

Control structures

In this lecture

- if-else-if family
- for loop
- Nested for loops
- for loop with if break
- while

Control structures

- Execute certain commands only when certain condition(s) is satisfied (if-then-else)
- Execute certain commands repeatedly and use a certain logic to stop the iteration (for, while loops)

If else family of constructs

If , If else and If-elseif - else are a family of constructs where:

- A condition is first checked, if it is satisfied then operations are performed
- If condition is not satisfied, code exits construct or moves on to other options

If construct

```
if (condition) {  
statements  
}
```

If-else construct

```
if (condition) {  
statements  
} else {  
alternate statements  
}
```

If-else if-else construct

```
if (condition) {  
statements  
} else if (condition) {  
alternate statements  
} else {  
alternate statements  
}
```

If else family of constructs: Example

IN THE EXAMPLE BELOW:

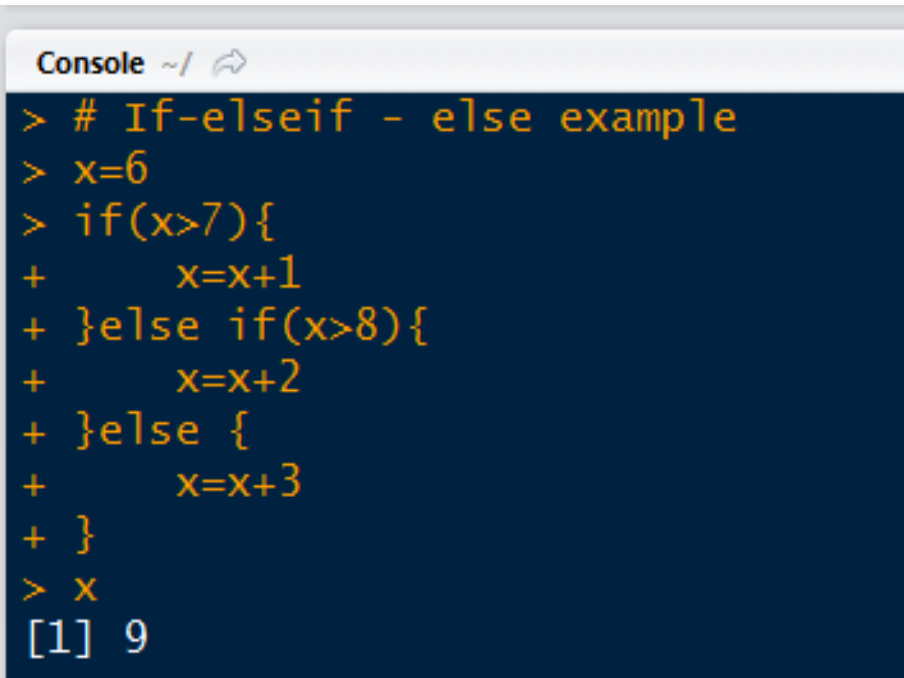
- IF x is greater than 7 operations inside the curved braces would occur
- Else the next condition i.e. $x > 8$ would be checked, if this too is not true
- Final else condition is checked and if that too is false, there is no change in 'x'

Code

```
# If-elseif - else example
```

```
x=6
if(x>7){
  x=x+1
}else if(x>8){
  x=x+2
}else {
  x=x+3}
```

Console Output




```
Console ~/
> # If-elseif - else example
> x=6
> if(x>7){
+   x=x+1
+ }else if(x>8){
+   x=x+2
+ }else {
+   x=x+3
+ }
> x
[1] 9
```

Sequence function

- A sequence is one of the components of a 'for loop'
 - Sequence function syntax : **seq(from, to, by, length)**
 - Creates equi-spaced points between 'from' and 'to'

Parameter	Description
from	starting number
to	ending number
by	increment or decrement (width)
length	Number of elements required

Console Output

```
Console ~/   
> seq(from=1,to=10,by=3)  
[1] 1 4 7 10  
> seq(from=1,to=10,length=4)  
[1] 1 4 7 10  
> seq(from=1,to=10,by=4)  
[1] 1 5 9  
> |
```

for loop, Nested for Loops

The structure of a for loop construct comprises:

- A 'sequence' which could be a vector or a list
- 'iter' is an element of the sequence
- Statements

Nested for-loop : one or more for loop constructs are located within another.

