Institute of Analytics

The Global Body for Analytics

The Future is Here.

Basics of R

Data sets:
target
inventory

Companies using R

Top Tier Companies using R

The following is a list of top brands or large organizations using R.

1. Facebook – For behavior analysis related to status updates and profile pictures.
2. Google – For advertising effectiveness and economic forecasting.
3. Twitter – For data visualization and semantic clustering
4. Microsoft – Acquired Revolution R company and use it for a variety of purposes.
5. Uber – For statistical analysis
6. Airbnb – Scale data science.
7. IBM – Joined R Consortium Group
8. ANZ – For credit risk modeling
9. HP
10. Ford
11. Novartis
12. Roche
13. New York Times – For data visualization
14. Mckinsey
15. BCG
16. Bain

Companies using R

IT Companies using R

It includes major companies providing IT and professional services using R in India and other parts of the world.

1. Accenture
2. Amadeus IT Group
3. Capgemini
4. Cognizant
5. CSC
6. HCL Technologies
7. Hexaware Technologies
8. HP
9. IBM
10. IGATE
11. Infosys
12. Larsen & Toubro Infotech
13. Microsoft
14. Mindtree
15. Mphasis
16. NIIT Tech
17. Oracle Financial Services Software
18. Paytm
19. Snapdeal
20. R Systems Ltd
21. Tata Consultancy Services
22. Tech Mahindra
23. Wipro

Companies using R

Analytics and Consulting Companies using R

The below list comprises of niche analytics companies as well as consulting companies providing analytics or market research services.

1. A.T. Kearney
2. AbsolutData
3. AC Nielsen
4. Accenture
5. Bain & Company
6. Booz Allen Hamilton
7. Capgemini
8. Convergytics
9. Deloitte Consulting
10. Evalueserve
11. EXL
12. EY
13. Fractal Analytics
14. Gartner
15. Genpact
16. IBM
17. KPMG
18. Latent View
19. Manthan Systems
20. McKinsey & Company
21. Mu Sigma
22. PricewaterhouseCoopers
23. SIBIA Analytics
24. Simplify360
25. SmartCube
26. Target
27. The Boston Consulting Group
28. Tiger Analytics
29. Tower Watson
30. WNS
31. ZS Associate

Companies using R

Financial Institutions

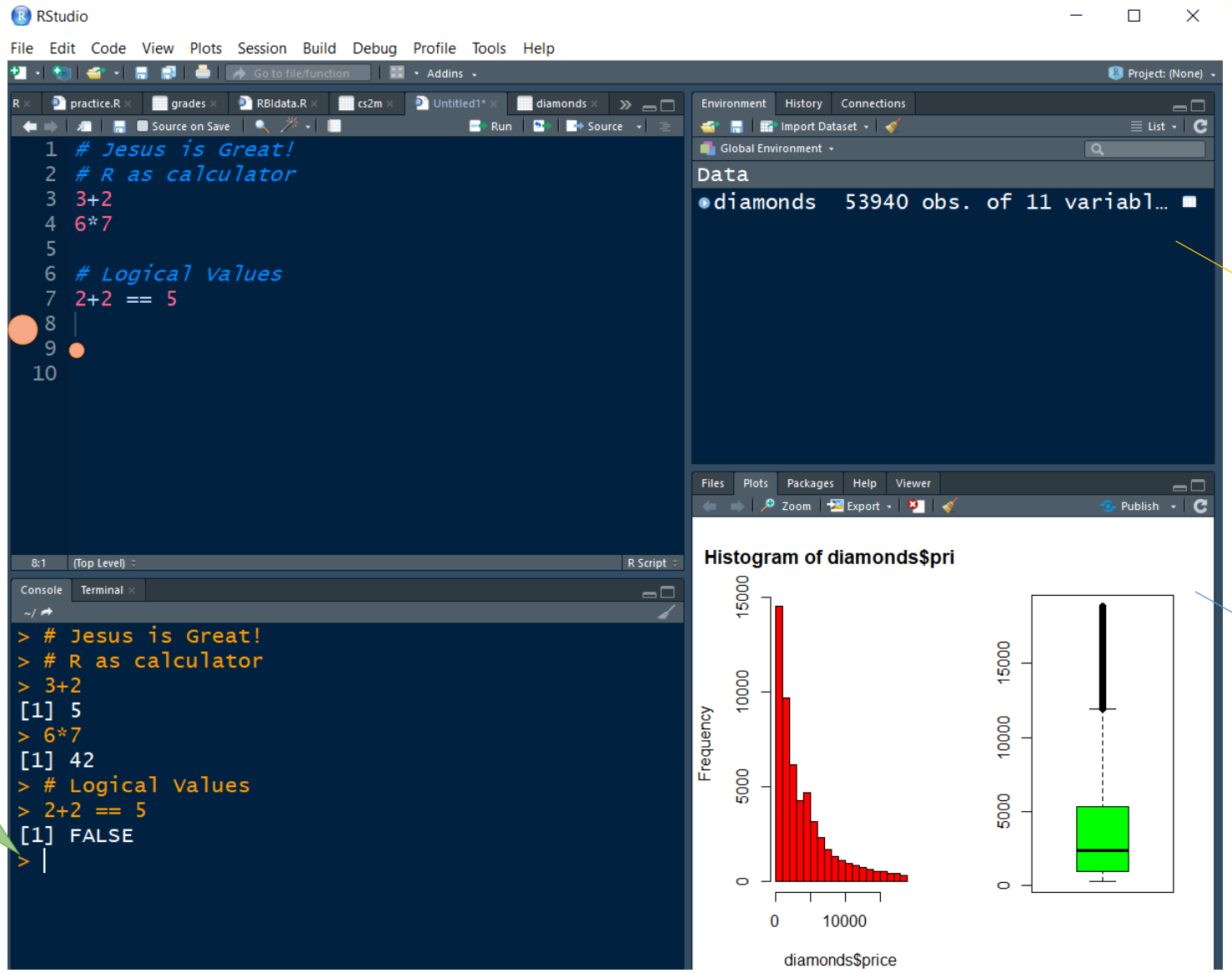
It includes major US and European Banks, Insurance Companies and Other financial institutions using R.

