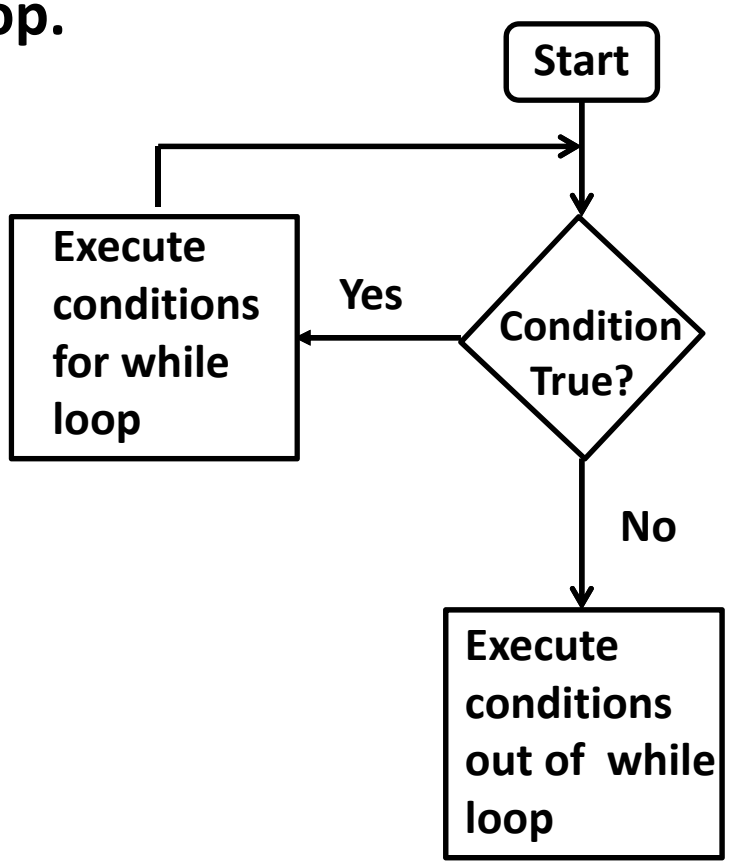
```
for (name in vector) {commands to be executed}
```

A variable with name `name` is sequentially set to all values, which contained in the vector vector.

All operations/commands are executed for all these values.

2. The `while` loop

If the number of loops is not known in before, e.g. when an iterative algorithm to maximize a likelihood function is used, one can use a `while()` loop.



2. The while loop

Syntax

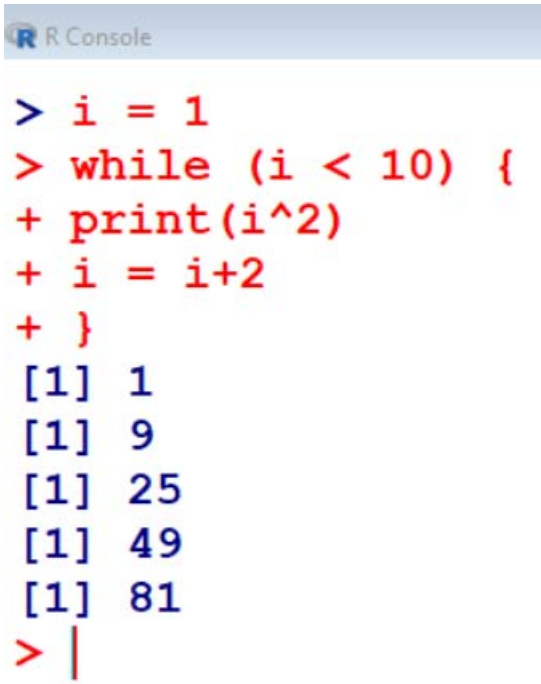
```
while(condition){ commands to be executed as
long as condition is TRUE }
```

If the condition is not true *before entering* the loop, no commands within the loop are executed.

2. The while loop

Example 1:

```
i = 1

while (i < 10) {

print(i^2)

i = i+2

}

[1] 1

[1] 9

[1] 25

[1] 49

[1] 81
```



