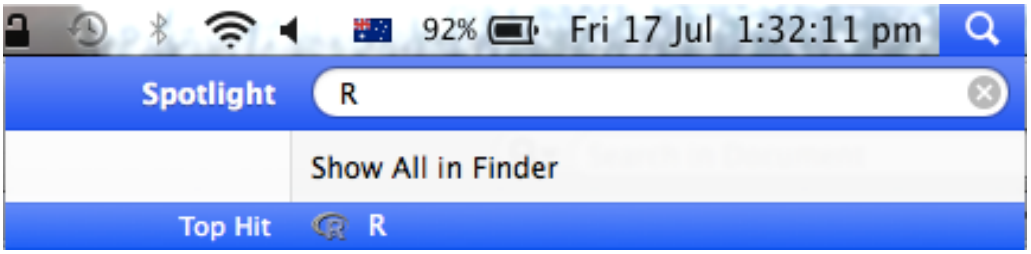
KIT306/606 Tutorial 1

Student ID	Name

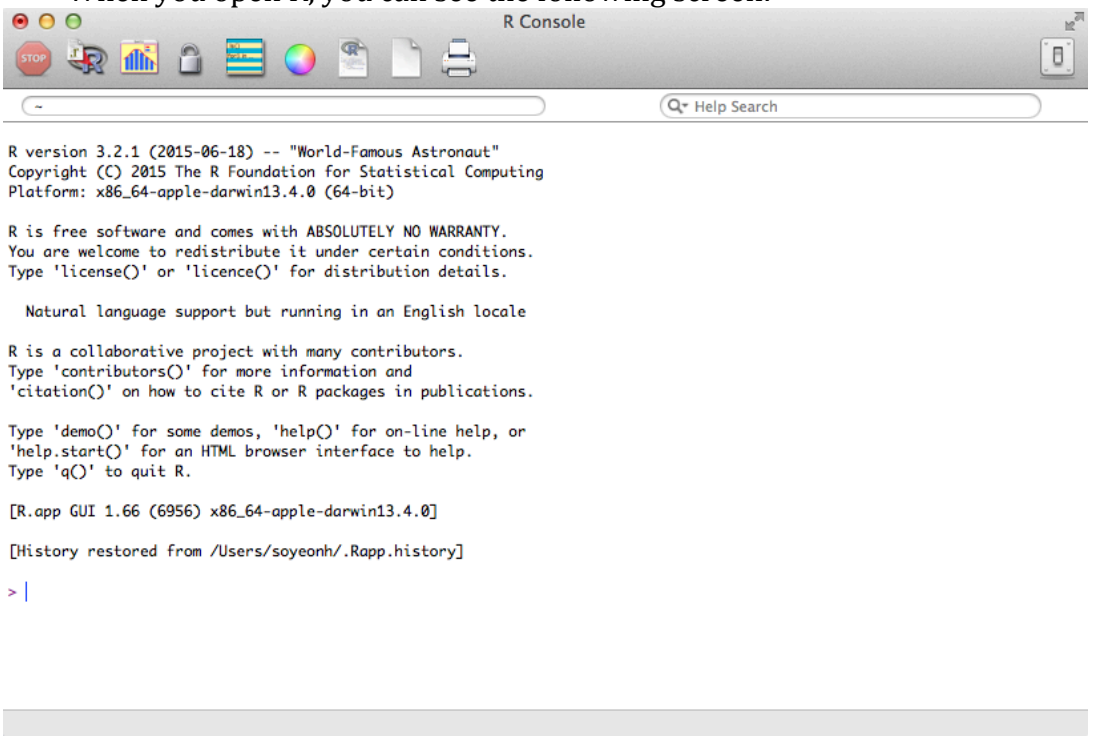
The following tutorial work should be completed by tutorial 2 (week3).

1. Execute R software

- Search R and Open the software



- When you open R, you can see the following screen:



2. Expressions

In R, you should type the input after '> ' sign, and the output can be found after that.