1. American Express
2. ANZ
3. Bank of America
4. Barclays Bank
5. Bazaj allianz Insurance
6. Bharti Axa insurance
7. Blackrock
8. Citibank
9. Dun &Bradstreet
10. Fidelity
11. HSBC
12. JP Morgan
13. KeyBank
14. Lloyds Banking
15. RBS
16. Standard Chartered
17. UBS
18. Wells Fargo
19. Goldman Sachs
20. Morgan Stanley
21. PNC Bank
22. Citizens Bank
23. Fifth Third Bank

R Studio

This is script…ignore time being

This is console. You need to type after yellow/orange > symbol (called cursor prompt) and press ENTER

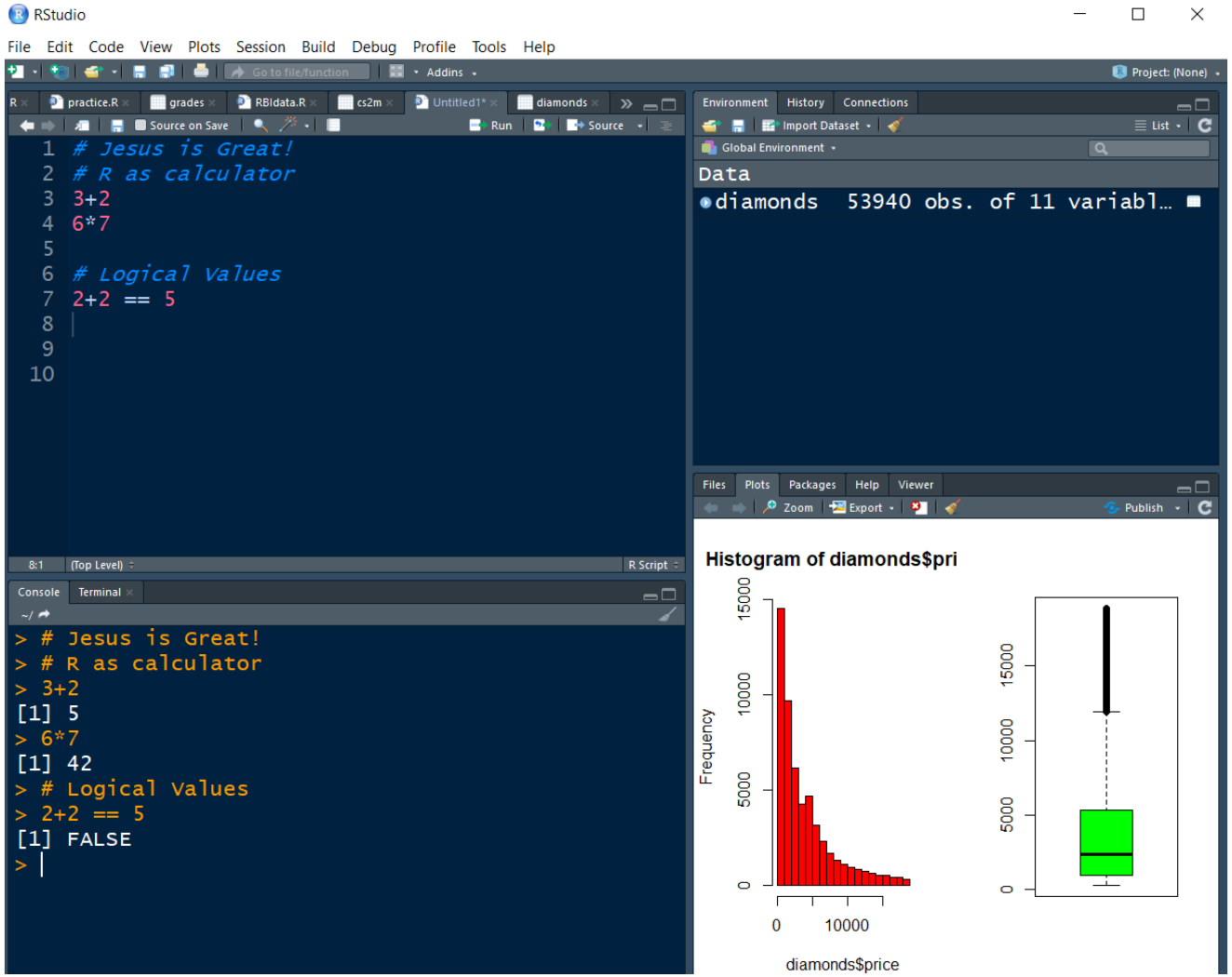


History & Environment comes here

Plots & Diagrams come here

Basics of R

Whatever you will write after #, is not a code (its remark for you)



Dr Vinod on Basics of R 8971073111
vinodanalytics@gmail.com

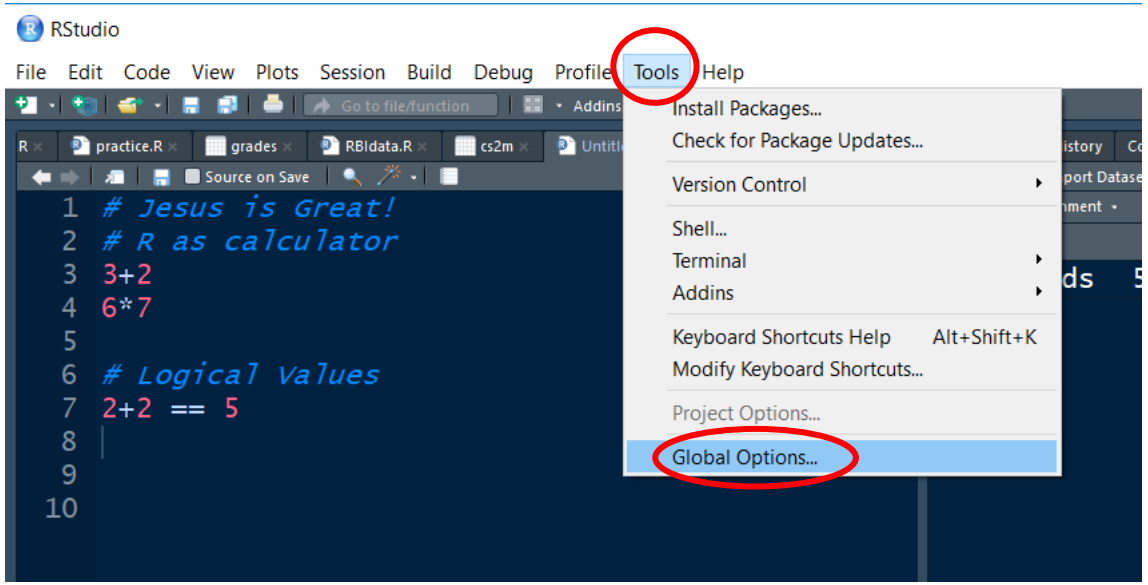
Type exactly the same as you see after > and ENTER