2. The while loop

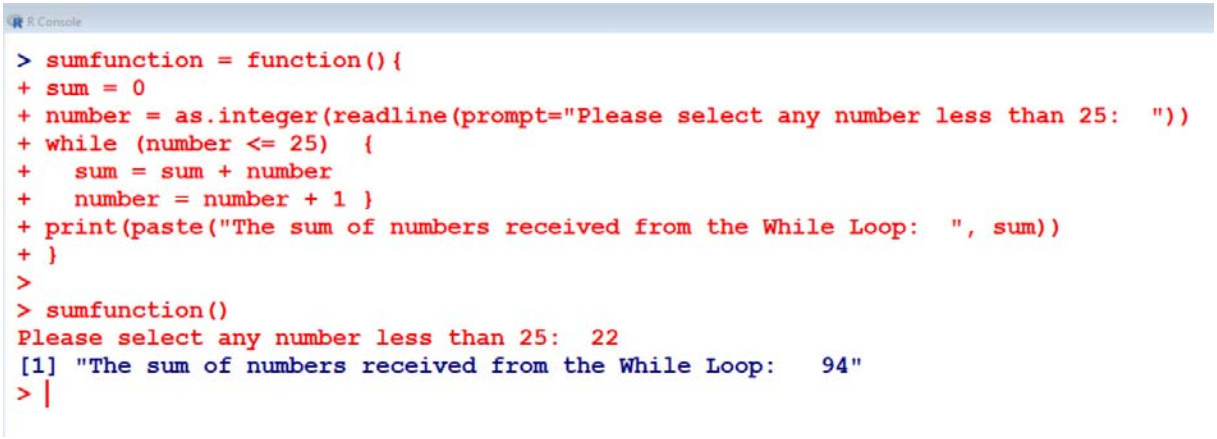
Example 2:

```
sumfunction = function(){
sum = 0
number = as.integer(readline(prompt="Please
select any number less than 25:  "))
while (number <= 25)  {
  sum = sum + number
  number = number + 1 }
print(paste("The sum of numbers received from
the While Loop:  ", sum))
}
```

2. The while loop

Example 2:

```
> sumfunction()
Please select any number less than 25 :  22
[1] "The sum of numbers received from the
While Loop:   94"
> 22+23+24+25
[1] 94
```



3. The `repeat` loop

The repeat loop doesn't test any condition — in contrast to the `while()` loop — *before entering* the loop and also not during the execution of the loop.

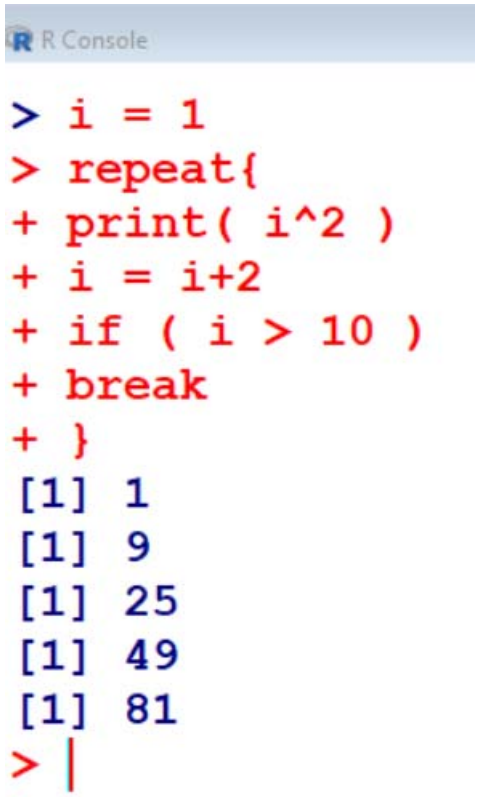
Again, the programmer is responsible that the loop terminates after the appropriate number of iterations. For this the `break` command can be used.

Syntax

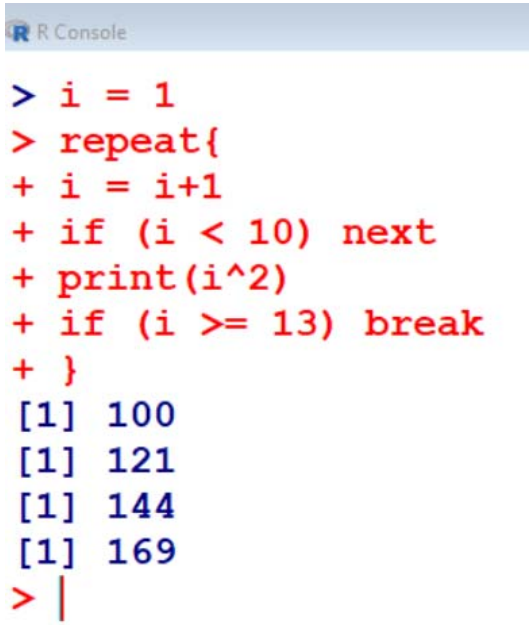
```
repeat{ commands to be executed }
```

Example 3:

```
i = 1
repeat{
print( i^2 )
i = i+2
if ( i > 10 )
break
}
[1] 1
[1] 9
[1] 25
[1] 49
[1] 81
```



Example 4:

Additionally, the command `next` is available, to return to the beginning of the loop (to return to the first command in the loop).

```
i = 1
repeat{
i = i+1
if (i < 10) next
print(i^2)
if (i >= 13) break
}
[1] 100
[1] 121
[1] 144
[1] 169
```