Lab-01

1. Vectors

Part a

Here we create a vector of 11 consecutive numbers starting from -10. There are several ways this can be accomplished

```
seq(-10,0)
```

```
##  [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1   0
```

```
seq(-10,0, length.out = 11)
```

```
##  [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1   0
```

```
c(-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0)
```

```
##  [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1   0
```

Part b

Here we create a vector of 11 numbers, starting from -0.1 with intervals of 0.5. We can do this one of several ways

```
seq(-0.1, by=0.5, length.out = 11)
```

```
##  [1] -0.1  0.4  0.9  1.4  1.9  2.4  2.9  3.4  3.9  4.4  4.9
```

```
seq(-0.1, 4.9, length.out = 11)
```

```
##  [1] -0.1  0.4  0.9  1.4  1.9  2.4  2.9  3.4  3.9  4.4  4.9
```

```
c(-0.1, 0.4, 0.9, 1.4, 1.9, 2.4, 2.9, 3.1, 3.9, 4.4, 4.9)
```

```
##  [1] -0.1  0.4  0.9  1.4  1.9  2.4  2.9  3.1  3.9  4.4  4.9
```

Part c

Here we calulate the sum and products of the vectors created in parts a&b

```
u <- seq(-10,0)

v <- seq(-0.1, by=0.5, length.out = 11)
```

Sum

```
u + v
```

```
##  [1] -10.1  -8.6  -7.1  -5.6  -4.1  -2.6  -1.1   0.4   1.9   3.4   4.9
```

Product

```
u * v
```

```
##  [1]   1.0  -3.6  -7.2  -9.8 -11.4 -12.0 -11.6 -10.2  -7.8  -4.4   0.0
```

Part d

Here we add 1 to all the elements in vector u

```
u + 1
```

```
##  [1] -9 -8 -7 -6 -5 -4 -3 -2 -1  0  1
```

And we subtract 20% from all the elements in vector v

```
v * 0.8
```

```
##  [1] -0.08  0.32  0.72  1.12  1.52  1.92  2.32  2.72  3.12  3.52  3.92
```

Part e

Here we concatanate vectors u & v

```
w <- c(u, v)

w
```

```
##  [1] -10.0  -9.0  -8.0  -7.0  -6.0  -5.0  -4.0  -3.0  -2.0  -1.0   0.0
## [12]  -0.1   0.4   0.9   1.4   1.9   2.4   2.9   3.4   3.9   4.4   4.9
```

And we report the length of the new vector

```
length(w)
```

```
## [1] 22
```

Part f

Here we return a new vectors, containing the 14th, 15th & 16th elements in vector w

```
w[c(14, 15, 16)]
```

```
## [1] 0.9 1.4 1.9
```

Similarly this vectors is formed of the 2nd, 5th, 9th & 21st elements of w

```
w[c(2, 5, 9, 21)]
```

```
## [1] -9.0 -6.0 -2.0  4.4
```

As w only contains 22 element this comment to return the 23rd element will return nothing

```
w[23]
```

```
## [1] NA
```

Part g

This command reassign the value the 3rd element in w to 100

```
w[3] <- 100

w
```

```
##  [1] -10.0  -9.0 100.0  -7.0  -6.0  -5.0  -4.0  -3.0  -2.0  -1.0   0.0
## [12]  -0.1   0.4   0.9   1.4   1.9   2.4   2.9   3.4   3.9   4.4   4.9
```

This command replaces the values in the 7th, 15th & 22nd positions with new values (200, 300, 400)

```
replace(w, c(7, 15, 22), c(200, 300, 400) )
```

```
##  [1] -10.0  -9.0 100.0  -7.0  -6.0  -5.0 200.0  -3.0  -2.0  -1.0   0.0
## [12]  -0.1   0.4   0.9 300.0   1.9   2.4   2.9   3.4   3.9   4.4 400.0
```

Part h

This command removes vector u from the enviroment

```
rm(u)
```

Part i

This command clears the entire enviroment

```
rm(list = ls())
```

2. Matrices

Part a

Here we create a new matrix of dimensions 4x5, starting with the value 1 & incrementing each new value by 2. The matrix will be populated by row

```
B <- matrix(seq(1, by=2, length.out = 20), 4, 5, byrow = TRUE)

B
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    3    5    7    9
## [2,]   11   13   15   17   19
## [3,]   21   23   25   27   29
## [4,]   31   33   35   37   39
```

This command selects element that are in both the 2nd & 4th row & the 2nd and 3rd column

```
B[c(2,4), c(2,3)]
```

```
##      [,1] [,2]
## [1,]   13   15
## [2,]   33   35
```

Part b

This function creates a new vector of length n, and populates it with the Fibonacci sequence, start with the value 1

```
Fib <- function(n) {
  if (n == 1) return(1)
  x <- c(1, 1)
  while (length(x) < n) {
    index <- length(x)
    new <- x[index] + x[index - 1]
```

```
    x <- c(x,new)
  }
  return(x)
}
```

Now we uses the function as a parameter to create a new matrix of dimentions 3x3

```
C <- matrix(Fib(9), 3,3)

C
```

```
##      [,1] [,2] [,3]
## [1,]    1    3   13
## [2,]    1    5   21
## [3,]    2    8   34
```

Here we select the elements not present in the both the 2nd and 3rd row & the 2nd & 3rd column. This returns a new vector with the element in position 1,1 of C

```
C[ -c(2, 3), -c(2, 3)]
```

```
## [1] 1
```

This command will return a new vector with the element in 1,1 & 1,2

```
C[ -c(3), -c(2 ,3)]
```

```
## [1] 1 1
```

This returns a new vector with all elements from the first row of C

```
C[ , -c(2, 3)]
```

```
## [1] 1 1 2
```