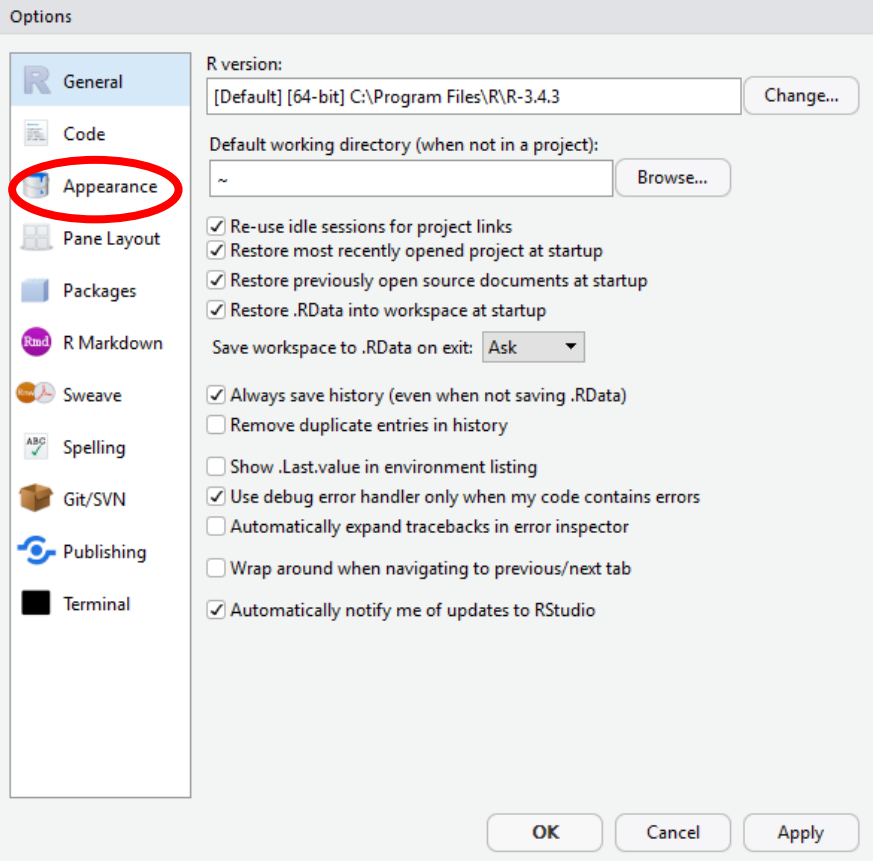

```
> # Jesus is Great!
> # R as calculator
> 3+2
[1] 5
> 6*7
[1] 42
> # Logical Values
> 2+2 == 5
[1] FALSE
> |
```

Play around with simple mathematics like subtract, divide

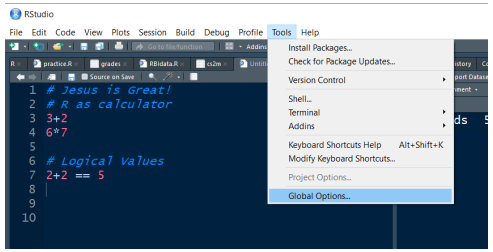
R will say either TRUE or FALSE

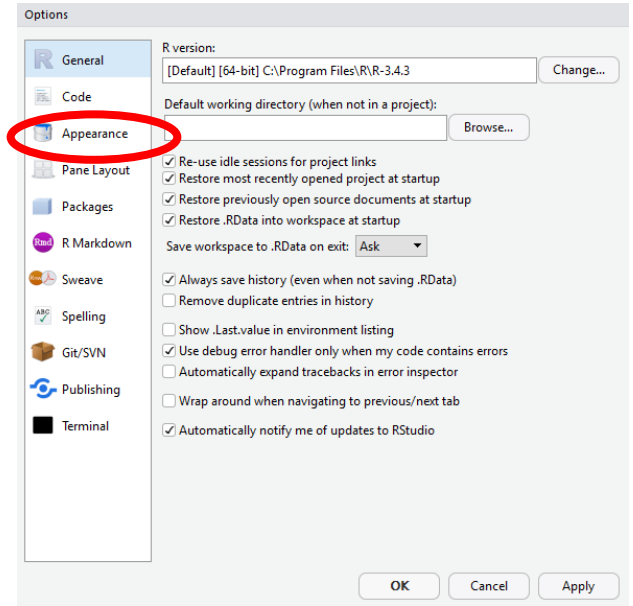
Do You want your screen like my screen?

