Control Structures

- Control structure allows you to control the flow of execution of a script typically inside of a function.
- It is more efficient to use built-in functions rather than control structures whenever possible.
- Common ones are
  - if-else
  - for
  - while
  - repeat
  - break
  - next

If

Simple if
if(condition)
{ #statements
}
	If-else
	if(condition)
	{ #statements }
	else
	{ #statements}

Simple if
if (a>1)
{
print(“a is greater than 1”)
}
	If-else
	if(x>1){
	print("x is greater than 1")
	}else{
	print("x is less than 1")
	}

for

for(var in seq)

{

    # your statements

}

Ex:

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

while

while(condition)

{

    # your statements

}

Ex:

```
x = 2.987
while(x <= 4.987)
{
x = x + 0.987
print(c(x,x-2,x-1))
}
```

Repeat

Repeat is an infinite loop which works with break

```
repeat
{
        # statements
    if(condition)
    {
    break
    }
}
```

Ex:

```
a = 1
repeat { print(a) a = a+1 if(a > 4) break }
```

Next and Break

Next	**Break**

if(condition)
{
next
}
	if(condition)
	{
	break
	}

Ex:
x = 1: 4
for (i in x) {
if (i == 2){
next}
print(i)
}
	Ex:
	x = 1:10
	for (i in x){
	if (i == 2){
	break
	}
	print(i)
	}

Functions

- There are two type of functions
- Built-in Functions
  - Numeric Functions
  - Character Functions
  - Statistical Functions
- User Defined Functions

Basic functions

Function	What It Does
abs(x)	Takes the absolute value of *x*
log(x, base=y)	Takes the logarithm of *x* with base *y*; If base is not specified, returns the natural logarithm
exp(x)	Returns the exponential of *x*
sqrt(x)	Returns the square root of *x*
factorial(x)	Returns the factorial of *x* (*x*!)

Numeric Functions

Function	Description
abs(*x*)	absolute value
sqrt(*x*)	square root
ceiling(*x*)	ceiling(3.475) is 4
floor(*x*)	floor(3.475) is 3
trunc(*x*)	trunc(5.99) is 5
round(*x*, digits=*n*)	round(3.475, digits=2) is 3.48
cos(*x*), sin(*x*), tan(*x*)	acos(*x*), cosh(*x*), acosh(*x*), etc.
log(*x*)	natural logarithm
log10(*x*)	common logarithm
exp(*x*)	e^*x*

Character Functions

Function	Description
substr(x, start=n1, stop=n2)	Extract or replace substrings in a character vector. x <- "abcdef" substr(x, 2, 4) is "bcd" substr(x, 2, 4) <- "22222" is "a222ef"
grep(pattern, x , ignore.case=FALSE, fixed=FALSE)	Search for pattern in x. If fixed =FALSE then pattern is a regular expression. If fixed=TRUE then pattern is a text string. Returns matching indices. grep("A", c("b","A","c"), fixed=TRUE) returns 2
sub(pattern, replacement, x, ignore.case =FALSE, fixed=FALSE)	Find pattern in x and replace with replacement text. If fixed=FALSE then pattern is a regular expression. If fixed = T then pattern is a text string. sub("\\s",".","Hello There") returns "Hello.There"

Character Functions

strsplit(x, split)	Split the elements of character vector x at split. strsplit("abc", "") returns 3 element vector "a","b","c"
paste(..., sep="")	Concatenate strings after using sep string to seperate them. paste("x",1:3,sep="") returns c("x1","x2" "x3") paste("x",1:3,sep="M") returns c("xM1","xM2" "xM3") paste("Today is", date())
toupper(x)	Uppercase
tolower(x)	Lowercase

Statistical Functions

Function	Description
mean(x, trim=0, na.rm=FALSE)	mean of object x # trimmed mean, removing any missing values and # 5 percent of highest and lowest scores mx <- mean(x,trim=.05,na.rm=TRUE)
sd(x)	standard deviation of object(x). also look at var(x) for variance and mad(x) for median absolute deviation.
median(x)	median
quantile(x, probs)	quantiles where x is the numeric vector whose quantiles are desired and probs is a numeric vector with probabilities in [0,1]. # 30th and 84th percentiles of x y <- quantile(x, c(.3,.84))

Statistical Functions

Function	Description
range(*x*)	range
sum(*x*)	sum
min(*x*)	Minimum
max(*x*)	maximum
scale(*x*, center=TRUE, scale=TRUE)	column center or standardize a matrix.

User Defined functions

- Function Definition

  myfunction <- function(*arg1, arg2*)

  {

  Statements

  return(*object*)

  }

- Function invocation
  - <object>=myfunction(x,y)

User defined functions – Contd..

```
#function definition
words = c("R", "datascience", machinelearning","algorithms","AI")
words.names = function(x)
{
  for(name in x)
  {
     print(name)
  }
}
#Calling the function
words.names(words)
```

User defined functions – Contd..

- Create 3 vectors name, age, salary
- Create a data frame DF by combining the 3 vectors using cbind() function
- Write a function which displays the max salaried person name