Introductory Programming in R

By Asef Nazari

asef.nazari@monash.edu

Faculty of IT

Monash university

1. The First Touch of R

1.1 Using R as a Calculator to Do Arithmetic

You can perform simple arithmetic by inserting numbers and an operation into a cell. Use as much as possible and appropriate paranthesis to make your expresions clear. Use # sign to add comments. Anything typed after the # symbol is ignored by R. Comments are very importanr in documenting programming.

In [47]:

```
# Addition  #comments
3+5
```

8

In [48]:

```
3-5  #subtraction
3*5  # multiplication
3/5  #division
```

-2

15

0.6

In [49]:

```
#Exponentiation could be done in two ways
2**3
2^3
```

8

8

In [50]:

```
#integer division. This is different from 3/5.
3 %/% 5
```

0

In [51]:

```
# pay attention to their difference
5/3
5 %/% 3
```

```
1.66666666666667
```

```
1
```

In [52]:

```
# modulus or remainder
5 %% 3
```

```
2
```

In [53]:

```
3+5/2*5-2^3 ##very hard to understand what is going on!
```

```
7.5
```

In [54]:

```
3+((5/2)*(5-2))^3 # use many paranthesis to clarify what you want.
```

```
424.875
```

1.2 Assignment

A variable in R is a named storage that we can have access through R commands and change its value. A valid variable name consists of letters, numbers, the dot, and underline characters. A variable name strats with a letter or the dot. However, please don't name your variable strating with a dot in this unit! Always strat with a letter.

In R, a variable is created at the same time you assign a value to it. After you created a variable, you can perform manipulations. You can assign values into variables using $< -$ (a greater sign and a hyphen), or $=$ sign. It is recommended to use $< -$, and I am going to use this symbole. It is recommended by experts to reserve $=$ for specifying arguments to functions.

In [55]:

```
x <- 5
x #implicit printing or auto-printing
```

```
5
```

In [56]:

```
print(x) #explicit printing. The differences just because of the setting of R in ju
# we will learn more on the difference and about [1] before the result, when we lea
```

```
[1] 5
```

In [57]:

```
x <- x+3
print(x)
```

```
[1] 8
```

In [58]:

```
y <- 7
z <- x*y
z
```

56

- R is case sensitive for capital letters. Thefore a variable x and X are different.

In [59]:

```
x <- 5
X<- 7
print(x)
print(X)
```

```
[1] 5
[1] 7
```

In [60]:

```
#Scientific notation
2.54e5  #2.54 * 10 ^ 5
7456.3e-2  #7456.3 * 10^(-2)
```

254000

74.563

In [61]:

```
#rounding numbers
2/3
round(2/3,4) #rounds the result of 2/3 into 4  decimal places
```

0.666666666666667

0.6667

In [62]:

```
? round  # to get more information about this function
```

Exercise

Based on Australian Bureau of Statistics (http://www.abs.gov.au/ausstats/abs@.nsf/Lookup/by%20Subject/1370.0.55.001~2011~Main%20Features~Pop Australian population in 2000 was 19.2 millions. If Australian population growth rate is 1.7% per year, what is the prediction of Australian population for 2020? If P_0 is the initial population, r is the annual growth rate, and we are interested to find the population t years later, P_t, we use the following formula

$$P_t = P_0(1 + r)^t$$

1.3 Managing Variables

List of Current Variables

To find the list of exisitng variables in the current environment use ls() or objects() functins.

In [63]:

```
ls()
```

'myVariables' 'x' 'X' 'y' 'z'

In [64]:

```
print(ls()) # Single and double quotes delimit character constants. They can be use
```

```
[1] "myVariables" "x"           "X"           "y"           "z"
```

In [65]:

```
myVariables <- ls()  # assign existing variable to a variables
```

In [66]:

```
print(myVariables)
```

```
[1] "myVariables" "x"           "X"           "y"           "z"
```

In [67]:

```
objects()
```

'myVariables' 'x' 'X' 'y' 'z'

Deleting Variables

You can delete any variale using rm() or remove() functions.

In [68]:

```
ls()
```

'myVariables' 'x' 'X' 'y' 'z'

In [69]:

```
objects()
```

'myVariables' 'x' 'X' 'y' 'z'