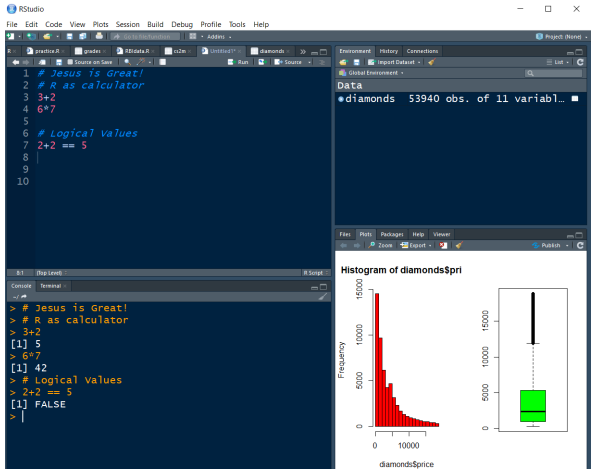


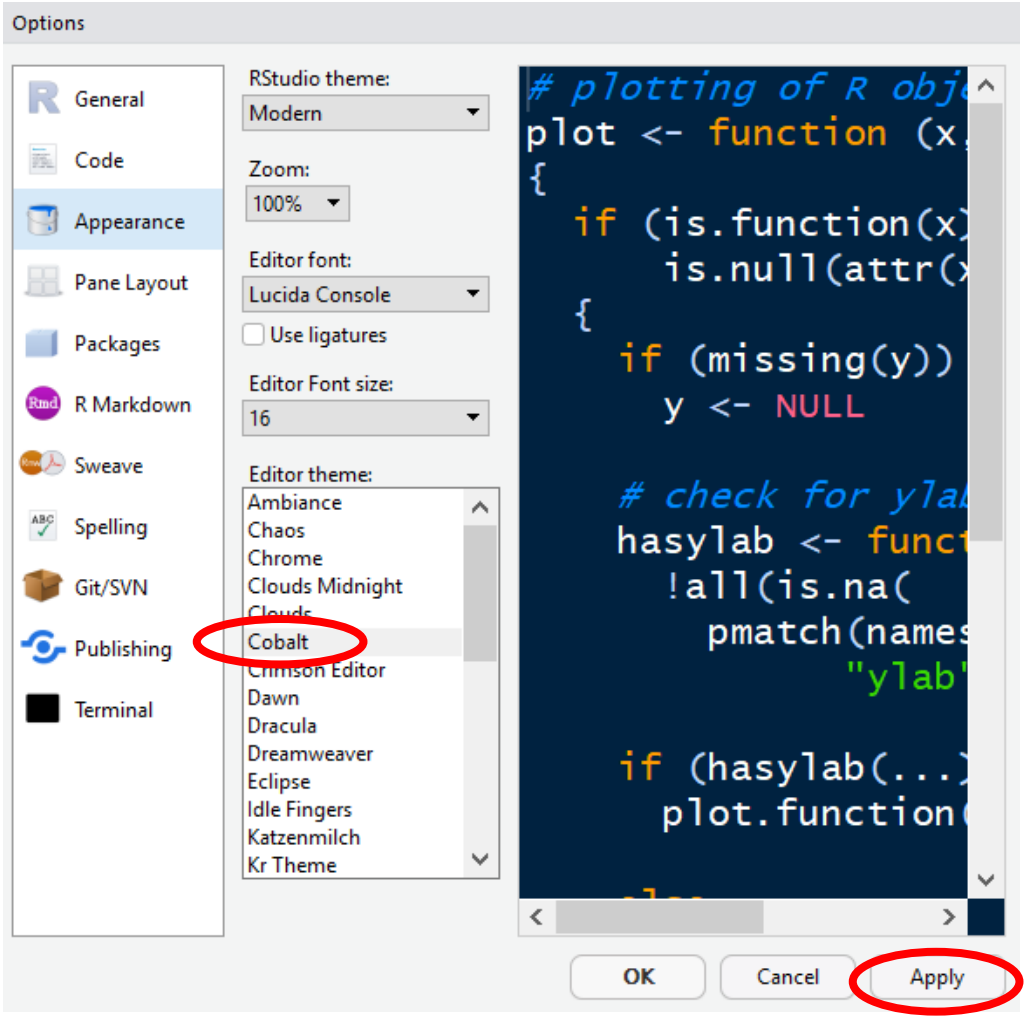


Dr Vinod on Basics of R 8971073111
vinodanalytics@gmail.com









Dr Vinod on Basics of R 8971073111
vinodanalytics@gmail.com

Typing words

```
> Happy Learning
Error: unexpected symbol in "Happy Learning"
> "Happy Learning"
[1] "Happy Learning"
```

Other than numbers, you need to put within inverted commas, single or double….both works

Magic Formula/ Rule

Assign to an object/vector; <- and =

42 is stored in x OR 42 is assigned to x

```
> x<- 42
> x
[1] 42
> x/7
[1] 6
```

If you call x, uncle R will return/show 42

x/7 is 42/7 = 6

100 is stored in y OR 100 is assigned to y

```
> y = 100
> y
[1] 100
> y/5
[1] 20
>
```

If you call y, uncle R will return/show 100

y/5 is 100/5 = 20

Updating assigned value

No more 42 is stored in x

Omg, error!

```
> x<- 'Happy Learning'
> x
[1] "Happy Learning"
> x/7
Error in x/7 : non-numeric argument to binary opera
tor
```

Magic Formula/ Rule

Dr Vinod on Basics of R 8971073111
vinodanalytics@gmail.com

Functions

sum is a function (s should be small)

```
> sum(2, 3, 5)
[1] 10
```

sqrt is a function (s should be small)

```
> sqrt(64)
[1] 8
```

```
> # Repeat a value 3 times
> rep(576, times = 3)
[1] 576 576 576
>
```

rep is a function (r should be small)

Observe **double inverted commas**

```
> rep("M Delighted", 4)
[1] "M Delighted" "M Delighted" "M Delighted"
[4] "M Delighted"
```

Vectors

c stands for concatenation

```
> q<- c('a', 'b', 'c')
> q
[1] "a" "b" "c"
```

Why are you writing names within inverted commas?

```
> w<- c('reena', 'teena', 'meena')
> w
[1] "reena" "teena" "meena"
```

```
> g<- 5:9
> g
[1] 5 6 7 8 9
> h<- c(5:9)
> h
[1] 5 6 7 8 9
> i<- seq(5,9)
> i
[1] 5 6 7 8 9
```

Same result with three styles!

Change in increment & direction

Increment of 0.5

```
[1] 5 6 7 8 9
> j <- seq(5,9,0.5)
> j
[1] 5.0 5.5 6.0 6.5 7.0 7.5 8.0 8.5 9.0
> k = seq(9,5, -0.5)
> k
[1] 9.0 8.5 8.0 7.5 7.0 6.5 6.0 5.5 5.0
```

Decrement of 0.5

Vector access

Create a vector 'learn' and call it

```
> learn<- c('you', 'me', 'R')
> learn
[1] "you" "me"  "R"
> learn[2]
[1] "me"
> learn[3]
[1] "R"
> learn[1]
[1] "you"
```

What is there at 2nd, 3rd and 1st position/location?

Whenever you see a rectangle bracket, it means Sub-setting

Replacing element

Let **SaS** appear at 3rd place

```
> learn[3] <- 'SaS'
> learn
[1] "you" "me"  "SaS"
```

Call **learn**

Add element

Lets add **SPSS** at 4th position

```
> learn[4]<- 'SPSS'
> learn
[1] "you"  "me"   "SaS"  "SPSS"
```

Vector addition

```
> a<- c(1,2,3)
> a
[1] 1 2 3
> # add 1 to a
> a+1
[1] 2 3 4
```

1 is added to each element of **a** [1+1 = **2**, 2+1 = **3**, 3+1 = **4**]

Try a/2 and a*2

Vector Math

```
> a<- c(1,2,3)
> a
[1] 1 2 3
```

```
> b<- c(4,5,6)
> b
[1] 4 5 6
> a+b
[1] 5 7 9
> a-b
[1] -3 -3 -3
> a*b
[1]  4 10 18
```

a + b=

1+4 = **5**; 2+5 = **7**; 3+6 = **9**

a-b=

1-4 = -**3**; 2-5 = **-3**; 3-6 = -**3**

a*b=

1*4 = **4**; 2*5 = **10**; 3*6 = **18**

Compare elements

Single equal to = means ASSIGNMENT

```
> a<- c(1,2,3)
> a
[1] 1 2 3
```

Double equal to, == means EQUIVALENCE

```
> a == c(1,99,3)
[1]  TRUE FALSE  TRUE
>
```

We are asking from Uncle R that whether 1 is at first position, 99 is at second position and 3 is at third position

Uncle R replies: **Yes,** it is **TRUE** that 1 is at *first* position; **No**, 99 is not at second position…in uncle R's language **FALSE**; **Yes**, it is **TRUE** that 3 is at *third* position

How to sum with missing values?

m is having 5 elements, 4 digits as 1, 3, 7 & 9 and one MISSING value (Uncle R reads as **NA**)

```
> m<- c(1, 3, NA, 7, 9)
> m
[1]  1  3 NA  7  9
> sum(m)
[1] NA
> sum(m, na.rm = TRUE)
[1] 20
```

You tried summing **m** but Uncle R has thrown NA (Cant do!)

If you write **na.rm = TRUE** (Na's to be removed in calculation, Yes!) then you will get 20

Lets create Matrix

matrix is a function

1st number will be **elements** in matrix, 2nd number is number of **ROWS** and 3rd number is number of **COLUMNS**

```
> matrix(0,3,4)
     [,1] [,2] [,3] [,4]
[1,]    0    0    0    0
[2,]    0    0    0    0
[3,]    0    0    0    0
>
```

Uncle R will first fill first column then next column…and so on!

