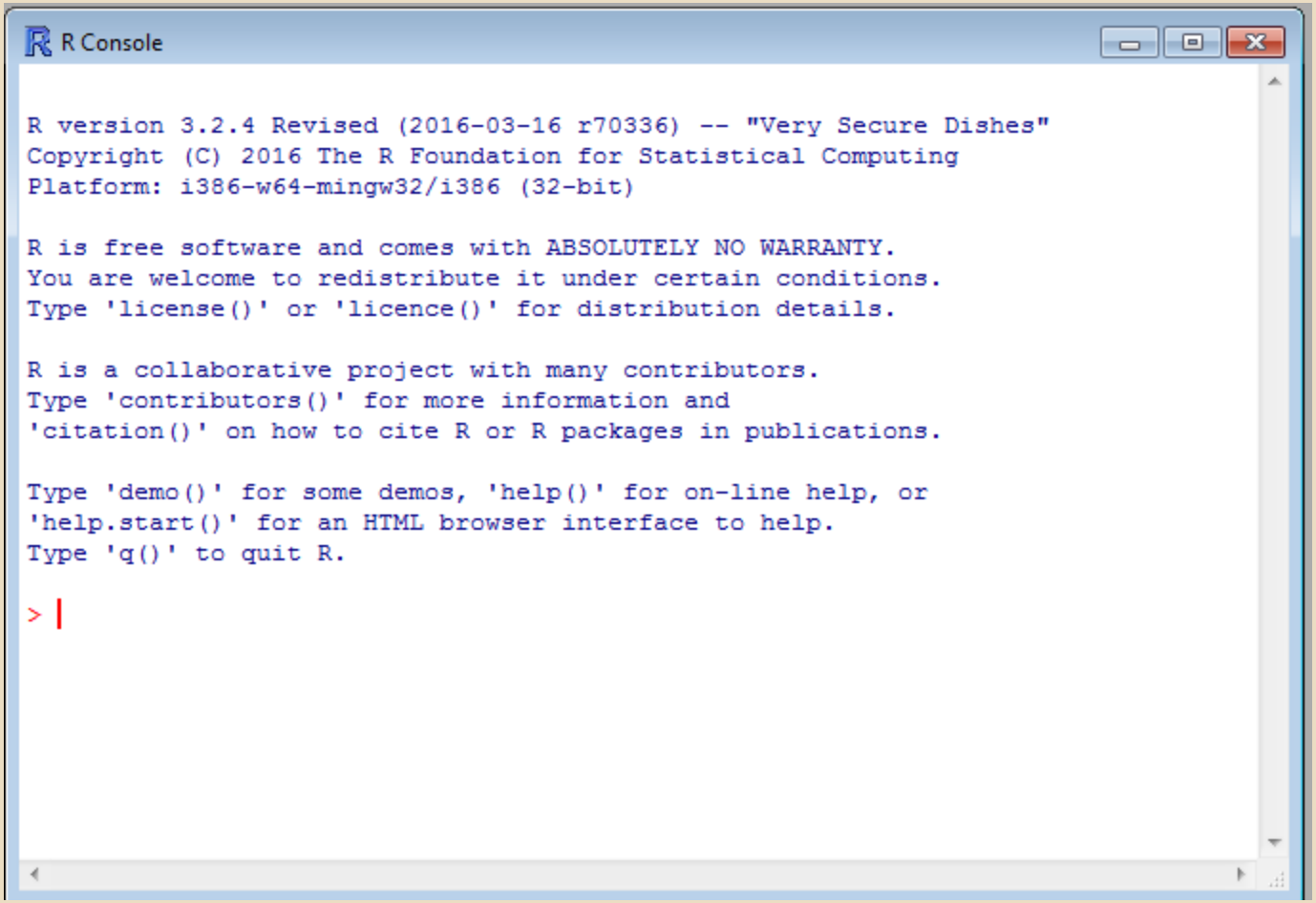
3. Introduction to R

3.1 R console

- When R starts you will see a window called the R Console.
- You are prompted to type commands at the position where greater than symbol (>) appears
- + symbol appears when a the command line is incomplete.

3.1 R console


R Console
R version 3.2.4 Revised (2016-03-16 r70336) -- "Very Secure Dishes"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>

3.1 R console - continued

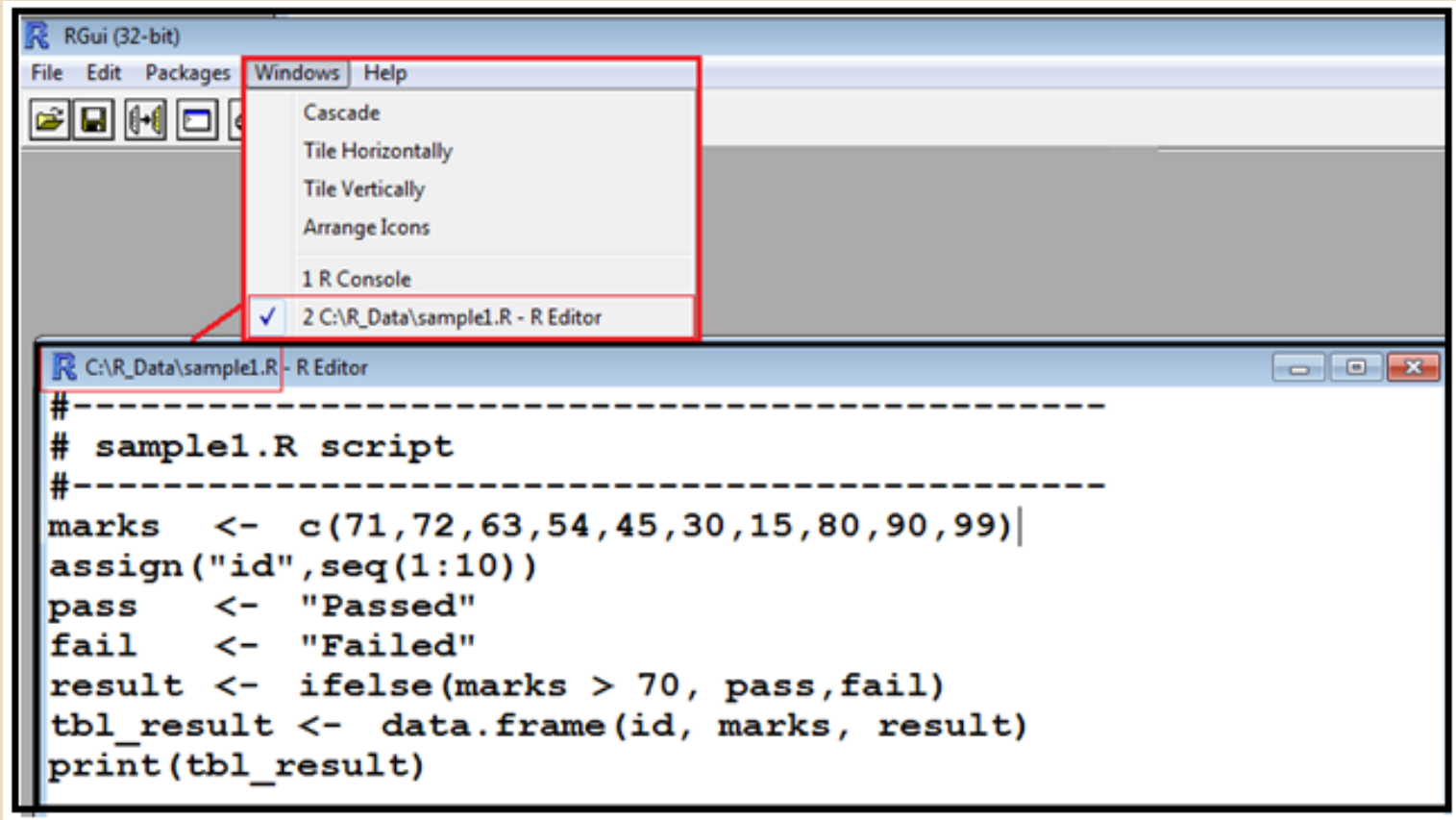
- The R Console allows command editing.
- You will find that the left and right arrow keys, home, end, backspace, insert, and delete work exactly as you would expect.
- You also get a command history; the up and down arrow keys can be used to scroll through recent commands.
- Thus, if you make a mistake all you need to do is press the up key to recall your last command and edit it.

3.1 R console - continued

- It is possible to prepare commands in a file and then have R execute them using the *source* function.

