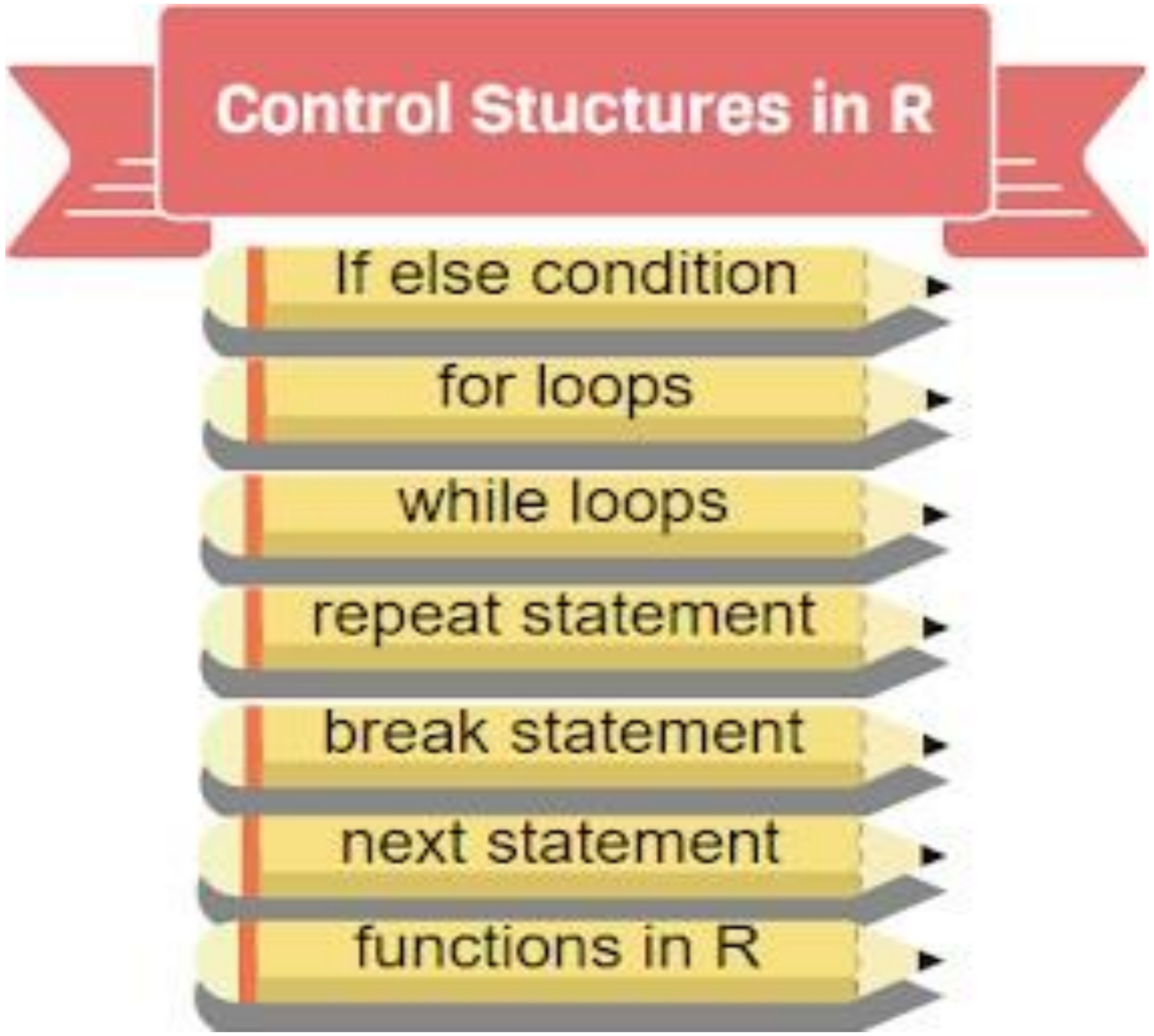
GKTCS INNOVATIONS

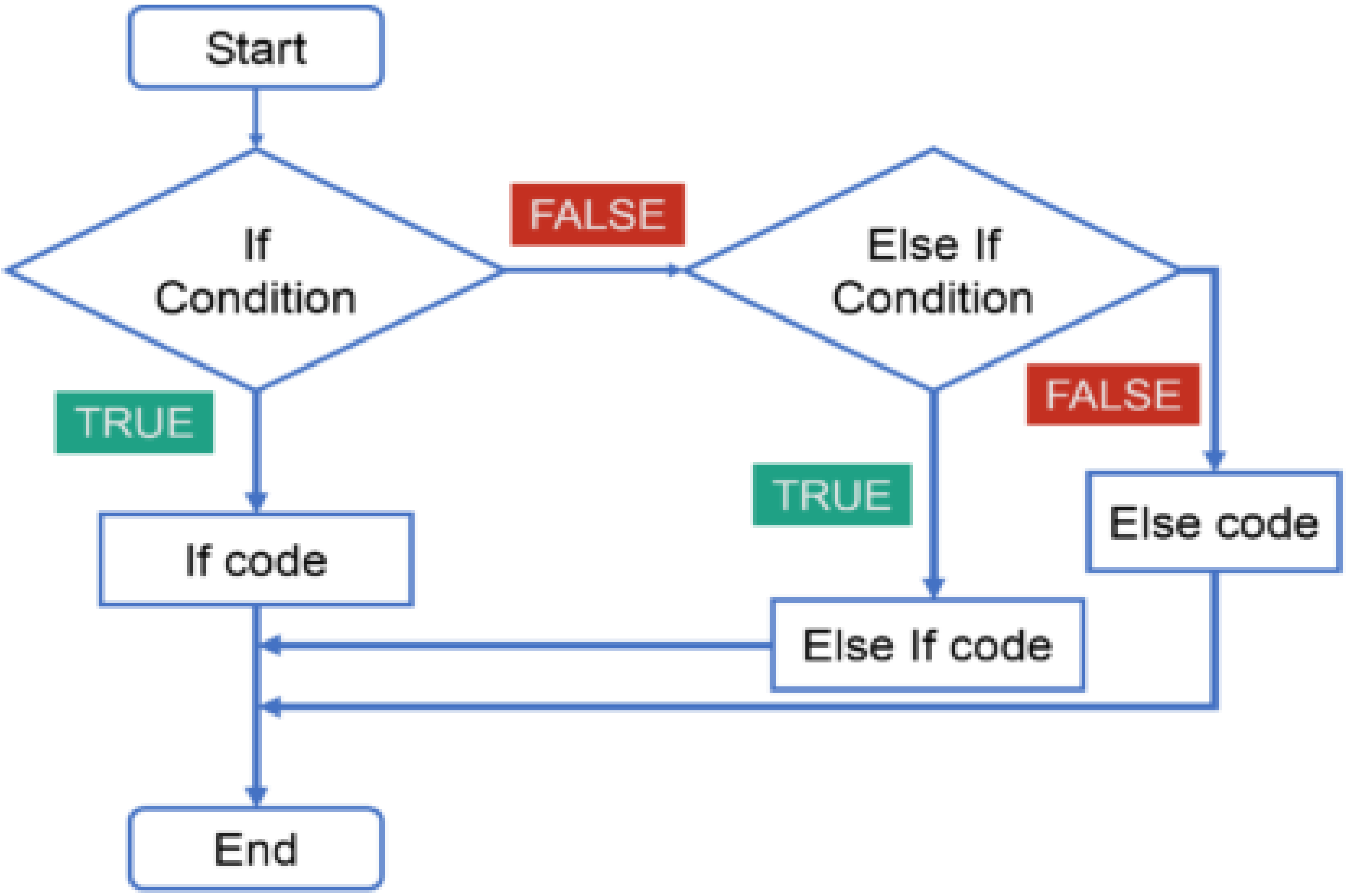
Control Structure in R

Control Structure in R



Control Structure in R

GKTCS INNOVATIONS



Control Structure in R

GKTCS INNOVATIONS

IF

```
if (condition) {
# do something
}
else {
# do something else
}
```

Example :

```
x <- 1:15
if (sample(x, 1) <= 10) {
print("x is less than 10")
}
else {
print("x is greater than 10")
}
```

Control Structure in R

If else statement:

```
x<-5
if(x>1){
 print("x is greater than 1")
 }
 else{
  print("x is less than 1")
  }
```

Control Structure in R

Vectorization with ifelse

```
ifelse(x <= 10, "x less than 10", "x greater than 10")
```

Other valid ways of writing if/else

```
if (sample(x, 1) < 10) {
        y <- 5
} else {
        y <- 0
}
y <- if (sample(x, 1) < 10) {
  5
} else {
  0
}
```

