Lab - 3

1 Matrix and Array

1. Creating matrix:

```
A <- matrix(data=c(11,12,21,22),nrow=2,ncol=2)
print(A)

##      [,1] [,2]
## [1,]   11   21
## [2,]   12   22

B <- matrix(data=c(11,12,21,22),nrow=2,ncol=2,byrow=TRUE)
print(B)

##      [,1] [,2]
## [1,]   11   12
## [2,]   21   22
```

2. Using rbind and cbind:

```
rbind(1:3,4:6)

##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6

C <- cbind(c(1,2),c(3,4),5:6)
print(C)

##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

3. Dimension of matrix:

```
dim(A)

## [1] 2 2

dim(C)

## [1] 2 3

nrow(C)
```

```
## [1] 2

ncol(C)

## [1] 3

dim(C)[2]

## [1] 3
```

4. Matrix with random values:

```
randomM1 <- matrix(rnorm(16), nrow = 4)
print(randomM1)

##             [,1]       [,2]       [,3]       [,4]
## [1,] -1.4148986 -1.6618409  1.2725760 -0.1074430
## [2,]  0.5089974  0.6508693 -0.1391225  0.8035442
## [3,] -1.5417864 -0.7337483 -0.3491571  0.4493131
## [4,]  0.1420696  0.2669558 -0.2771453  1.1518678

randomM2 <- matrix(runif(16, min = 0, max = 1), nrow = 4)
print(randomM2)

##           [,1]      [,2]      [,3]        [,4]
## [1,] 0.5144201 0.1600339 0.3464184 0.035282322
## [2,] 0.8623506 0.6432687 0.4744472 0.806271371
## [3,] 0.6848705 0.9151322 0.7994656 0.001574266
## [4,] 0.9162985 0.3758129 0.1948332 0.275465696

randomM3 <- matrix(sample(1:20, 100, replace = TRUE), ncol = 10)
print(randomM3)

##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    6   12   20    6    5   17   20   15   20    15
##  [2,]    9    1   18   10   11    3    2    1   11     7
##  [3,]    7   11    8   16    7   18    8    4    2     5
##  [4,]   18   11   11    9    6   11    5    2    4     2
##  [5,]   11    7   16    6   19    5    6   12    9    13
##  [6,]   11    5   17   13    4    7    5   15   14     9
##  [7,]   17    2    2   17    4   19    4   10    2    13
##  [8,]    9   17   11    6   12    9   11    7   15     2
##  [9,]   15   15   11    5    1    6    2   16   15     5
## [10,]   11   20    3    2   16   10   16   16   17    19
```

   There are several more: rexp(), rpois(), rbinom() etc.

5. Identity and zero matrices:

```
diag(4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```
Z <- matrix(0, 4, 5)
print(Z)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    0    0
## [2,]    0    0    0    0    0
## [3,]    0    0    0    0    0
## [4,]    0    0    0    0    0
```

6. Extracting element, row, column and diagonal from a matrix:

```
randomM3[5,5]
```

```
## [1] 19
```

```
randomM3[,2]
```

```
##  [1] 12  1 11 11  7  5  2 17 15 20
```

```
randomM3[1,]
```

```
##  [1]  6 12 20  6  5 17 20 15 20 15
```

```
randomM3[c(3,1),2:3]
```

```
##      [,1] [,2]
## [1,]   11    8
## [2,]   12   20
```

```
diag(x=randomM3)
```

```
##  [1]  6  1  8  9 19  7  4  7 15 19
```

7. Omitting and Overwriting:

```
C[,-1]
```

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

```
## [1,]    3    5
## [2,]    4    6

randomM3[-c(1,2,3),-c(4,5,6)]

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,]   18   11   11    5    2    4    2
## [2,]   11    7   16    6   12    9   13
## [3,]   11    5   17    5   15   14    9
## [4,]   17    2    2    4   10    2   13
## [5,]    9   17   11   11    7   15    2
## [6,]   15   15   11    2   16   15    5
## [7,]   11   20    3   16   16   17   19

B <- randomM3[-c(5:10),-c(5:10)]
print(B)

##      [,1] [,2] [,3] [,4]
## [1,]    6   12   20    6
## [2,]    9    1   18   10
## [3,]    7   11    8   16
## [4,]   18   11   11    9

B[,1]=c(1,2,3,4)
print(B)

##      [,1] [,2] [,3] [,4]
## [1,]    1   12   20    6
## [2,]    2    1   18   10
## [3,]    3   11    8   16
## [4,]    4   11   11    9

B[c(1,4),c(2,3)] <- 0
print(B)

##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    6
## [2,]    2    1   18   10
## [3,]    3   11    8   16
## [4,]    4    0    0    9

diag(Z) <- rep(x=1,times=4)
print(Z)

##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    1    0    0    0
## [3,]    0    0    1    0    0
## [4,]    0    0    0    1    0
```

8. Operations between matrices:

```
A <- cbind(c(10,20,30),c(4,5,6))
B <- cbind(c(10,30),c(4,9), c(10,4))
print(B)
```

```
##      [,1] [,2] [,3]
## [1,]   10    4   10
## [2,]   30    9    4
```

```
3*B
```

```
##      [,1] [,2] [,3]
## [1,]   30   12   30
## [2,]   90   27   12
```

```
dim(A)
```

```
## [1] 3 2
```

```
dim(B)
```

```
## [1] 2 3
```

```
C <- A%*%B
print(C)
```

```
##      [,1] [,2] [,3]
## [1,]  220   76  116
## [2,]  350  125  220
## [3,]  480  174  324
```

```
det(x=C)
```

```
## [1] 0
```

```
diag(C) <- c(1,1,1)
det(C)
```

```
## [1] 14969441
```

```
C.INV <- solve(C)
C%*%C.INV
```

```
##               [,1]          [,2]         [,3]
## [1,] 1.000000e+00 -5.551115e-17 5.551115e-17
## [2,] 0.000000e+00  1.000000e+00 1.110223e-16
## [3,] 4.857226e-17  5.377643e-17 1.000000e+00
```

```
round(C%*%C.INV,digits = 15)
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

9. Multidimensional Arrays:

```
AR <- array(data=1:24,dim=c(3,4,2))
print(AR)
```

```
## , , 1
##
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
##
## , , 2
##
##      [,1] [,2] [,3] [,4]
## [1,]   13   16   19   22
## [2,]   14   17   20   23
## [3,]   15   18   21   24
```

2 Logical variables

1. Logical values:

```
statement1 <- TRUE
print(statement1)
```

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

```
statement2 <- F
print(statement2)
```

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

```
logical.vector <- c(T,F,F,F,T,F,T,T,T,F,T,F)
logical.matrix <- matrix(data=logical.vector,nrow=3,ncol=4,byrow=T)
print(logical.matrix)
```

```
##      [,1]  [,2]  [,3]  [,4]
## [1,] TRUE FALSE FALSE FALSE
## [2,] TRUE FALSE  TRUE  TRUE
## [3,] TRUE FALSE  TRUE FALSE
```

2. Note down the following operators:

Operator	interpretation
==	Equal to
!=	Not equal to
¿	Greater than
¡	Less than
¿=	Greater than or equal to
¡=	Less than or equal to

```
1==3
```

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

```
5==5
```

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

```
2!=4
```

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

```
6>=5
```

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

```
3<2
```

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

```
randomM4 <- matrix(sample(-10:10, 25, replace = TRUE), ncol = 5)
print(randomM4)
```

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

```
randomM4==9
```

```
##       [,1]  [,2]  [,3]  [,4]  [,5]
## [1,] FALSE FALSE FALSE FALSE FALSE
## [2,] FALSE FALSE FALSE FALSE FALSE
## [3,] FALSE FALSE FALSE FALSE FALSE
## [4,] FALSE FALSE FALSE FALSE FALSE
## [5,] FALSE FALSE FALSE FALSE FALSE
```

```r
any(randomM4==9)
```

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

```r
all(randomM4==9)
```

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

3. Multiple Comparisons:

```r
a<-5
a<6 & a>2 & a!=4
```

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

```r
a>8 | a<6
```

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

```r
a>4 & a<4.5
```

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

```r
A=rbind(1:3,3:5,7:9)
B=cbind(1:3,3:5,7:9)
A>3 & B>3
```

```
##       [,1]  [,2]  [,3]
## [1,] FALSE FALSE FALSE
## [2,] FALSE  TRUE  TRUE
## [3,] FALSE  TRUE  TRUE
```

```r
A>3 && B>3
```

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

4. Logical subsetting and extraction:

```r
vec1 <- c(1.2,1.3,-2.7,4.4,9.8,-3)
vec1[c(T,T,F,T,T,F)]
```

```
## [1] 1.2 1.3 4.4 9.8
```

```r
vec1[!c(T,T,F,T,T,F)]
```

```
## [1] -2.7 -3.0
```

```r
vec1[vec1<0]
```

```
## [1] -2.7 -3.0
```

```r
vec1[vec1>=0]
```

```
## [1] 1.2 1.3 4.4 9.8
```

```r
which(vec1<0)
```

```
## [1] 3 6
```

3 Character variables

1. Strings:

```r
hello <- "Hello World!"
hello
```

```
## [1] "Hello World!"
```

```r
nchar(hello)
```

```
## [1] 12
```

```r
a <- "0.1" # now try: a <- a+1
"alpha" == " alpha"
```

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

```r
c("alpha","beta","gamma")=="beta"
```

```
## [1] FALSE  TRUE FALSE
```

```r
"alpha"<="beta"
```

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

```r
str1=c("boring", "class", "this", "is")
str1
```

```
## [1] "boring" "class"  "this"   "is"
```

```r
cat(str1[3],str1[2],str1[4],"really", str1[1])
```

```
## this class is really boring
```

```r
paste(str1[3],str1[2],str1[4],"really", str1[1],sep="-")
```

```
## [1] "this-class-is-really-boring"
```

You can not use single backslash (/) and double inverted comma (") inside string. The following can be done:

```
cat("I really want a backslash: \\\nand a double quote: \"")

## I really want a backslash: \
## and a double quote: "
```

```
str2 <- "I am a beginner in R!"
substr(x=str2,start=8,stop=15)

## [1] "beginner"

substr(x=str2,start=8,stop=15) <-"PRO"
str2

## [1] "I am a PROinner in R!"
```