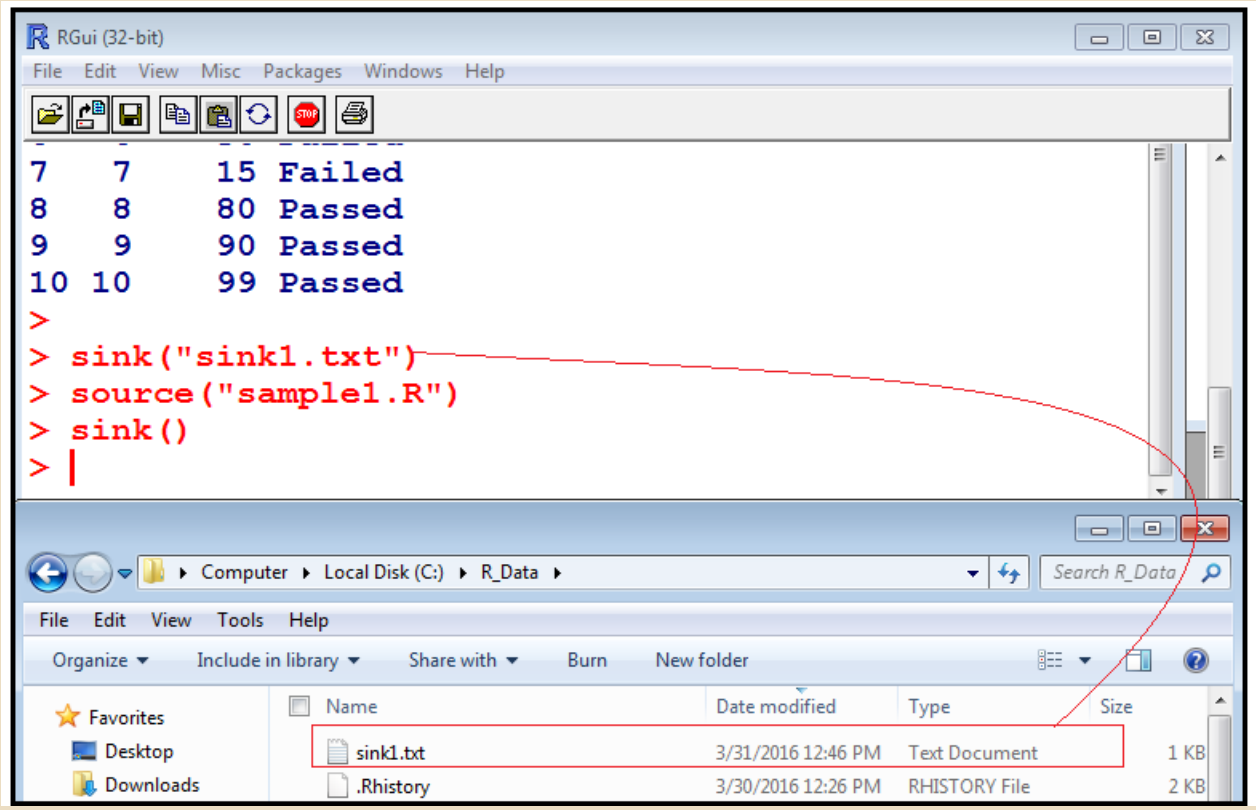
```
> setwd("C:/R_Data")
> source("sample1.R")
   id marks result
1   1    71 Passed
2   2    72 Passed
3   3    63 Failed
4   4    54 Failed
5   5    45 Failed
6   6    30 Failed
7   7    15 Failed
8   8    80 Passed
9   9    90 Passed
10 10    99 Passed
>
```

3.1 R console - continued



3.1 R console - continued

- You can send the output to a file instead of the R Console by using the sink function.



3.1 R console - continued

sink1 - Notepad

File Edit Format View Help

```
   id marks result
1   1    71 Passed
2   2    72 Passed
3   3    63 Failed
4   4    54 Failed
5   5    45 Failed
6   6    30 Failed
7   7    15 Failed
8   8    80 Passed
9   9    90 Passed
10 10    99 Passed
```

3.2 Getting Help

Following commands show how to invoke R help system :

1. help() # gives information about the help system.
2. help(mean) # gives extra information about the mean function.
3. ?mean # gives extra information about the mean function. This function does the same as help(mean).
4. help("if") # If you are searching for a function with special characters, then you must enclose those special characters with double quotes. You also have to do this for functions including the words: if, for and.

Note: # indicates the commencement of a comment line

3.2 Getting Help - continued

```
> help()
starting httpd help server ... done
```

>help.start() # shows the html help documentation. This command will launch a web browser where you can navigate through to find the help you need.

```
> help(if)
Error: unexpected ')' in "help(if)"
> help("if")
```

- ***If you are not sure about the name of the function you are looking for, you can perform a fuzzy search with the apropos().***

```
> apropos("Sol")
[1] "backsolve"     "flush.console" "forwardsolve"
[4] "loadRconsole"  "qr.solve"      "solve"
[7] "solve.default" "solve.qr"
```

3.2 Getting Help - continued

```
> example(mean)

mean> x <- c(0:10, 50)

mean> xm <- mean(x)

mean> c(xm, mean(x, trim = 0.10))
[1] 8.75 5.50
```

- **The function *example()* in utils package run all the R code from Examples part of R's topic with two possible exceptions, dontrun and dontshow.**

For more details, refer to http://127.0.0.1:14321/library/utils/html/example.html

- **For, R FAQ, please refer to http://CRAN.R-project.org/doc/FAQ/R-FAQ.html**
  Author is Kurt Hornik, year = 2014)

3.3 R environment

- The workspace is your current R working environment and includes any user-defined objects (vectors, matrices, data frames, lists, functions).
- At the end of an R session, the user can save an image of the current workspace that is currently reloaded the next time R is started.
- You should keep different projects in different physical directories.
- Assume you have created a directory, "R" in the D drive and you want to use this as your current working directory.

3.3 R environment - continued

- **The following commands help you to manage your workspace:**

setwd("D:/R") # to set the current working directory

getwd() # print the current working directory

ls() # list the objects in the current workspace

options() # to view the current option settings

- **Define objects, vectors x and y and list the objects in the current workspace.**

```
>x <- c(1:10,15,20)
>y <- x^2
>ls()
[1] "x" "y"
```

3.3 R environment - continued

```
>options()
$add.smooth
[1] TRUE

$browserNLdisabled
[1] FALSE

$CBoundsCheck
[1] FALSE

$check.bounds
[1] FALSE

$citation.bibtex.max
[1] 1

$continue
[1] "+ "
```

3.3 R environment - continued

- You can change the default continue character “+” to “?”.

```
> options(continue = "?")
> c <- c("ABC",
?"XYZ")
> c
[1] "ABC" "XYZ"
```

3.3 R environment - continued

- You can change the default prompt character from “>” to “$”.

```
> options(prompt="$")
$
```

3.3 R environment - continued

- If you want to list the objects which contain a given character in their name, the option pattern (which can be abbreviated with pat) can be used:

```
> # to restrict the list of objects whose names
> # contain the word mean
> ls (pat = "mean")
[1] "assumed_mean_x" "mean_a"          "mean_b"          "meanx"
[5] "meany"
> print(ls (pat = "mean"))
[1] "assumed_mean_x" "mean_a"          "mean_b"          "meanx"
[5] "meany"
> # to restrict the list of objects whose names
> # begin with the word mean
> ls (pat = "^mean")
[1] "mean_a" "mean_b" "meanx"  "meany"
> print(ls (pat = "^mean"))
[1] "mean_a" "mean_b" "meanx"  "meany"
```

3.3 R environment - continued

- The function ls.str displays some details on the objects in memory.

```
> # to display the internal structure of the object
> # defined by ls(pat = "mean")
> ls.str(pat = "mean")
assumed_mean_x :  num 1000
mean_a :  num 1017
mean_b :  num 940
```

3.3 R environment - continued

- **To delete the objects in memory, we use the functions rm:**
  rm(x) deletes the object x
  rm(x,y) deletes both x and y
  rm(list=ls()) deletes all the objects in memory.

```
> ay<-ax^2
> az<-ax+2
> # to delete an object ax
> rm(ax)
> # to delete both objects mean_x and mean_y
> rm(ay, az)
> # to delete all objects in memory
> rm(list=ls())
> print(ls(pat="^a"))
character(0)
> print(ls())
character(0)
```

3.4 Managing objects in memory

- An object can be created with the assign operator "<- " ; this symbol can be oriented left-to-right or the reverse.
- All objects have two intrinsic attributes: mode and length.
- The mode is the basic type of the elements of the object; there are four main modes, namely, numeric, character, complex and logical
- The length is the number of elements in the object.