For loop construct

```
for(iter in sequence) {  
  statements  
}
```

Nested For loop construct

```
for(iter1 in sequence1) {  
  for(iter2 in sequence2) {  
    statements  
  }  
}
```

for loop: Example

Open a script file and type the following statements. Save the file as “forloop” and execute the script file

The value of ‘sum’ keeps changing inside the loop

```
1 n=5
2 sum=0
3 for(i in seq(1,n,1)){
4     sum = sum + i
5     print(c(i,sum))
6 }
```

Initializing sum=0

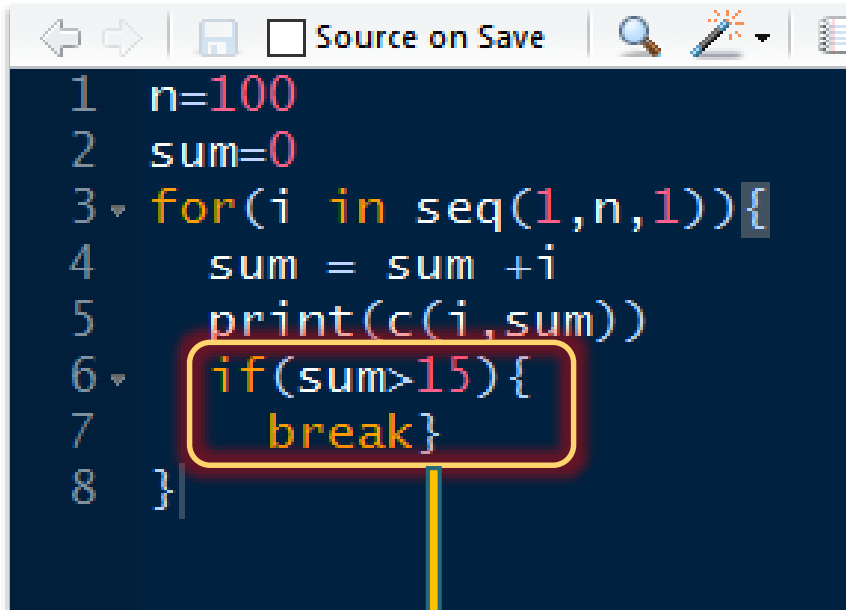
loop variable

Loop variable (i)	sum
1	1
2	3
3	6
4	10
5	15

For- loop with if-break

A “break” statement once executed, program exits the loop even before iterations are complete

“break” command comes out of the innermost loop for nested loops



```
1 n=100
2 sum=0
3 for(i in seq(1,n,1)){
4   sum = sum + i
5   print(c(i,sum))
6   if(sum>15){
7     break
8   }
```

“break” command, the loop is terminated after the 6th iteration

Loop variable (i)	sum	Condition (sum>SUM)
1	1	False
2	3	False
3	6	False
4	10	False
5	15	False
6	21	True

While loop

A while loop is used whenever you want to execute statements until a specific condition is violated

Consider the sequence of natural numbers.

What is the value of the natural number up to which the calculated sum is less than specified “Fin_sum”?

$$1+2+3+...+n = \text{Fin_sum (15)}$$

```

1 sum=0
2 i=0 } Initialize the
        variables
3 Fin_sum=15
4 while(sum<Fin_sum){
5     i=i+1
6     sum=sum+i
7     print(c(i,sum))
8 }
```

Loop variable (i)	sum	Condition (sum<Fin_sum)
1	1	True
2	3	True
3	6	True
4	10	True
5	15	False