Dr Vinod on Basics of R 8971073111
vinodanalytics@gmail.com

Matrix creation

Storing 1 to 8 numbers in a vector/object name **sijo**

```
> sijo<- 1:8
> sijo
[1] 1 2 3 4 5 6 7 8
```

calling **sijo**

Creating a matrix of 2 rows and 4 columns name **s** with elements stored in **sijo**

```
> s<- matrix(sijo, c(2,4))
> s
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

Column identification **[,c]**

Row identification **[r,]**

Matrix Access

2nd Row, ALL columns

```
> s[2, ]
[1] 2 4 6 8
```

4th column, ALL rows

```
> s[, 4]
[1] 7 8
```

```
> s<- matrix(sijo, c(2,4))
> s
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

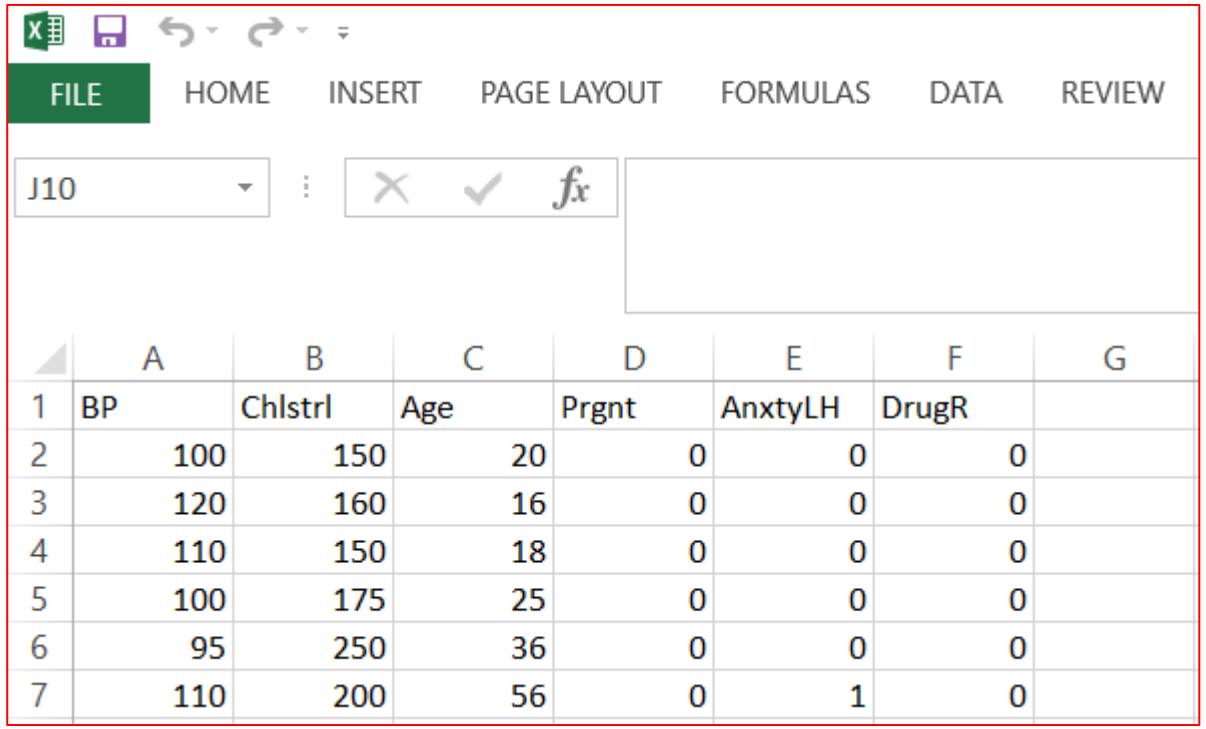
2nd to 4th column, ALL rows

```
> s[, 2:4]
     [,1] [,2] [,3]
[1,]    3    5    7
[2,]    4    6    8
```

```
> s[2,4]
[1] 8
```

2nd row & 4th column

Data Frame



	A	B	C	D	E	F	G
1	BP	Chlstrl	Age	Prgnt	AnxtyLH	DrugR	
2	100	150	20	0	0	0	
3	120	160	16	0	0	0	
4	110	150	18	0	0	0	
5	100	175	25	0	0	0	
6	95	250	36	0	0	0	
7	110	200	56	0	1	0	

We will create a Data Frame like this!

	A	B	C
1	**names**	**percent**	**lunch**
2	joel	85	biryani
3	chris	88	chicken kabab
4	julie	92	biryani
5	mary	95	chicken kabab
6	sprina	89	veg pulao

Variable Types

Numbers with decimals are NUMERIC and without decimal are INTEGERS

Numeric or Integer

	A	B	C
1	**names**	**percent**	**lunch**
2	joel	85	biryani
3	chris	88	chicken kabab
4	julie	92	biryani
5	mary	95	chicken kabab
6	sprina	89	veg pulao

Categorical/Factor

Character

Create Variables/Vectors

```
> names<- c('joel', 'chris', 'julie', 'mary', 'sprina')
> names
[1] "joel"   "chris"  "julie"  "mary"
[5] "sprina"
```

```
> percent
[1] 85 88 92 95 89
>
```

	A	B	C
1	**names**	**percent**	**lunch**
2	joel	85	biryani
3	chris	88	chicken kabab
4	julie	92	biryani
5	mary	95	chicken kabab
6	sprina	89	veg pulao

```
> lunch<- c('biryani', 'chicken kabab', 'biryani', 'chicken kabab', 'veg pulao')
> lunch
[1] "biryani"       "chicken kabab" "biryani"
[4] "chicken kabab" "veg pulao"
```

Structure of lunch

```
> lunch<- c('biryani', 'chicken kabab', 'biryani', 'chicken kabab', 'veg pulao')
> lunch
[1] "biryani"       "chicken kabab" "biryani"
[4] "chicken kabab" "veg pulao"
```

str is a function

```
> str(lunch)
 chr [1:5] "biryani" "chicken kabab" "biryani" ...
```

Type is character (**chr**), 1 to 5 observations, sample observations

	A	B	C
1	**names**	**percent**	**lunch**
2	joel	85	biryani
3	chris	88	chicken kabab
4	julie	92	biryani
5	mary	95	chicken kabab
6	sprina	89	veg pulao

yummy is the new name of **lunch**

as.factor is a function

A factor variable is shown with Levels (these are mentioned alphabetically)

Conversion to factor

```
> yummy<- as.factor(lunch)
> yummy
[1] biryani       chicken kabab biryani       chicken kabab
[5] veg pulao
Levels: biryani chicken kabab veg pulao
> str(yummy)
 Factor w/ 3 levels "biryani","chicken kabab",..: 1 2 1 2 3
```

Structure is showing 3 **levels** (with 3 levels)

Age = 21, 23, 24, 25….40k
Gender = M, F, F, M…..40k
Str(Gender) : chr
Gender as FACTOR
Region = 1, 2, 3, 4

	A	B	C
1	**names**	**percent**	**lunch**
2	joel	85	biryani
3	chris	88	chicken kabab
4	julie	92	biryani
5	mary	95	chicken kabab
6	sprina	89	veg pulao

Create Data Frame

joy is the name of data frame

data.frame is a function

Name of variables within **data.frame** (joy)

```
> joy<- data.frame(names, percent, yummy)
> joy
   names percent         yummy
1   joel      85       biryani
2  chris      88 chicken kabab
3  julie      92       biryani
4   mary      95 chicken kabab
5 sprina      89     veg pulao
```