3.4 Managing objects in memory - continued

- The value assigned may be the result of an operation and / or a function, like The function rnorm(1) generates a normal random variable with mean zero and variance unity.

```
> # Four types of mode
> # 1. Numeric
> x <- rnorm(1000)
> cat("\n mode is",mode(x),"and length is",
+ length(x),"\n",sep=" ")

 mode is numeric and length is 1000
> #
> # 2. Character
> y <- c("a","b","c","d","e","f","g")
> cat("\n mode is",mode(y),"and length is",
+ length(y),"\n",sep=" ")

 mode is character and length is 7
```

3.4 Managing objects in memory - continued

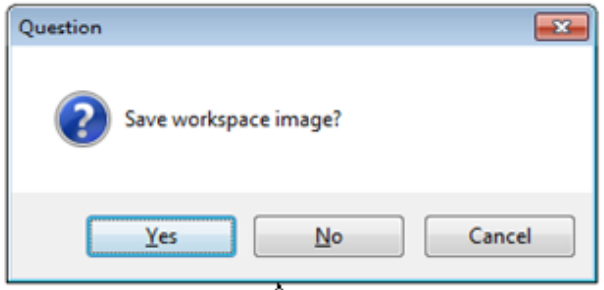
```
> #
> # 3. Complex
> z <- c(1+0i,1+2i,1+3i,1+4i,1+5i,1+6i,1+7i)
> cat("\n mode is",mode(z),"and length is",
+ length(z),"\n",sep=" ")

 mode is complex and length is 7
> #
> # 4. Logical
> a1<-10^2
> a2<-sqrt(100)
> l <- c(a1==a2,a1>a2)
> cat("\n mode is",mode(l),"and length is",
+ length(l),"\n",sep=" ")

 mode is logical and length is 2
>
```

3.5 Customizing R environment

- You can customize the R environment through a site initialization file or a directory initialization file.
- At start up, R will source the Rprofile.site file.
- It will then look for a .Rprofile file to source in the current working directory.
- There are two special functions you can place in this file.
- .First() will be run at the start of the R session and .Last() will be run at the end of the session.
- To quit the session, type quit() or its alias q() to quit R.
- At that point, you will be asked if you want to save the workspace image. This save will save all the work you have done so far, and load it up when you next start R.

3.5 Customizing R environment

```
> setwd("C:/R_Data")
> source("my_first.R")
   ref_id   ht wt      bmi       obs
1       1 1.68 65 23.03005 Not obese
2       2 1.69 58 20.30741 Not obese
3       3 1.70 63 21.79931 Not obese
4       4 1.72 67 22.64738 Not obese
5       5 1.75 72 23.51020 Not obese
6       6 1.76 75 24.21229 Not obese
7       7 1.73 80 26.72993     obese
8       8 1.80 65 20.06173 Not obese
9       9 1.82 74 22.34030 Not obese
10     10 1.83 68 20.30517 Not obese
[1] 1.748
[1] 68.7
[1] 0.05391351
[1] 6.532823
> q()
```

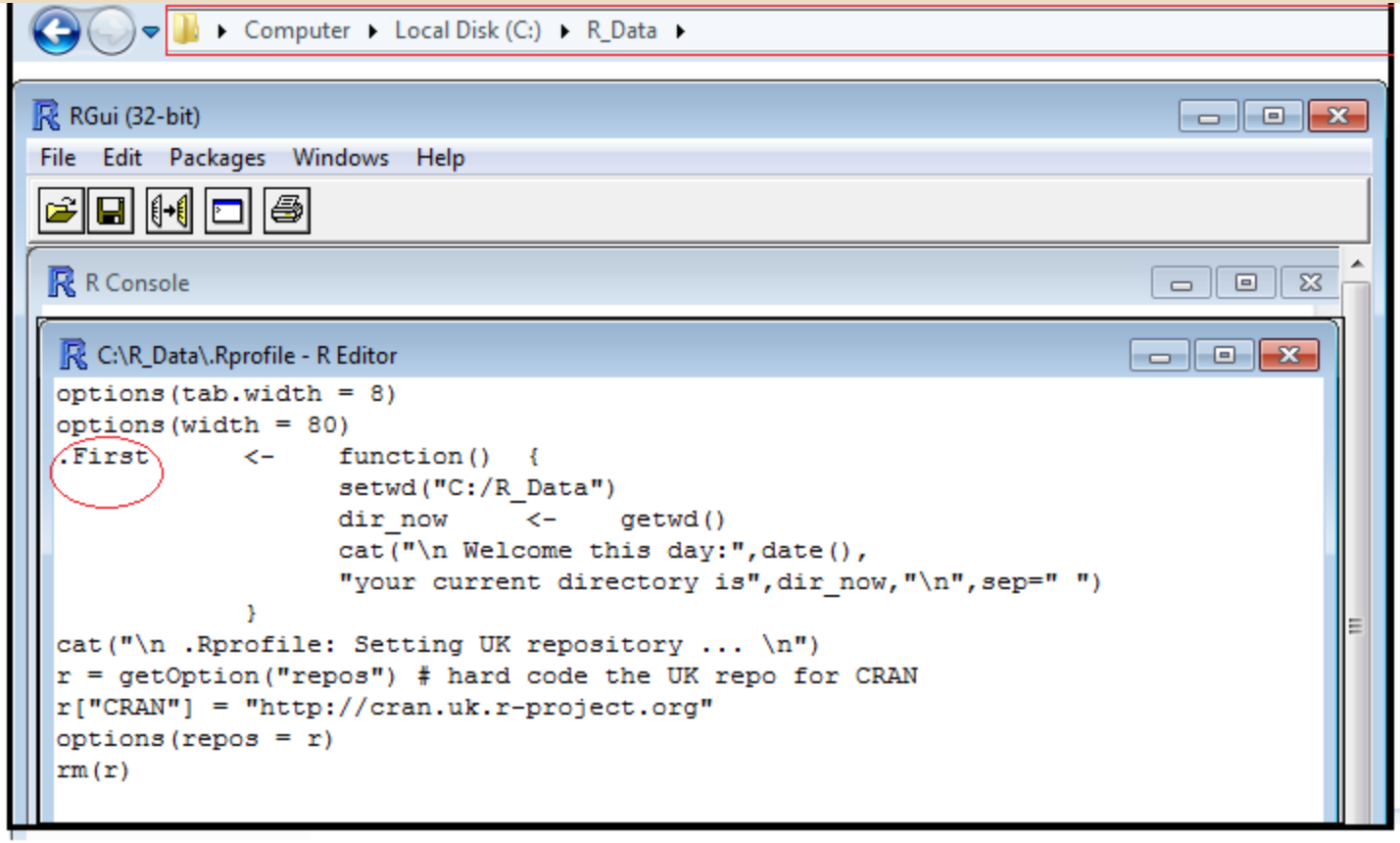


When you quit, you are asked if you want to save the workspace image

3.5 Customizing R environment

1. Create a file (.Rprofile) as follows and place it in C:/R_Data.
   - Create function .First() to do the following:
     1. Set working directory to C:/R_Data
     2. Print a welcome message by printing the date and the current working directory.
2. Double click on the R icon in the C:/R_Data directory.

3.5 Customizing R environment



3.5 Customizing R environment

R Console

```
R version 3.2.4 Revised (2016-03-16 r70336) -- "Very Secure Dishes"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

.Rprofile: Setting UK repository ...
[Previously saved workspace restored]

Welcome this day: Thu Mar 31 13:11:34 2016 your current directory is C:/R_Data
>
>
```

3.6 Text Editors and Integrated Development

Graphical User Interfaces (GUI) include:

1. RGui - comes with the pre-compiled version of R for Windows
2. RStudio - cross-platform open source Integrated Development Environment (IDE) (which can also be run on a remote linux server)

Editors and IDEs

Text editors and IDEs with some support for R include:

jEdit, Eclipse (StatET), Rstudio and Tinn-R

3.7 R Environment (IDE)- R Studio

- **RStudio projects are associated with R working directories.**
- **Rstudio has four panels.**
- **Bottom left: Console panel**

  The console is where you can type R commands and see output.
- **Top left: Editor panel**

