1. **Starting and Quitting R:**
   - To start R from PC machine, choose <Program> from the <start menu> and click R.
   - The > is the prompt for entering commands
   - To terminate R, you can just type q()
     i.e. >q()
   - When the program is running, it can be interrupt by ESC

2. **Entering Commands:**
   - Getting help with functions in R
     e.g. > help(sqrt)
   - To perform addition, just type in the formula and press <ENTER>
     e.g. >2+3
     [1] 5
     Similarly, arithmetic and order of operations work exactly in the ordinary way.
     We can see help(Arithmetic)
   - The format for calling a function is functionname (arg1, arg2,…..) or functionname( )for a function which take zero arguments.
     e.g. >sqrt(16)
     [1] 4
     e.g. >abs(-3.81)
     [1] 3.81

3. **Naming conventions in R**
   Names for objects are made up from the upper and lower case roman letters, the digits, 0-9, in any non-initial position and also the period, '.', which behaves as a letter, except in names such as .37 where it acts as decimal point. Notice that some names are reserved as identifiers include break, for, function, if, in, next, repeat, return and while. Avoid using system names for your own objects, in particular avoid c, q, s, t, C, D, F, I, T, diff, mean, pi, range, rank, tree, var. Notice that R is CASE SENSITIVE, so abc and Abc are distinct R names.
4. **Expressions or assignments in R**

An expression command is evaluated and printed

e.g. `>abs(-3.81)`

```
[1] 3.81
```

The assignment symbol is the “<-”, which can also help us to assign value, scalar, vector or matrix to objects. Notice that R does not return any output after an assignment.

To write comment, all statement after the “#” symbol in the current will not be read by R

- **For assign 6 into a scalar “a”**

  ```r
  >a<-6     #assign 6 to a
  ```

  You can also further assign value “a” into “b”

  ```r
  >b<-a
  ```

- **To construct a vector [1, 8, 9, 5, 6, 7, 12]’ into vector “mydata”**

  ```r
  >mydata<-c(1, 8, 9, 5, 6, 7, 12)
  ```

  ```r
  >mydata     or   >print(mydata)
  ```

  ```r
  [1] 1 8 9 5 6 7 12
  ```

  ```r
  >c(rep(mydata,2))
  ```

  ```r
  [1] 1 8 9 5 6 7 12 1 8 9 5 6 7 12
  ```

- **For calling specific cell/cells**

  ```r
  >mydata[2]
  ```

  ```r
  [1] 8
  ```

  or

  ```r
  >mydata[2:4]
  ```

  ```r
  [1] 8 9 5
  ```

- **To change a value of specific cell of a vector**

  ```r
  >mydata[3]<-11
  ```

  ```r
  >mydata
  ```

  ```r
  [1] 1 8 11 5 6 7 12
  ```

- **To compute the summary statistics for “mydata”**

  ```r
  >summary(mydata)
  ```

  ```r
  Min. 1st Qu.  Median    Mean  3rd Qu.    Max.
  1.000   5.500   7.000   7.143   9.500    12.000
  ```

- **Logical operators**

  ```r
  >mydata>3
  ```

  ```r
  [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
• To input character strings
>ch<-c("red", "orange", "yellow", "green", "blue", "violet", "grey")

• To give names from the vector “ch” to mydata”
>names(mydata)<-ch

• To remove names, we just replace the “ch” by “NULL”
>names(mydata)<-NULL.

• To construct a matrix $\begin{bmatrix} 1 & 6 & 8 & 10 \\ 11 & 15 & 16 & 18 \end{bmatrix}$ into matrix “mat123” by filling a vector in a matrix by column
>mat123<-matrix(c(1,11,6,15,8,16,10,18), 2,4)

• To construct a matrix $\begin{bmatrix} 1 & 11 & 6 & 15 \\ 8 & 16 & 10 & 18 \end{bmatrix}$ into matrix “mat456” by filling a vector in a matrix by row
>mat456<-matrix(c(1,11,6,15,8,16,10,18), 2,4, byrow=T)

• For calling specific cells in a matrix
>mat123[1:2,3:4]

Select row or column
>mat123[2,]  #select the second row
[1] 11 15 16 18
>mat123[,1]  #select the first row
[1] 1 11

5. Matrix manipulation
>mat7<-mat123+mat456
$\begin{bmatrix} 1 & 6 & 8 & 10 \\ 11 & 15 & 16 & 18 \end{bmatrix} + \begin{bmatrix} 1 & 11 & 6 & 15 \\ 8 & 16 & 10 & 18 \end{bmatrix}$
$\begin{bmatrix} .1 & .2 & .3 & .4 \\ 1.1 & 1.7 & 1.4 & 1.25 \end{bmatrix}$
### To take transpose of a matrix:

```r
> t(mat456)
```

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>18</td>
</tr>
</tbody>
</table>

### Matrix multiplication

```r
> mat11 <- mat123 %*% mat456
```

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>265</td>
<td>364</td>
</tr>
<tr>
<td>542</td>
<td>812</td>
</tr>
</tbody>
</table>

### Construct a vector m

```r
> m <- c(1, 2, 3, 4)
> mat123 %*% m
```

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>
• **Dot Product m’m**
  
  \[ \text{\texttt{\textbackslash t(m)\%\*\%m}} \]
  
  \[
  \begin{array}{l}
  [1,] 30 \\
  \end{array}
  \]

• **Outer product mm’**
  
  \[ \text{\texttt{m\%\*\%t(m)}} \]
  
  \[
  \begin{array}{cccc}
  [1,] & 1 & 2 & 3 & 4 \\
  [2,] & 2 & 4 & 6 & 8 \\
  [3,] & 3 & 6 & 9 & 12 \\
  [4,] & 4 & 8 & 12 & 16 \\
  \end{array}
  \]

6. **Some useful commands in:**

• To find the dimension, number of rows, number of columns of the matrix:
  
  \[ \text{\texttt{dim(mat123)}} \]
  
  \[
  \begin{array}{l}
  [1] 2 4 \\
  \end{array}
  \]

• To carry out simulation, the most important step is to generate random variables, to generate a vector \textbf{k} of 1000 standard normal random variables:
  
  e.g. > k<-rnorm(1000)

• It is also important for us to know how to generate uniform random variables, to generate a vector \textbf{u} of 1000 uniform[0,1] random variables
  
  e.g. > u<-runif(1000, min=0,max=1)

• To construct a vector “\textbf{x}” of sequence number with same differences
  
  >d<-seq(1.5, 0.5)

• To rank a vector of values
  
  >e<-c(9, 5, 8, 6, 4, 7, 10)

• To sort the elements of a vector
  
  >z<-sort(e)

• To find the ranking of the elements of a vector
  
  >r<-rank(e)
• To find the length of a vector
>l<-length(e)

• To construct an n by n identity matrix
>Iden<-diag(c(rep(1,n)))

• To check the history of the command (default just show 25 lines)
>history()
To show more, for example 50 lines
>history(50)

• List the objects store in the workspace
>ls() or >objects()

• Remove all objects
>rm(list = ls())

• To solve system of equations
\[
\begin{align*}
4x_1 + 2x_2 &= 1 \\
2x_1 + 6x_2 &= 2
\end{align*}
\]
Ax = y
\[
\begin{bmatrix}
4 & 2 \\
2 & 6
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
2
\end{bmatrix}
\]

>A<-matrix(c(4,2,2,6),2,2)
>y<-c(1,2)
>x<-solve(A,y)

\[
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= 
\begin{bmatrix}
0.1 \\
0.3
\end{bmatrix}
\]
Check
>A%*%x

> [,1] 
> [1,] 1 
> [2,] 2

• To compute inverse of A
Since \(A \cdot A^{-1} = I\)

>A<-matrix(c(4,2,2,6),2,2)
>Iden2<-diag(c(rep(1,2)))
>invA<-solve(A,Iden2)

> [,1] [,2]
> [1,] 0.3 -0.1 
> [2,] -0.1 0.2