Access Data Frame

Type **$** after data frame name (**joy**) and you will see all variable names stored in that data frame

```
> joy$
    names
    percent
    yummy
```

Select name and press tab key, it will appear after **$** [try with **yummy**]

```
> joy$yummy
[1] biryani       chicken kabab biryani       chicken kabab
[5] veg pulao
Levels: biryani chicken kabab veg pulao
```

Access Data Frame

Names are being considered as factor

```
> joy$names
[1] joel   chris  julie  mary   sprina
Levels: chris joel julie mary sprina
> joy$percent
[1] 85 88 92 95 89
```

This is numeric

You can change these to character by using **as.character** [we will discuss this later]

Access Data Frame

See double rectangle bracket & variable number (3rd is **yummy**)

```
> joy
   names percent         yummy
1   joel      85        biryani
2  chris      88 chicken kabab
3  julie      92        biryani
4   mary      95 chicken kabab
5 sprina      89      veg pulao
```

```
> joy[[3]]
[1] biryani        chicken kabab biryani        chicken kabab
[5] veg pulao
Levels: biryani chicken kabab veg pulao
```

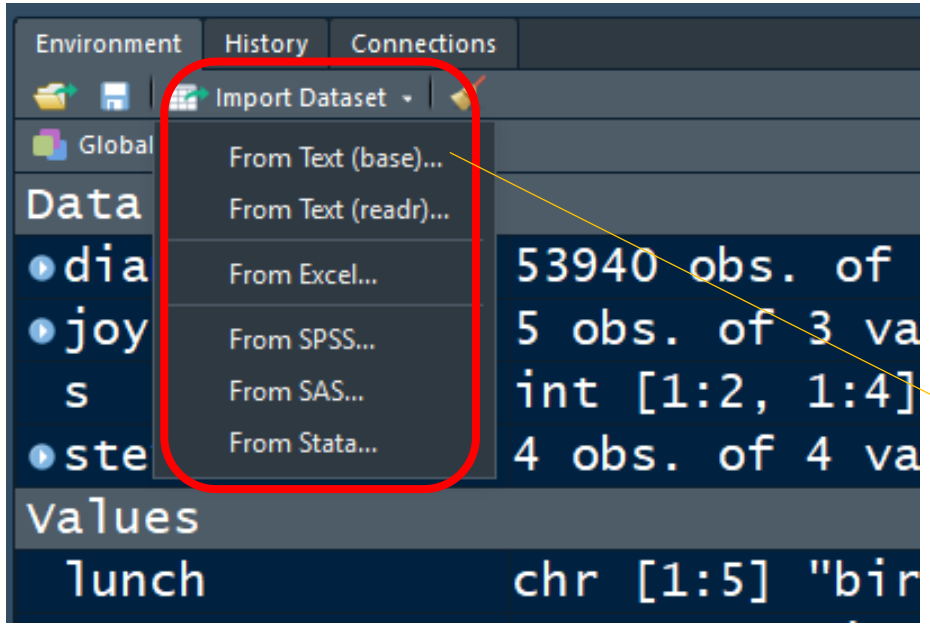
Try with 1 and 2

Access Data Frame

You can write variable name within double rectangle bracket like **"yummy"** (between inverted commas (single or double)

```
> joy[["yummy"]]
[1] biryani       chicken kabab biryani       chicken kabab
[5] veg pulao
Levels: biryani chicken kabab veg pulao
```

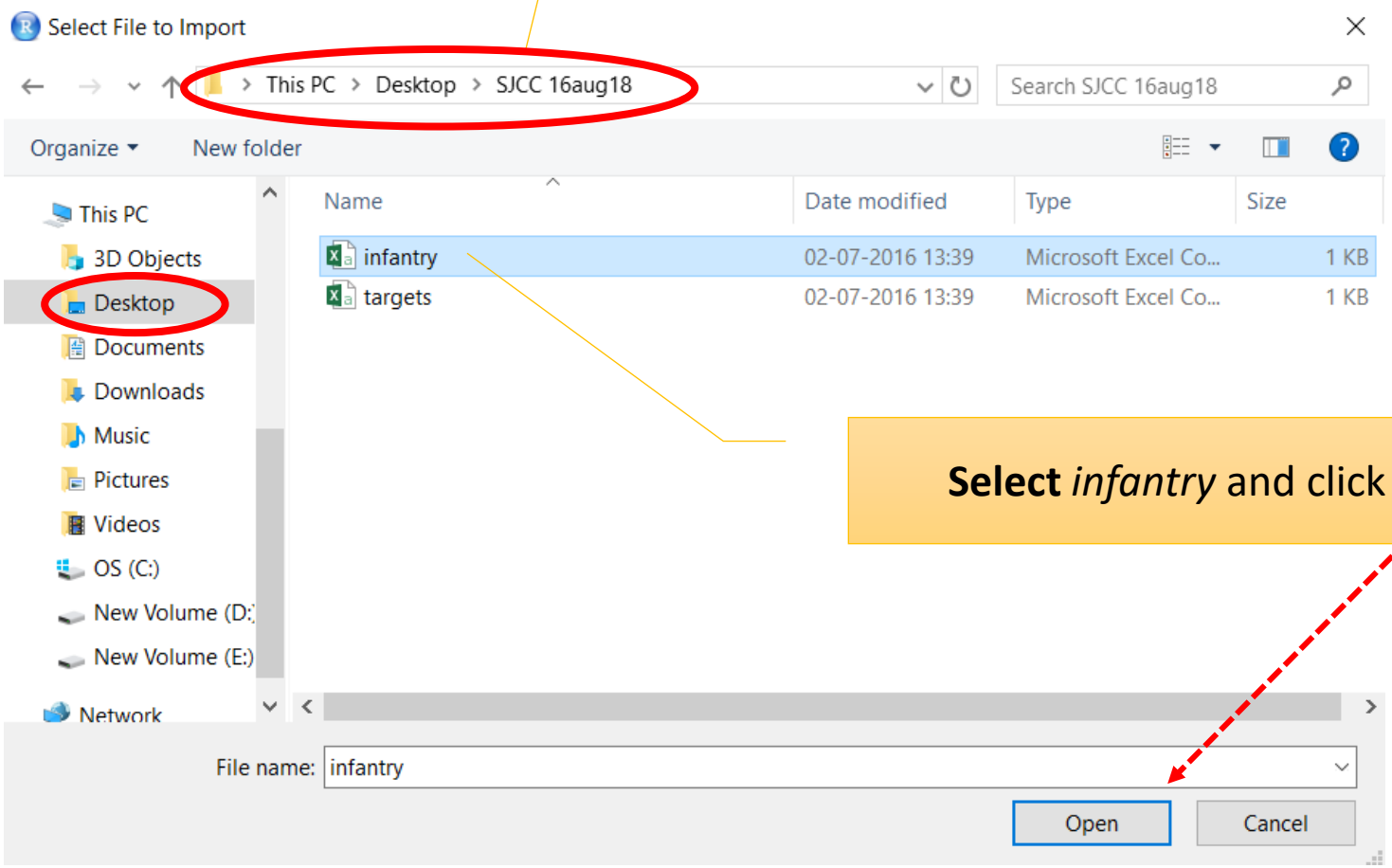
Import Data Frame



Click **From Text(base)**

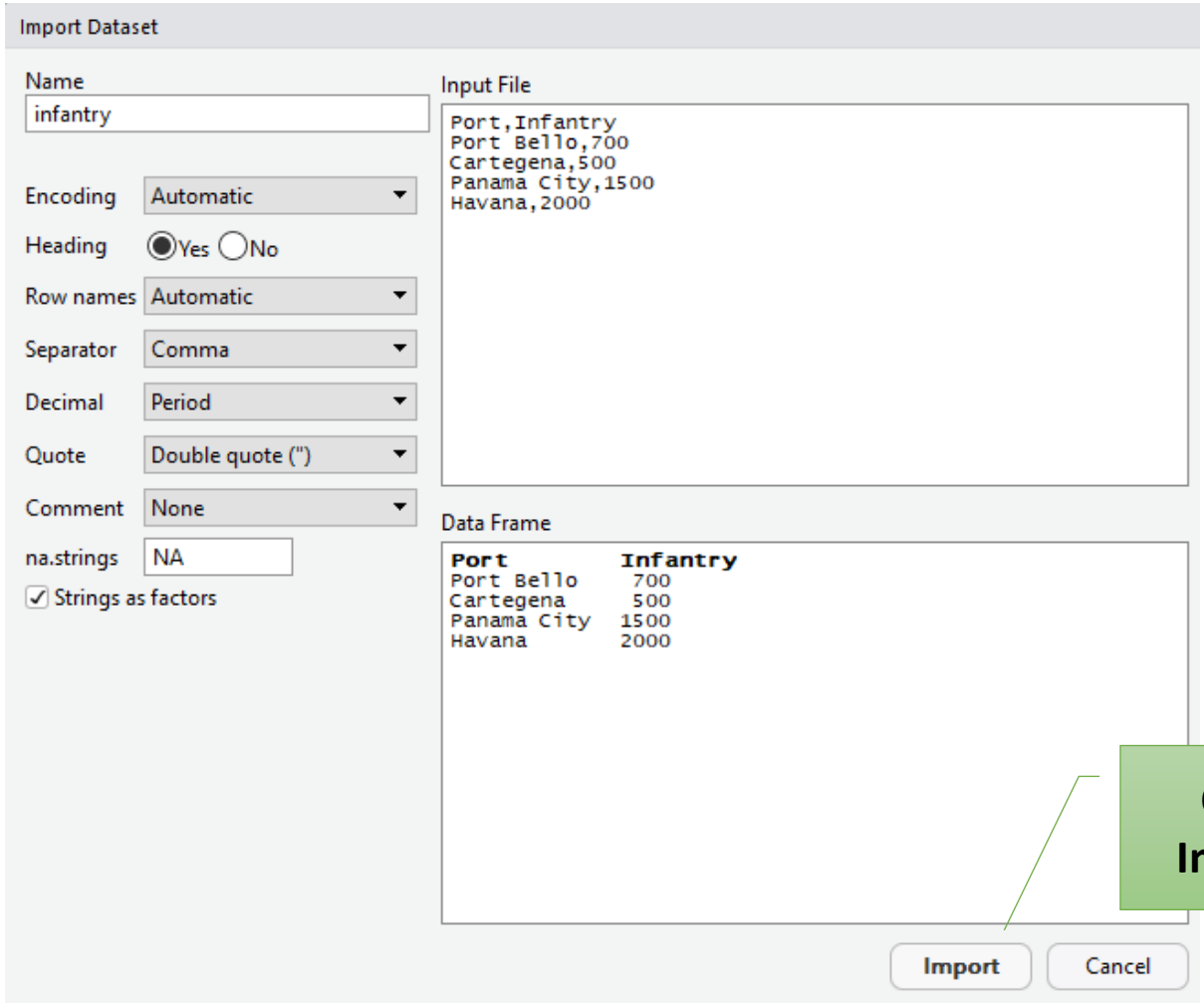
Import

Go to the folder where you have kept these files



Select *infantry* and click **Open**

Yes should be checked!



Click **Import**

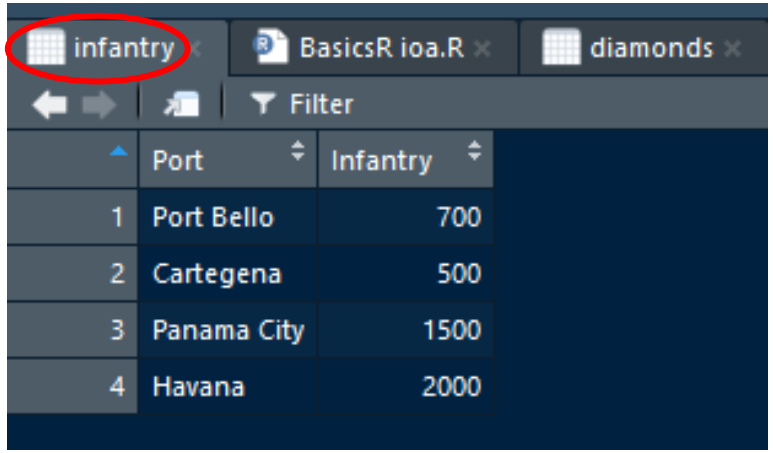
Command for reading file

Path of file

Name of file in R

```
> infantry <- read.csv("C:/Users/Dr Vinod/Desktop/SJCC 16aug18/infantry.csv")
>   View(infantry)
```

You can see file in left top quadrant



	Port	Infantry
1	Port Bello	700
2	Cartegena	500
3	Panama City	1500
4	Havana	2000

Import targets

```
> targets <- read.csv("C:/Users/Dr Vinod/Desktop/SJCC 16aug18/targets.csv")
>   View(targets)
```

infantry | targets | BasicsR ioa.R | diamonds

Filter

	Port	Population	Worth
1	Cartegena	35000	10000
2	Port Bello	49000	15000
3	Havana	140000	50000
4	Panama City	105000	35000

Can we merge these two files?

infantry | BasicsR ioa.R | diamonds

Filter

	Port	Infantry
1	Port Bello	700
2	Cartegena	500
3	Panama City	1500
4	Havana	2000

infantry | targets | BasicsR ioa.R | diamonds

Filter

	Port	Population	Worth
1	Cartegena	35000	10000
2	Port Bello	49000	15000
3	Havana	140000	50000
4	Panama City	105000	35000

Merge

```
> steve<- merge(x = infantry, y = targets)
> steve
         Port Infantry Population Worth
1   Cartegena      500      35000 10000
2      Havana     2000     140000 50000
3 Panama City     1500     105000 35000
4  Port Bello      700      49000 15000
```

```
View(steve)
```