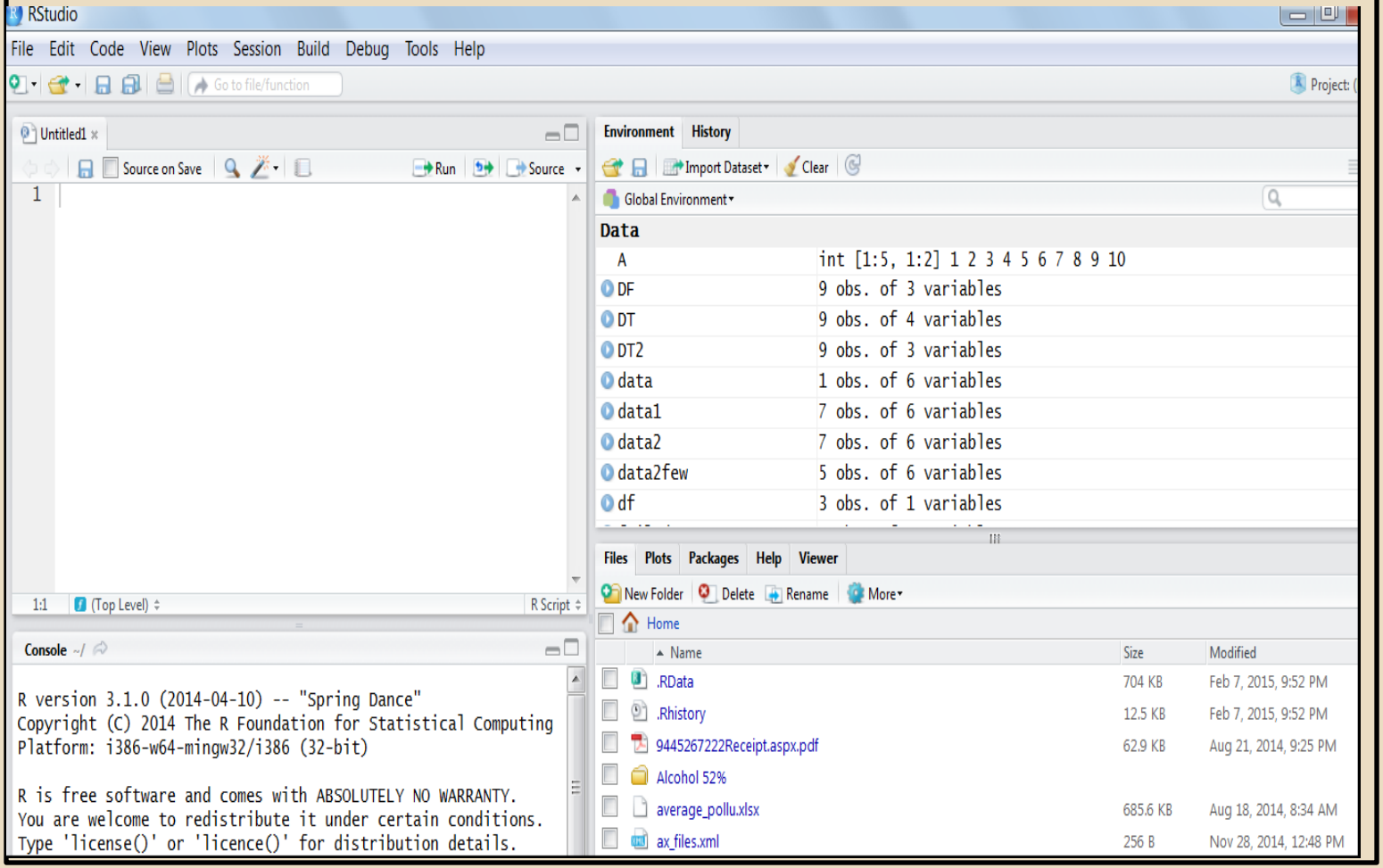
  Here, collection of commands or scripts can be edited and saved.
- **Top right: environment/ history panel**

  1. ***In the environment tab, you see data and values R has in memory.***
  2. ***The history tab shows what was typed before.***

3.7 R Environment (IDE)- R Studio

- **Bottom Right: files/ plots/ packages/ help / viewer panel**
  1. ***The file tab shows all the files and folders in your default workspace.***
  2. ***The plots tab shows all your graphs.***
  3. ***The packages tab shows lists the series of packages or add-ons needed to run certain proceses.***
  4. ***For additional help see the help tab.***
  5. ***The Viewer tab displays local web content (e.g. graphical output).***