Control Structure in R

```
x=10
 if(x>1 & x<7){
        print("x is between 1 and 7")
        }
        else if(x>8 &  x< 15){
        print("x is between 8 and 15")
        }

[1] "x is between 8 and 15"
```

Control Structure in R

for

A **for** loop works on an itterable variable and assigns successive values till the end of a sequence.

```
for (i in 1:10) {
print(i)
}
x <- c("apples", "oranges", "bananas", "strawberries")
for (i in x) {
print(x[i])
}
```

Control Structure in R

for

```
x = c(1,2,3,4,5)
for(i in 1:5){
print(x[i])
}
```

o/p

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

Control Structure in R

for

```
for (i in 1:4) {
print(x[i])
}
for (i in seq(x)) {
print(x[i])
}
for (i in 1:4) print(x[i])
```

Control Structure in R

GKTCS INNOVATIONS

Nested loops

```
m <- matrix(1:10, 2)
for (i in seq(nrow(m))) {
  for (j in seq(ncol(m))) {
    print(m[i, j])
  }
}
```

Control Structure in R

While

```
i <- 1
while (i < 10) {  print(i)

   i <- i + 1

}
```

Be sure there is a way to exit out of a while loop.

Control Structure in R

GKTCS INNOVATIONS

Example:

```
x = 2.987
while(x <= 4.987) {
        x = x + 0.987

        print(c(x,x-2,x-1))

        }
```

o/p:

```
[1] 3.974  1.974  2.974
[1] 4.961  2.961  3.961
[1] 5.948  3.948  4.948
```

Control Structure in R

Repeat and break

```
repeat {
    # simulations; generate some value have an expectation  if
          within some range,
    # then exit the loop
    if ((value - expectation) <= threshold) {
     break
    }
}
```

Control Structure in R

GKTCS INNOVATIONS

Repeat Loop:

The repeat loop is an infinite loop and used in association with a break statement.

```
 #Below code shows repeat loop:
 a = 1
repeat {
print(a); a = a+1;  if(a > 4)
break
}
```

o/p:

```
[1] 1
[1] 2
[1] 3
[1] 4
```

Control Structure in R

Break Statement

A break statement is used in a loop to stop the iterations and flow the control outside of the loop.

```
#Below code shows break statement:
x = 1:10
for (i in x){
    if (i == 2){
        break
    }
    print(i)
}
[1] 1
```

Control Structure in R

Next

```
for (i in 1:20) {
   if (i%%2 == 1) {
  next
 } else
      {
     print(i)
   }
}
```

This loop will only print even numbers and skip over odd numbers.

Control Structure in R

Next

Next statement enables to skip the current iteration of a loop without terminating it.

```
 for (i in x) {
       if(i == 2){
                    Next
                    }
            print(i)
 }
```

o/p

```
[1] 1
[1] 3
[1] 4
```

Control Structure in R

Switch Statement

- A switch statement permits a variable to be tested in favor of equality against a list of case values.

- In the switch statement, for each case the variable which is being switched is checked. This statement is generally used for multiple selection of condition based statement.

Syntax:

```
switch (test_expression, case1, case2, case3 .... caseN)
```

Control Structure in R

Switch Statement

```
i=2
gk<-switch (
i,
"First",
"Second",
"Third",
"Fourth")
print (gk)

## [1] "Second"
```

Control Structure in R

scan() function

scan() function helps to read data from console or file.

reading data from console

```
x <- scan()
```

Reading data from file.

```
x <- scan("http://www.ats.ucla.edu/stat/data/scan.txt", what = list(age = 0,name = ""))
```

Reading a file using scan function may not be efficient way always.

Control Structure in R

GKTCS INNOVATIONS

Scan Function

Read data from screen if let the file name "", or just without any parameter:

```
>x <- scan("",what="int")
1: 43 #input 43 from the screen
2:
Read 1 item
>x
```

```
[1] "43"
```

Control Structure in R

```
>x <-scan("",what="int")
1: 43 #input 43 from the screen
2: 22
3: 67
4:
Read 3 items
>x
```

```
[1] "43" "22" "67"
```

Large data can be scanned in by just copy and paste, for example paste from EXCEL.

```
>x <- scan()
```

Then use "ctrl+v" to paste the data, the data type will be automatically determined.