- Let's type 1+2 after '>' sign. You can find [1] 3. **3** is the result, and **[1]** represents the first line of the result.

```
> 1+2
[1] 3
```

- Try the following examples, and check the result

> (12+34-56) * 78/90

> 1 + 2 #plus with space

> 1+2; 3-4

> 1+
+2

>"G'Day, Mate"

> 3^2; 4^2

3. Expressions

- Some expressions return a logical value: TRUE or FALSE.

```
> 7<9
[1] TRUE
> 1+1==3
[1] FALSE
> T==TRUE
[1] TRUE
> F==FALSE
[1] TRUE
```

4. Variable

- R allows you to define a variable with '<-' arrow symbol. Let's try the following examples, and check the result.

```
> x<-2
> y<-3
> x+y
```

```
> x<-2
> x<-3
> x
```

```
> x<-"Hello, World!"
> x
```

- What if you put the number with the variable?

```
> 3A<-1 #error occured
Error: unexpected symbol in "3A"
> A3<-1 #no error?!
> A3
[1] 1
```

- Can we define multiple variables?

```
> a<-b<-10
> a
[1] 10
> b
[1] 10
```

- With ( ), you don't have to call the variable name again

```
> (x<-1:4)
[1] 1 2 3 4
> (x<-1+3/2)
[1] 2.5
```

- You can check the type of variable by using class command.

```
> x=10.5
> x
[1] 10.5
> class(x)
[1] "numeric"
```

5. Functions

- R allows users to use function like the following example.

```
> myfunction <- function(x){
+ return(x)
+ }
```

Type myfunction(1) and check the result.

```
> myfunction2 <- function (x,y){
+ return(x^y)
+ }
```

Type myfunction2(3,2) and check the result.

What if we put myfunction or myfunction2? What type of result can you see?

6. Logical

- A **logical** value is often created via comparison between variables.

```
> x=2; y=3  #sample values
> a = x > y #is x larger than y?
> a
[1] FALSE
```

- Standard logical operations are "&" (and), "|" (or), and "!" (NOT).

7. Condition: If, else

- R allows users to use if-else condition like the following example.

```
> myifelse <- function(x){
+ if(x>0){
+ return (x);
+ }
+ else{
+ return (-x);
+ }
+ }
```

Type myifelse(3) or myifelse(-3) and see how it works.

- Try if-else with the string

```
> mytut<-function(x){
+ if(x>0)
+ {
+ print("good")
+ }
+ else
+ {
+ print("bad")
+ }
+ }
```

8. Loop: For

- R allows users to use for loop like the following example.

```
> myfor <- function(){
+ a<-1
+ for(i in 1:5)
+ {
+ a<-a+1
+ }
+ return (a)
+ }
```

Type myfor() and see how it works.

9. String Operation: Tokenisation

- You can tokenize the sentence and find the word from the sentence.

```
> str1<-strsplit("How are you"," ")
> for(i in str1[[1]]){
+ if(i=="are"){print("yes, we have 'are' in the sentence")}
+ }
[1] "yes, we have 'are' in the sentence"
```

10. Some useful Functions

- **sum function**
  'sum' returns the sum of all the values present in its arguments

```
> sum(1,3,5)
[1] 9
```

- **rep function**
  rep replicates the values in x. It is a generic function, and the (internal) default method is described here.

```
> rep("Hello World", times=3)
[1] "Hello World" "Hello World" "Hello World"
> rep(3, times=2)
[1] 3 3
```

- **abs function**
  abs(x) computes the absolute value of x.

```
> abs(-3)
[1] 3
```

- **sqrt function**
  sqrt(x) computes the (principal) square root of x, $\sqrt{x}$

```
> sqrt(16)
[1] 4
```

11. Some useful Commands

- **help(functionname)**
  help(functionname) brings up help for the given function.
  (e.g. help(sum))

- **example(functionname)**
  example(functionname) brings up examples of usage for the given function
  (e.g. example(sqrt))

- **list.files()**
  You can list the files in the current directory from within R, by calling the list.files function

- **setwd(“R file path”)**
  At all times, R looks in a certain folder (also called 'directory') on the computer. To find which folder R is currently looking in, type **getwd()**. The folder R looks in may be changed to ease access to files. For instance, if R was looking at the Desktop and all R files are kept in a folder on the Desktop called R files, the above command will make R look in that folder.
  When working on a Mac, to go to the desktop, type **setwd("~/Desktop")**To move back or up one folder (for instance, back to the Desktop from a folder on the Desktop), use **setwd(“../”)**

Tutorial Question

1. Fill the box. The box should include 'for' loop to print out the result 15.

```
> x<-0
> for(i in 1:5){
+ 
+ }
> x
[1] 15
```

2. Make a function, named myeven(x). The function sums all even numbers from 1 to x.

3. Make a function, named mystring(x). The function check whether it contains the word hello and world in the sentence x.

- If the sentence x contains both hello and world, the result of the function will be Well done.
- If the sentence x contains one of those words (hello or world), the result of the function will be OK.
- If the sentence x contains none of the word, the result of the function should be Nothing.