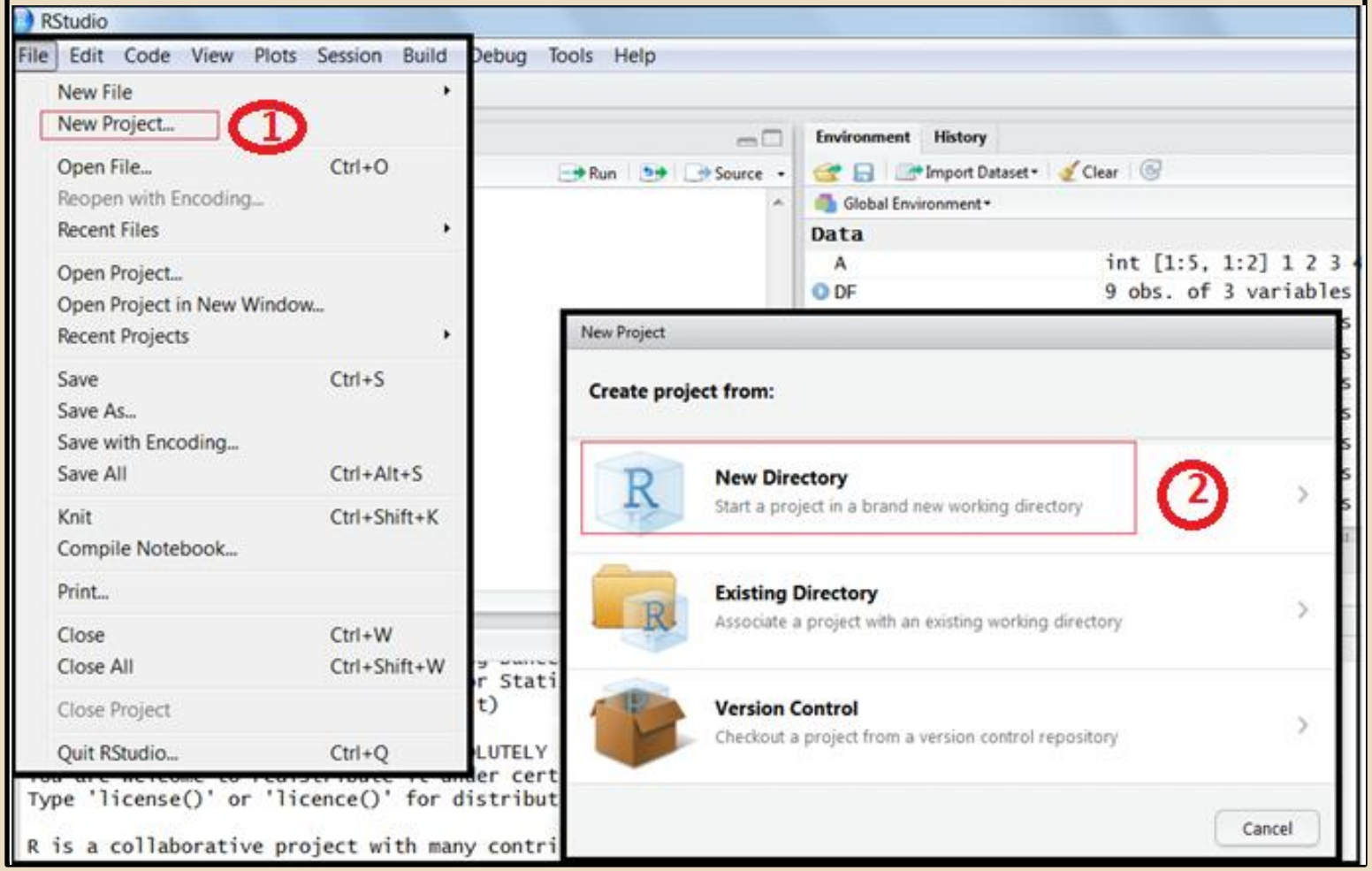
3.7 R Environment (IDE)- R Studio



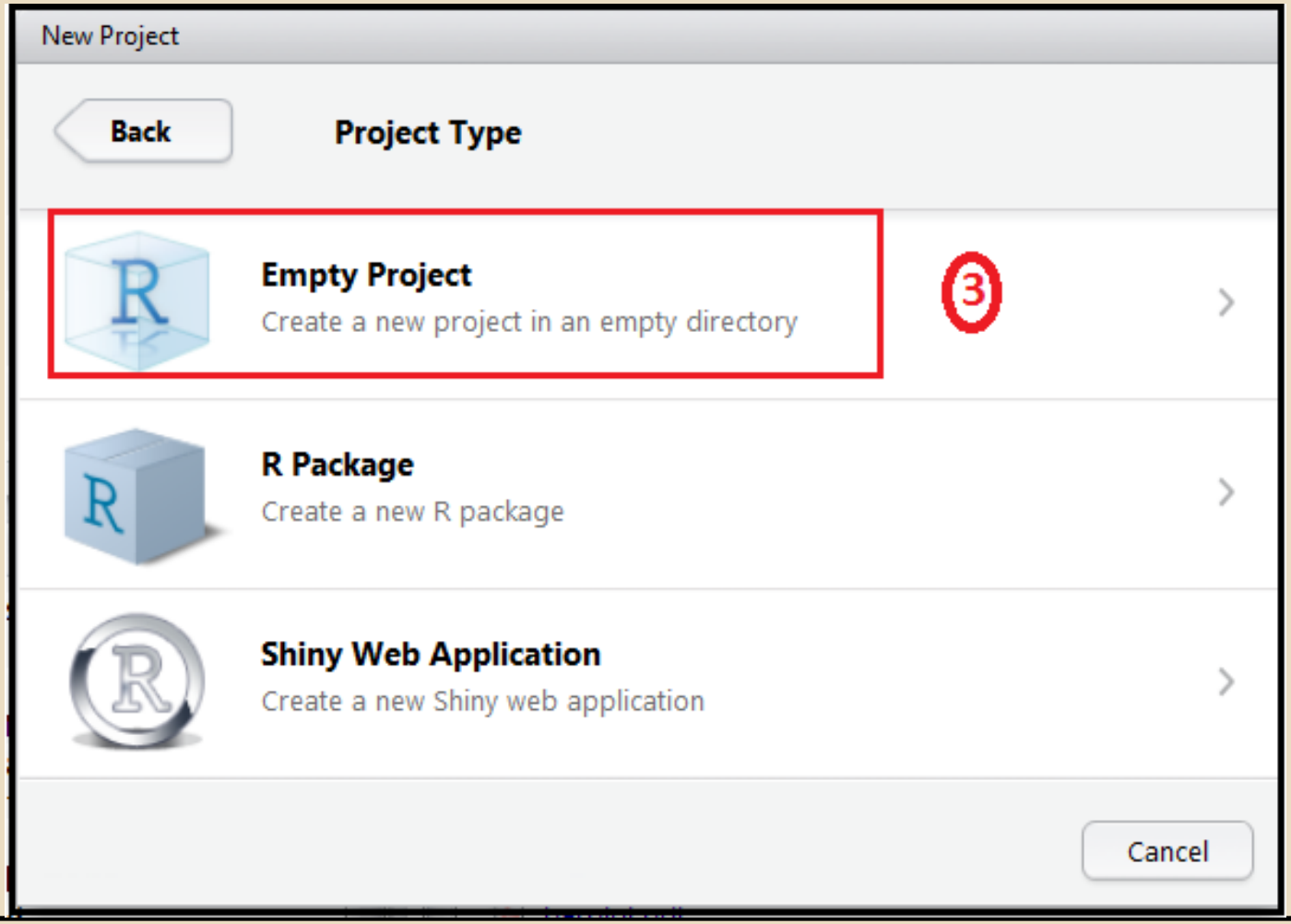
3.8 Working with Projects in RStudio

- Projects help you to manage your work with R language very easily by dividing into multiple contexts, each with their own working directory, workspace, history and source documents.
- RStudio projects are associated with R working directories.
- Let us create a project “1_Project”.