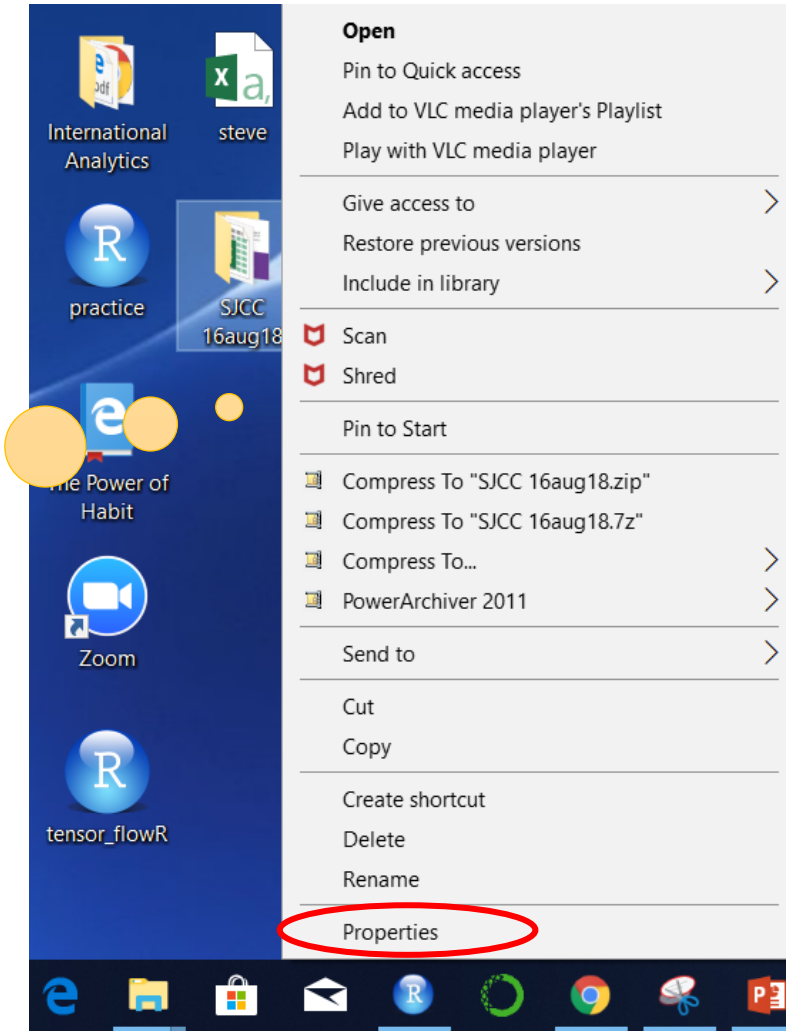
infantry | targets | BasicsR ioa.R* | steve

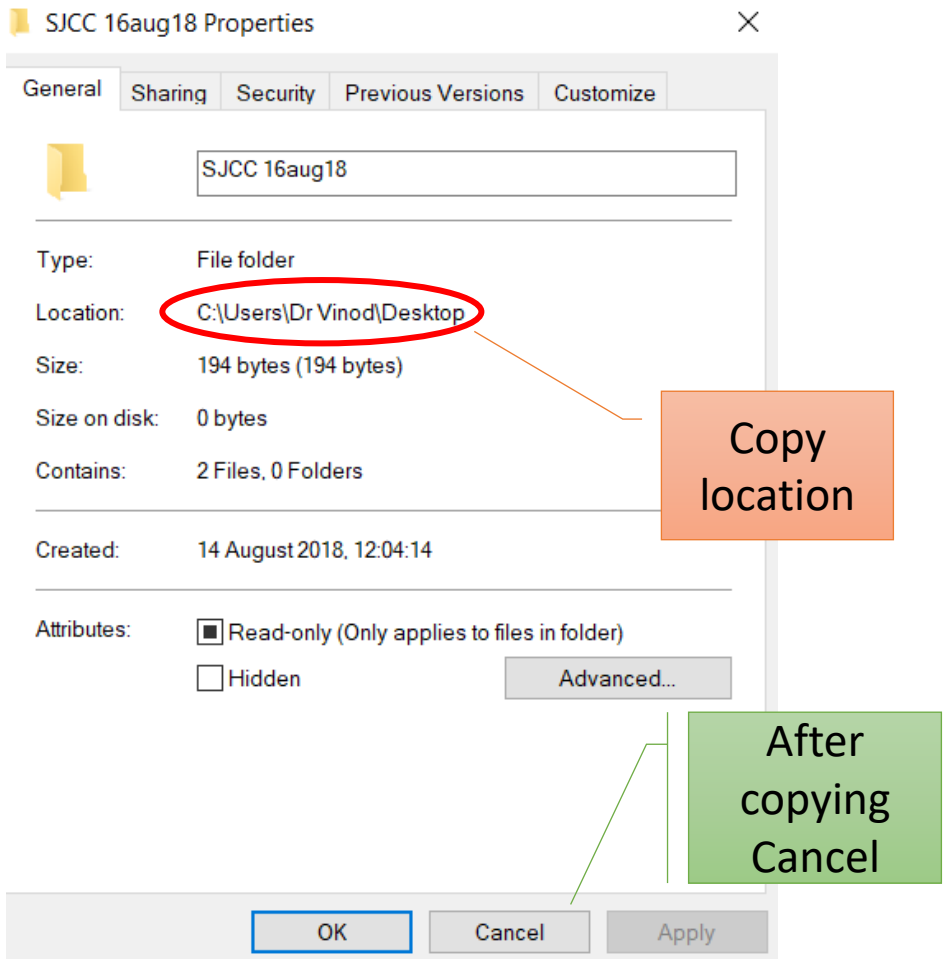
Filter

	Port	Infantry	Population	Worth
1	Cartegena	500	35000	10000
2	Havana	2000	140000	50000
3	Panama City	1500	105000	35000
4	Port Bello	700	49000	15000

Export

Right Click any folder/file on Desktop and then click **Properties**





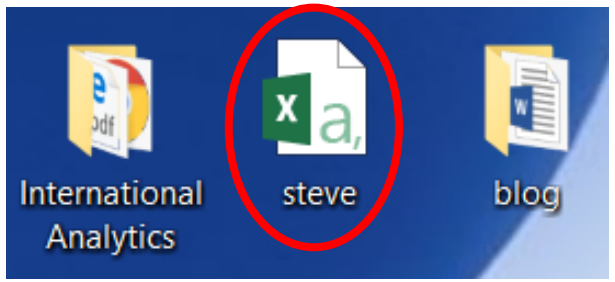
Export

Location

After pasting location, you need to make slashes as shown

```
write.csv(steve,"C:/Users/Dr Vinod/Desktop/steve.csv")
```

Name of file in R Environment



Name of file going to be acquired after export to location


Well done!


THANK YOU
Shift


GOD
bless
YOU