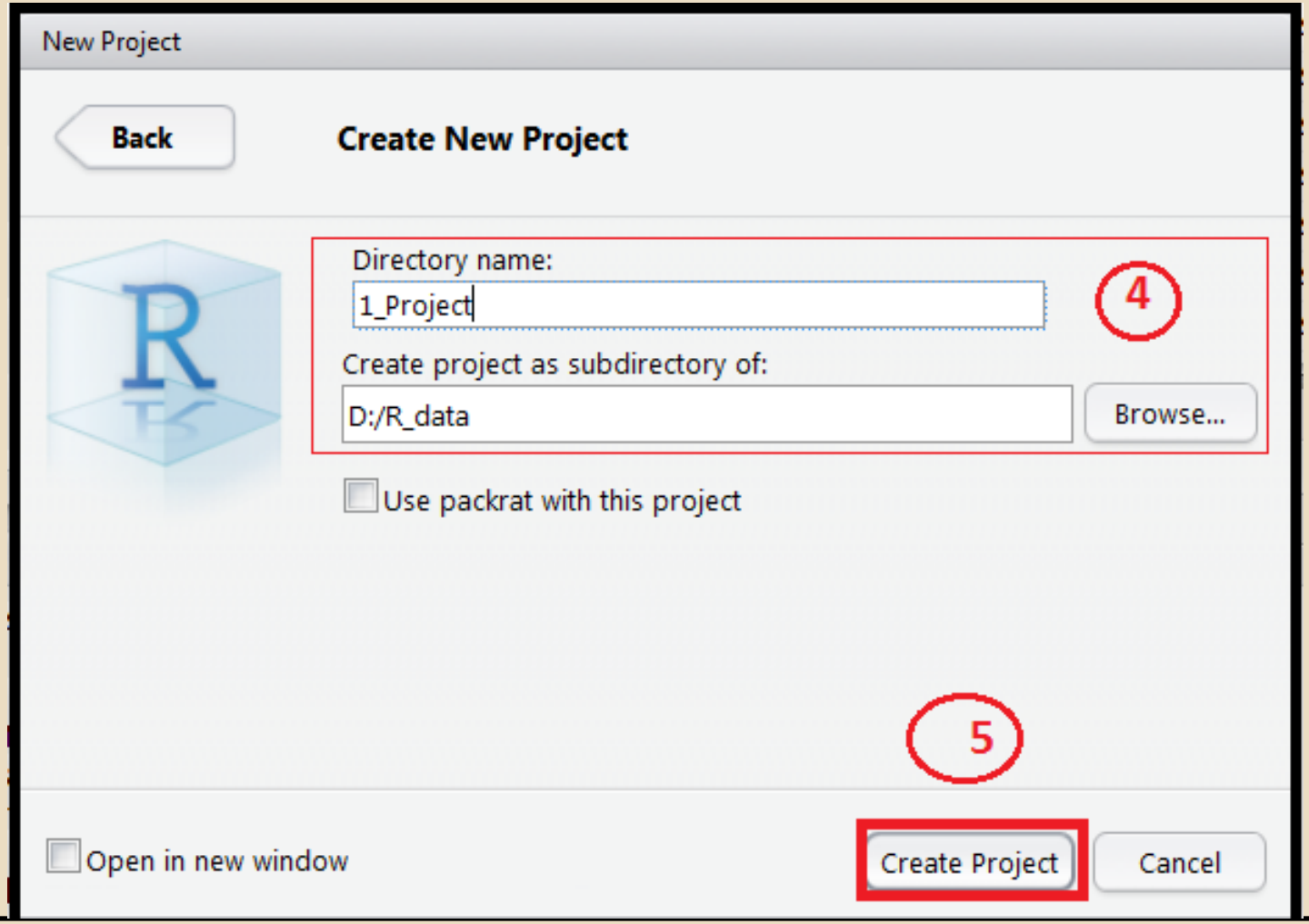
3.8 Working with Projects in RStudio



3.8 Working with Projects in RStudio

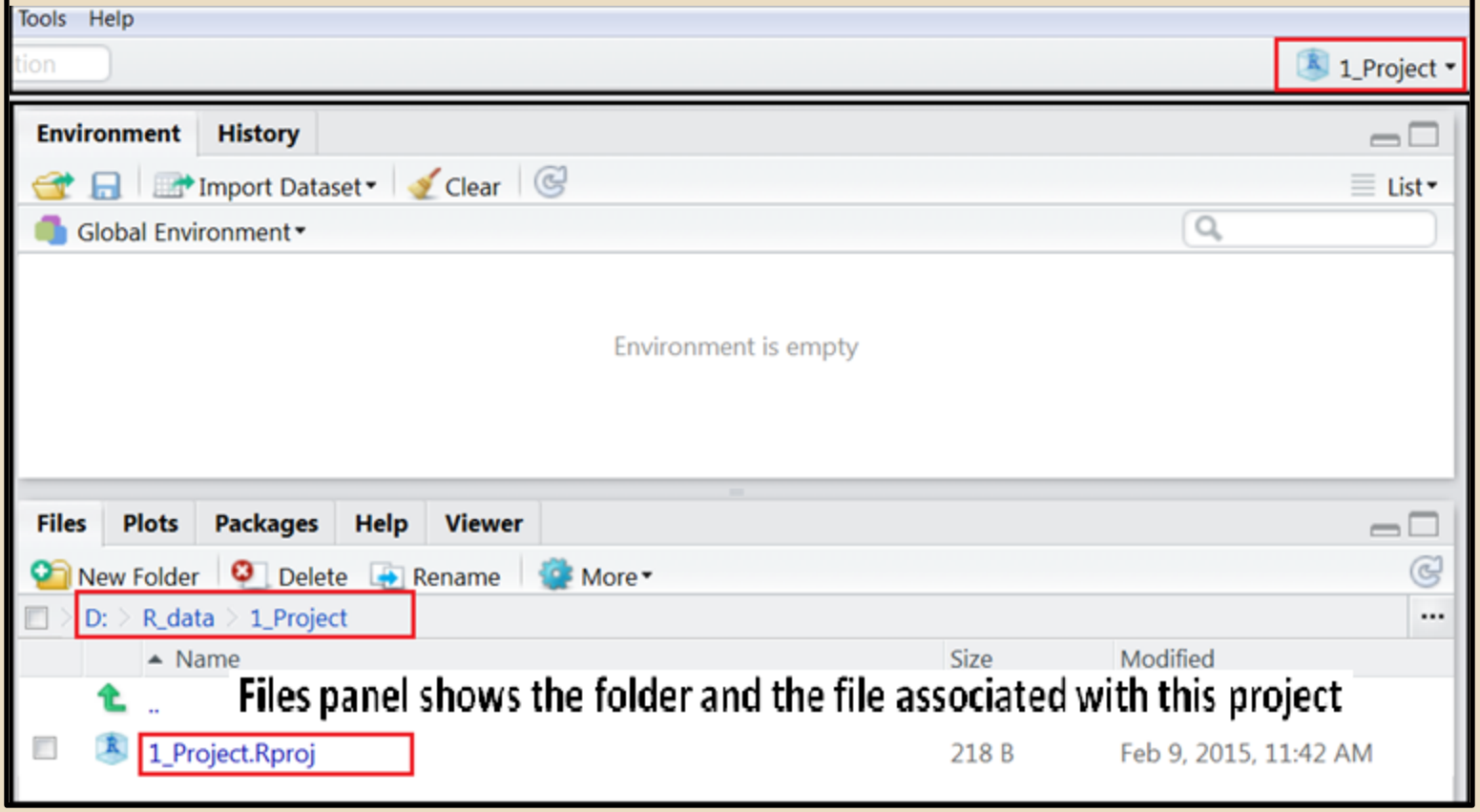


3.8 Working with Projects in RStudio



3.8 Working with Projects in RStudio

Now you can see the new project, 1_Project in the right most corner of the RStudio window.



3.8 Working with Projects in RStudio

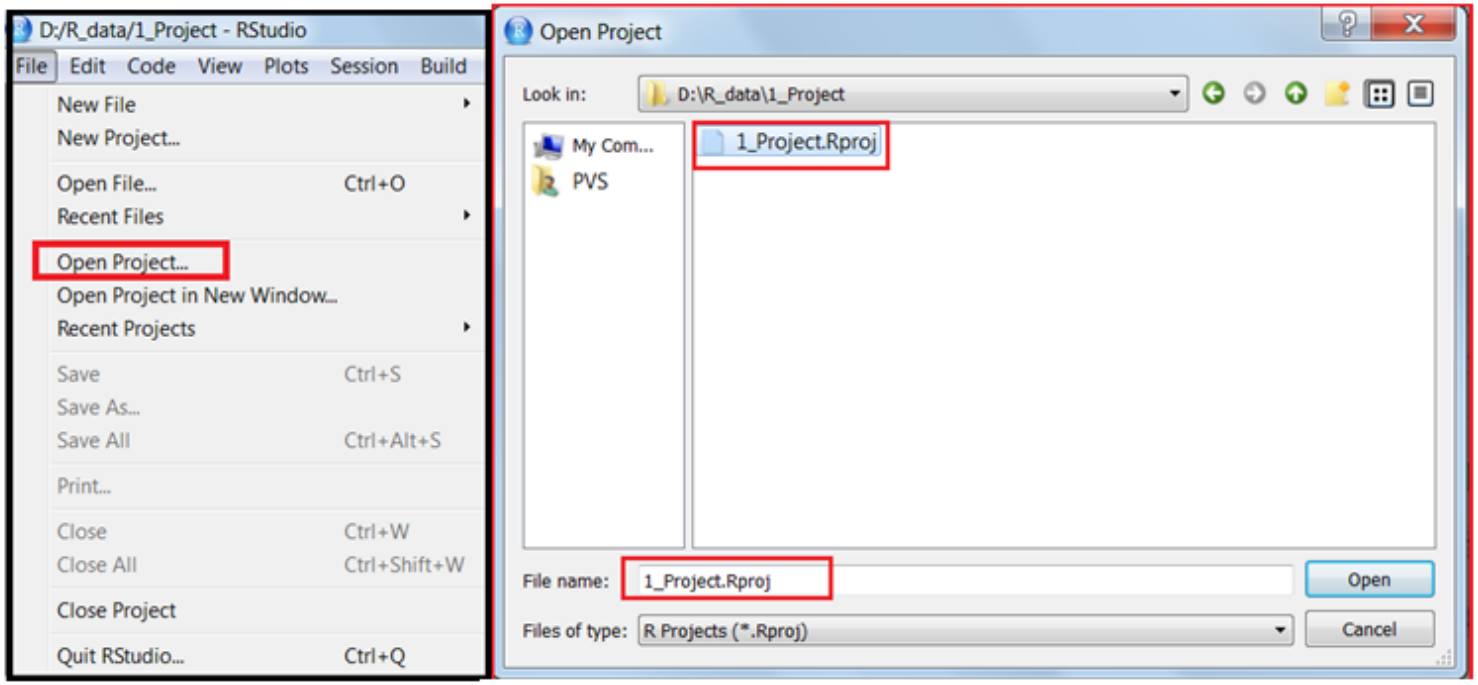
Console D:/R_data/1_Project/

```
R version 3.1.0 (2014-04-10) -- "Spring Dance"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> getwd()
[1] "D:/R_data/1_Project"
>
```

Working directory is set to "D:/R_Data/1_Project.

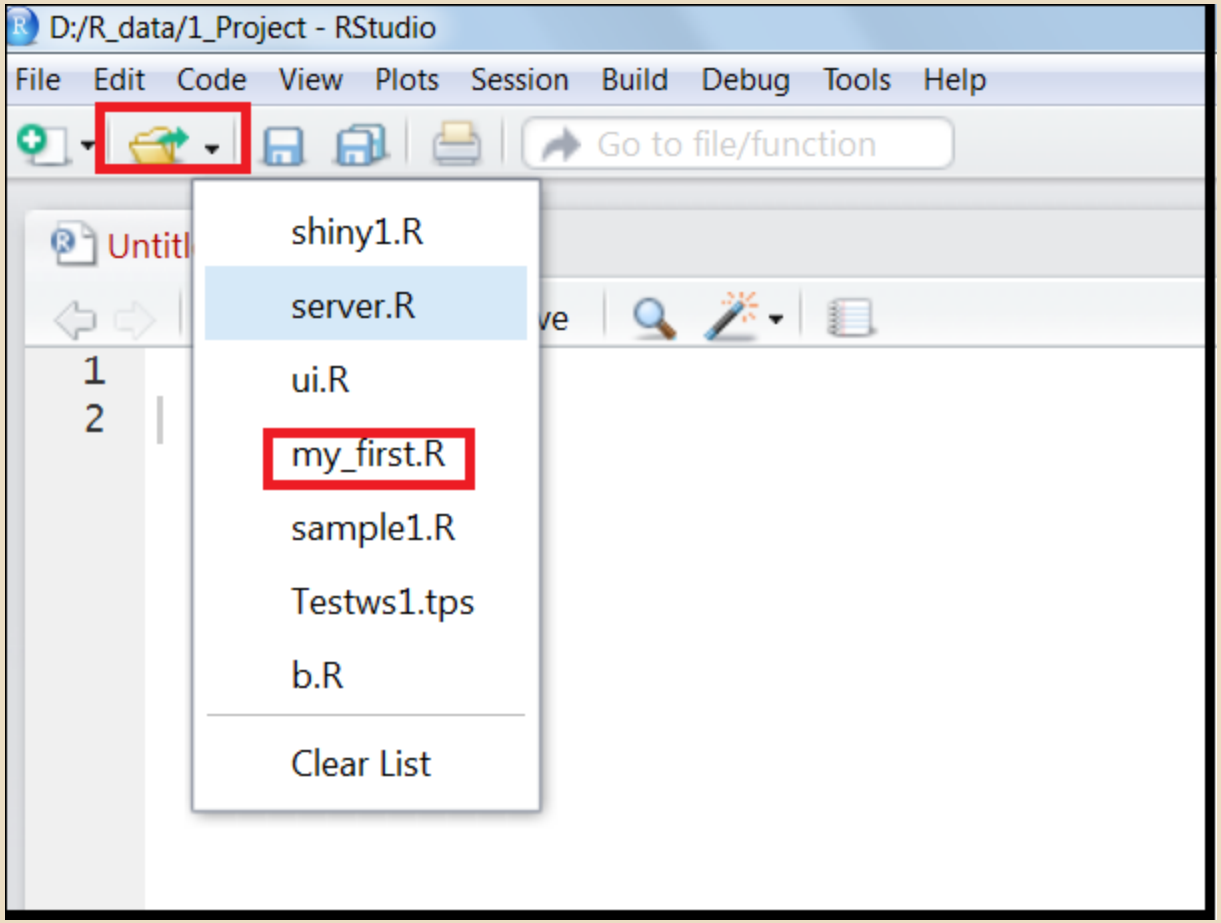
3.8 Working with Projects in RStudio

- **Open the project, *1_Project as shown below:***

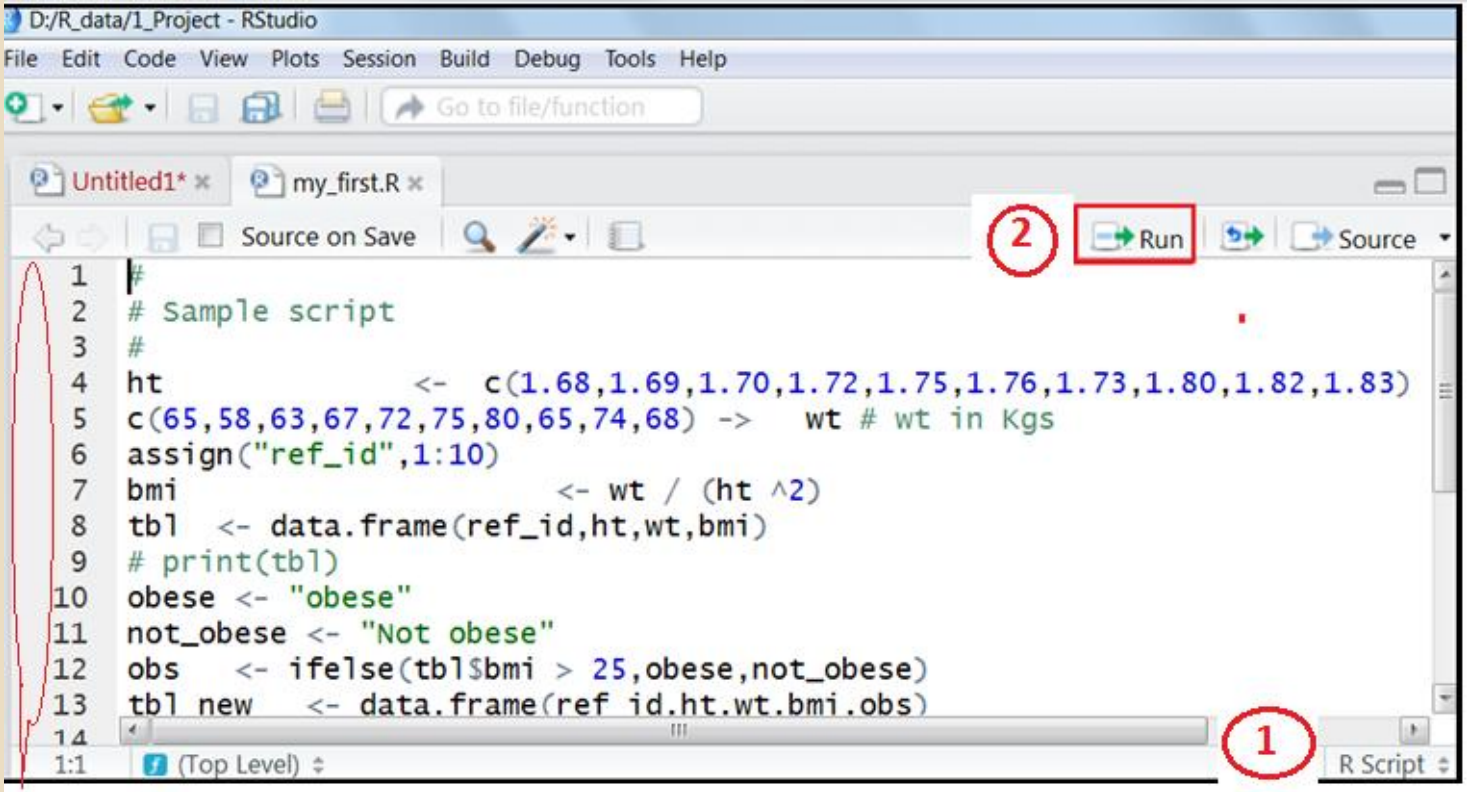


- **Load the program my_first.R and run the same**

3.8 Working with Projects in RStudio



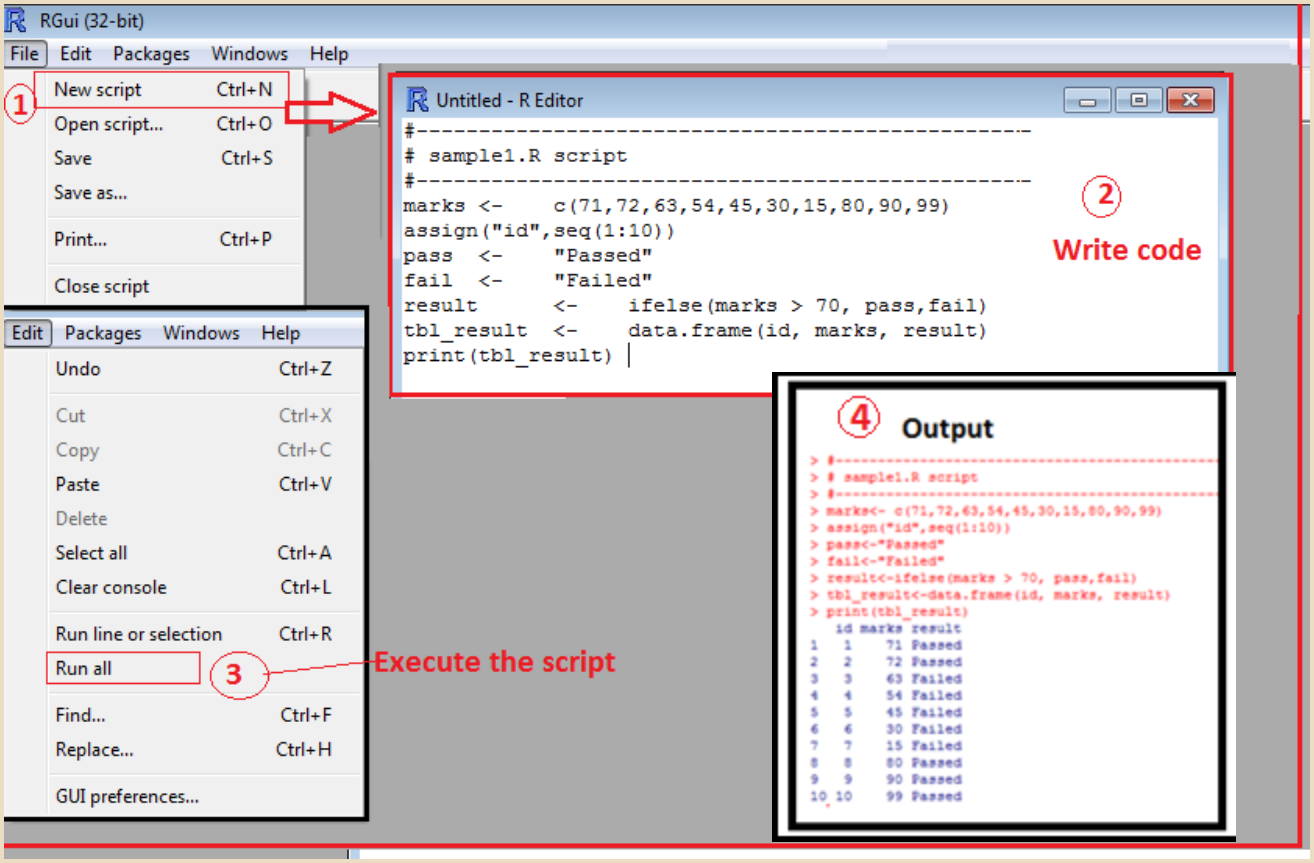
3.8 Working with Projects in RStudio



Highlight the entire code and click on the Run icon

- ***The script is executed in the console panel and the output is displayed.***

3.9 Writing and executing R Scripts




READY FOR A
QUIZ?

1. **Comment lines in R begin with the symbol "#".**
   a) *Always True*
   b) *Always False*
   c) *May or may not be true*
   d) *I do not know the answer*

2. **The command help(mean) does the following:**
   I **It gives extra information about the** *mean* **function.**
   II **It searches the complete documentation for the word** *mean*

   a) *Statement I is True and Statement II is false*
   b) *Statement II is True and Statement I is false*
   c) *Both the statements are False*
   d) *Both the statements are true*

3. **If you are not sure about the name of the function you are looking for, you can perform a fuzzy search with the _____ function.**
   a) *help()*
   b) *apropos()*
   c) *search()*
   d) *None of the above*

4. **The function _____ in *utils* package run all the R code from examples part of R's topic .**
   a) *example()*
   b) *help()*
   c) *All of the above*
   d) *None of the above*

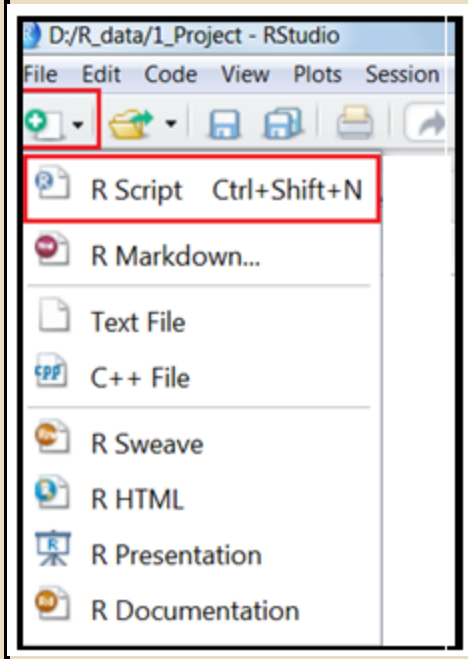
R Programming Course

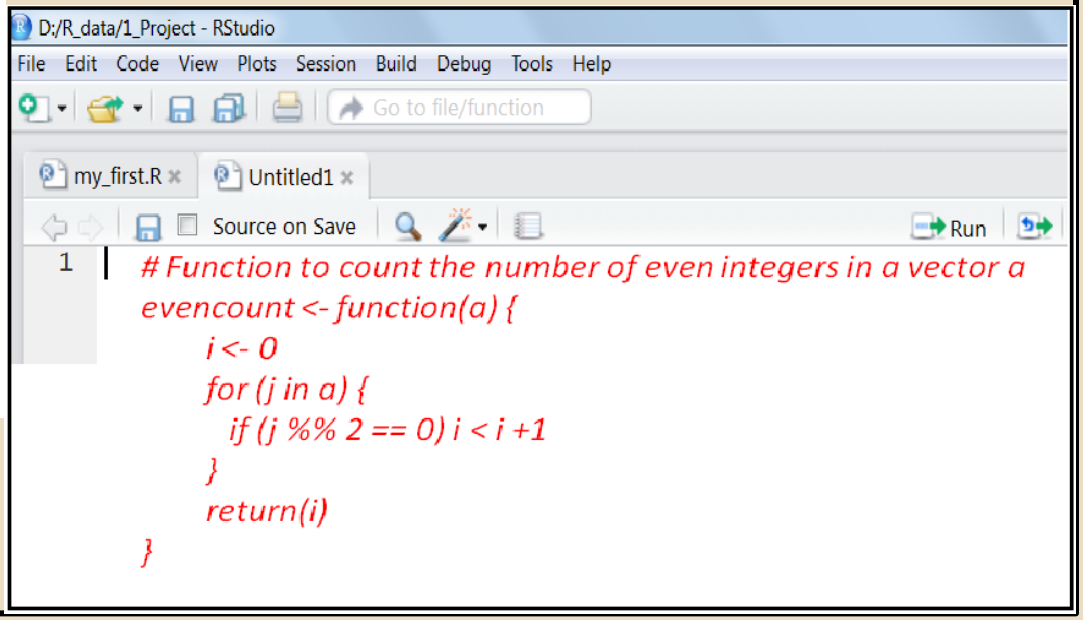
1. **Comment lines in R begin with the symbol "#".**
   a) *Always True*
2. **The command help(mean) does the following:**
   I It gives extra information about the *mean* function.
   II It searches the complete documentation for the word *mean*
   a) *Statement I is True and Statement II is false*
3. **If you are not sure about the name of the function you are looking for, you can perform a fuzzy search with the _____ function.**
   b) *apropos()*
4. **The function _____ in utils package run all the R code from examples part of R's topic .**
   a) *example()*

R Programming Course

Lab Exercise 1:

Write the function in RStudio as shown below:

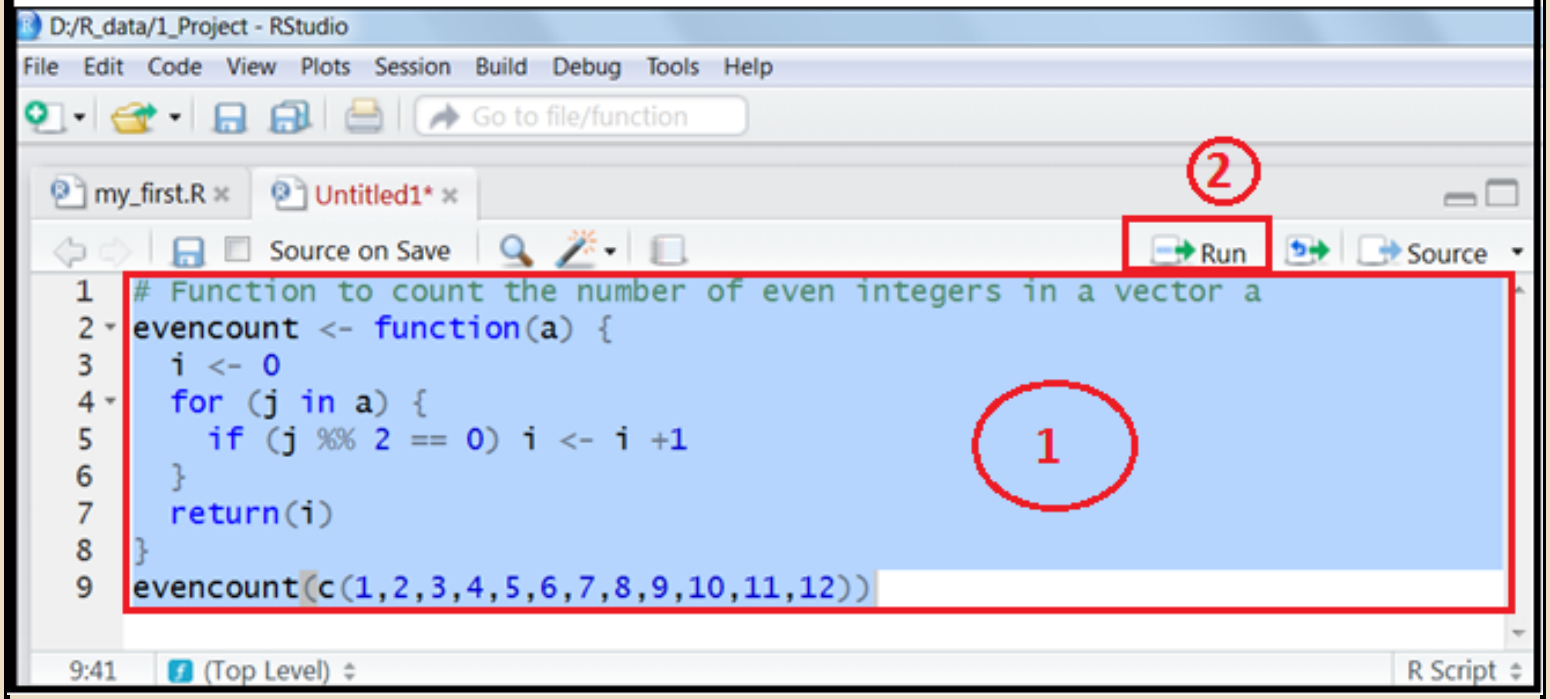




Lab Exercise 2:

Execute the function in RStudio as shown below:

Highlight entire code and click on the Run icon to execute



Check whether you have obtained the results as shown below:

Console D:/R_data/1_Project/

```
> # Function to count the number of even integers in a vector a
> evencount <- function(a) {
+   i <- 0
+   for (j in a) {
+     if (j %% 2 == 0) i <- i +1
+   }
+   return(i)
+ }
> evencount(c(1,2,3,4,5,6,7,8,9,10,11,12))
[1] 6
>
```

Lab Exercise 3:

Save the code as "evencount.R" in RStudio as shown below: