Lecture #1

A: **RStudio**

Objectives

1: Know how to use and navigate RStudio Integrated Development Environment (IDE)

   a. Navigate the IDE
   b. Learn some commonly used keyboard shortcuts
   c. A look at the complete cheat sheep for future reference

2: Why we should know how to use the IDE

B: **R Software**

Objectives

1: Use R as a calculator

2: Use Washington post article to describe the following concepts:

   a. an observation: the process of watching someone or something, informal or formal, and involves some form of data collection (e.g., a chart showing data collected over a period of time or a chart showing who is raising US-born children)

   b. variables: quantity capable of assuming any of a set of values, such as x in the expression x +1 or the assignment y = x

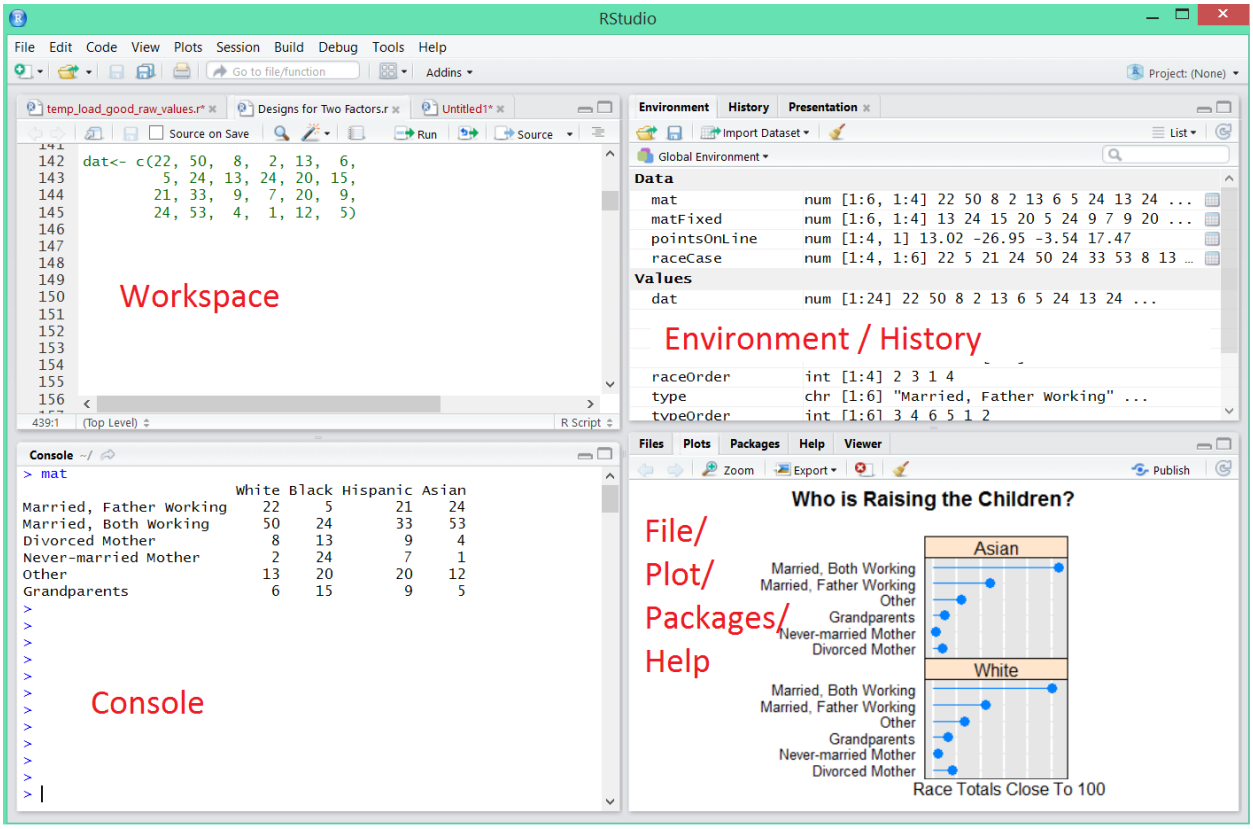
3: R data types: vectors, arrays, lists, matrices, data frame and factors

4: Write simple and save R programs (scripts) to be modified for future use.

5: In class lab exercise

6: Homework 1

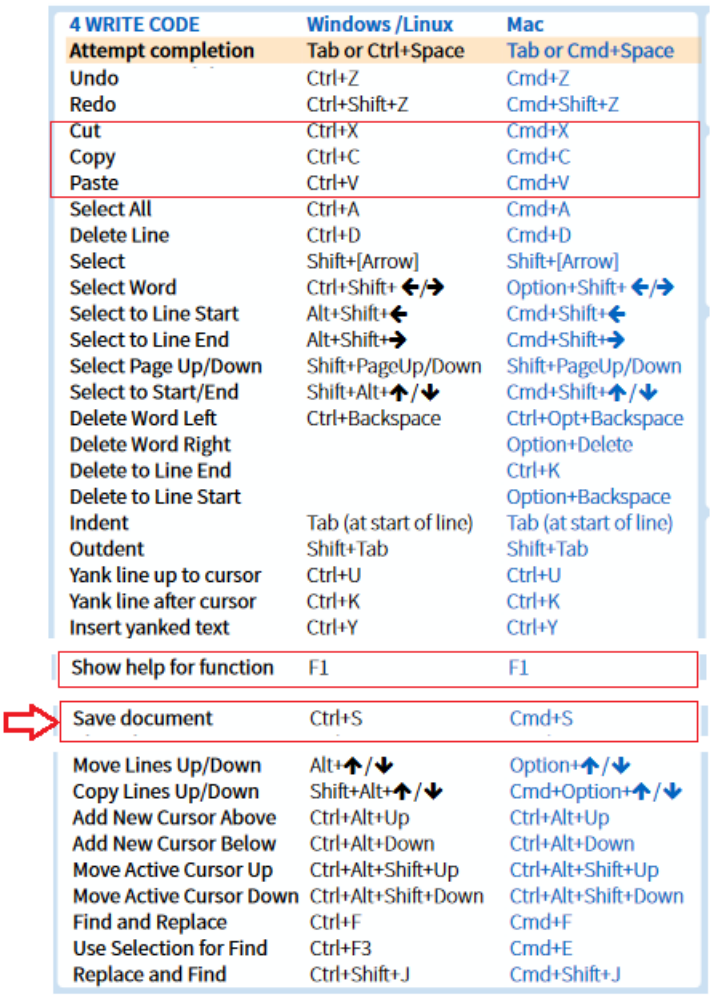
A: **RStudio**


RStudio
File Edit Code View Plots Session Build Debug Tools Help
Workspace
Environment / History
Console
File/
Plot/
Packages/
Help
Who is Raising the Children?
Asian
White
Married, Both Working
Married, Father Working
Other
Grandparents
Never-married Mother
Divorced Mother
Race Totals Close To 100

Keyboard short cuts

1 LAYOUT	Windows/Linux	Mac
Move focus to Source Editor	Ctrl+1	Ctrl+1
Move focus to Console	Ctrl+2	Ctrl+2
Move focus to Help	Ctrl+3	Ctrl+3
Show History	Ctrl+4	Ctrl+4
Show Files	Ctrl+5	Ctrl+5
Show Plots	Ctrl+6	Ctrl+6
Show Packages	Ctrl+7	Ctrl+7
Show Environment	Ctrl+8	Ctrl+8

2 RUN CODE	Windows/Linux	Mac
Search command history	Ctrl+↑	Cmd+↑
Navigate command history	↑/↓	↑/↓
Move cursor to start of line	Home	Cmd+←
Move cursor to end of line	End	Cmd+→
Change working directory	Ctrl+Shift+H	Ctrl+Shift+H
Interrupt current command	Esc	Esc
Clear console	Ctrl+L	Ctrl+L
Quit Session (desktop only)	Ctrl+Q	Cmd+Q
Restart R Session	Ctrl+Shift+F10	Cmd+Shift+F10
Run current line/selection	Ctrl+Enter	Cmd+Enter
Run current (retain cursor)	Alt+Enter	Option+Enter
Run from current to end	Ctrl+Alt+E	Cmd+Option+E
Run the current function	Ctrl+Alt+F	Cmd+Option+F
Source a file	Ctrl+Shift+O	Cmd+Shift+O
Source the current file	Ctrl+Shift+S	Cmd+Shift+S
Source with echo	Ctrl+Shift+Enter	Cmd+Shift+Enter



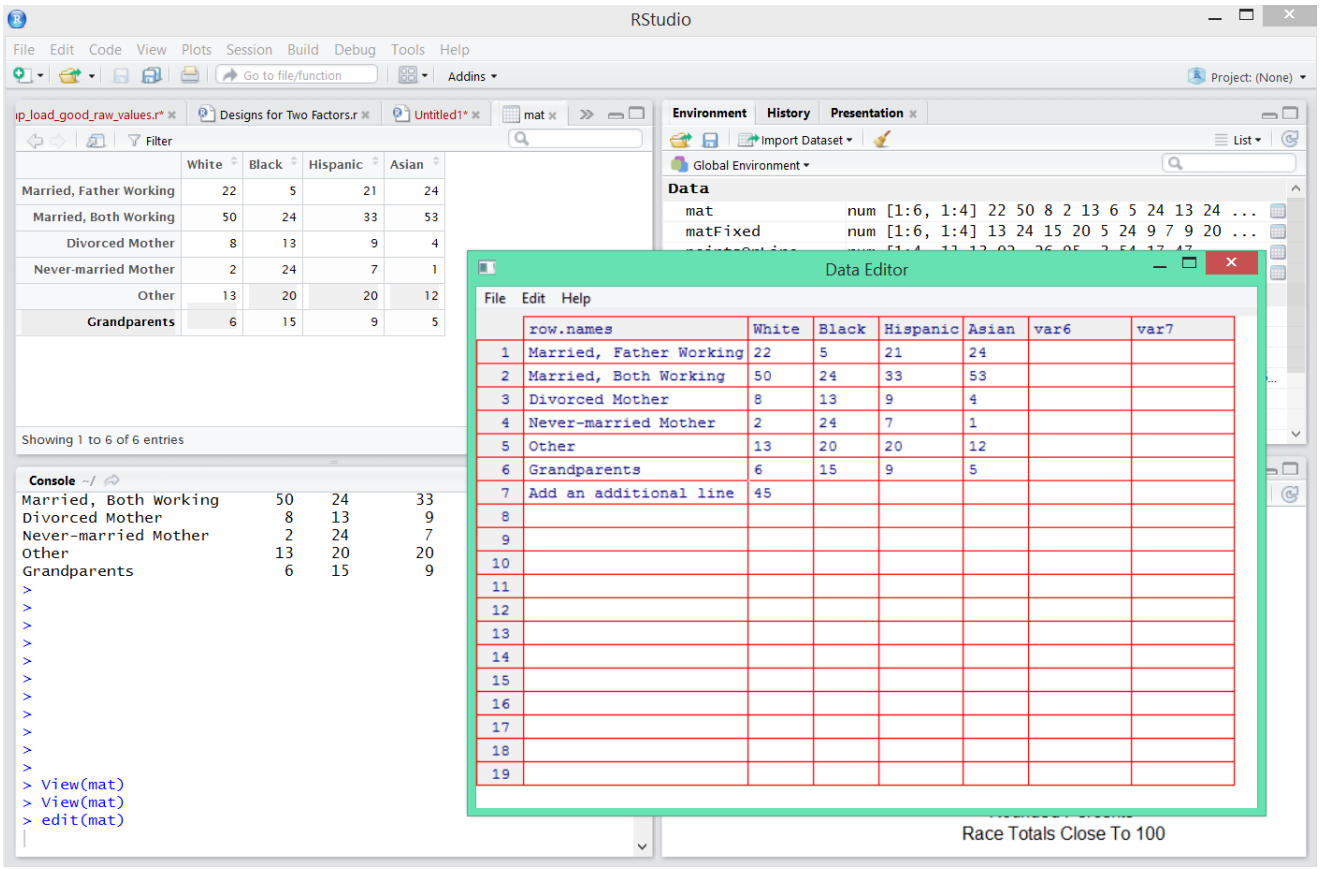
4 WRITE CODE	Windows /Linux	Mac
Attempt completion	Tab or Ctrl+Space	Tab or Cmd+Space
Undo	Ctrl+Z	Cmd+Z
Redo	Ctrl+Shift+Z	Cmd+Shift+Z
Cut	Ctrl+X	Cmd+X
Copy	Ctrl+C	Cmd+C
Paste	Ctrl+V	Cmd+V
Select All	Ctrl+A	Cmd+A
Delete Line	Ctrl+D	Cmd+D
Select	Shift+[Arrow]	Shift+[Arrow]
Select Word	Ctrl+Shift+ ←/→	Option+Shift+ ←/→
Select to Line Start	Alt+Shift+←	Cmd+Shift+←
Select to Line End	Alt+Shift+→	Cmd+Shift+→
Select Page Up/Down	Shift+PageUp/Down	Shift+PageUp/Down
Select to Start/End	Shift+Alt+↑/↓	Cmd+Shift+↑/↓
Delete Word Left	Ctrl+Backspace	Ctrl+Opt+Backspace
Delete Word Right		Option+Delete
Delete to Line End		Ctrl+K
Delete to Line Start		Option+Backspace
Indent	Tab (at start of line)	Tab (at start of line)
Outdent	Shift+Tab	Shift+Tab
Yank line up to cursor	Ctrl+U	Ctrl+U
Yank line after cursor	Ctrl+K	Ctrl+K
Insert yanked text	Ctrl+Y	Ctrl+Y
Show help for function	F1	F1
Save document	Ctrl+S	Cmd+S
Move Lines Up/Down	Alt+↑/↓	Option+↑/↓
Copy Lines Up/Down	Shift+Alt+↑/↓	Cmd+Option+↑/↓
Add New Cursor Above	Ctrl+Alt+Up	Ctrl+Alt+Up
Add New Cursor Below	Ctrl+Alt+Down	Ctrl+Alt+Down
Move Active Cursor Up	Ctrl+Alt+Shift+Up	Ctrl+Alt+Shift+Up
Move Active Cursor Down	Ctrl+Alt+Shift+Down	Ctrl+Alt+Shift+Down
Find and Replace	Ctrl+F	Cmd+F
Use Selection for Find	Ctrl+F3	Cmd+E
Replace and Find	Ctrl+Shift+J	Cmd+Shift+J

RStudio IDE cheat sheet

https://www.rstudio.com/wp-content/uploads/2016/01/rstudio-IDE-cheatsheet.pdf

Why do you have to use RStudio

1. A graphical workspace
2. Full-featured text editor
3. Tab-completion of filenames, function names and arguments
4. Variable inspection
5. Features, features, features

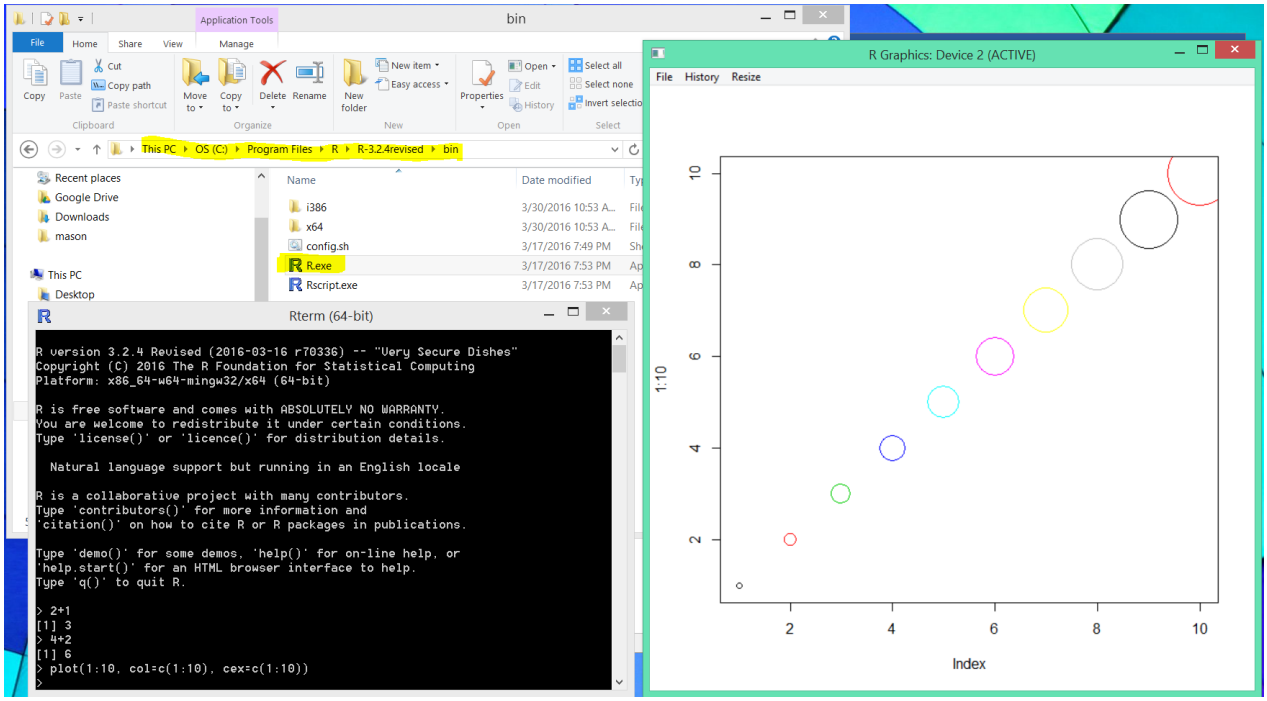


Lab #1: Take a look at the RStudio IDE cheat sheet and try some of the command not covered

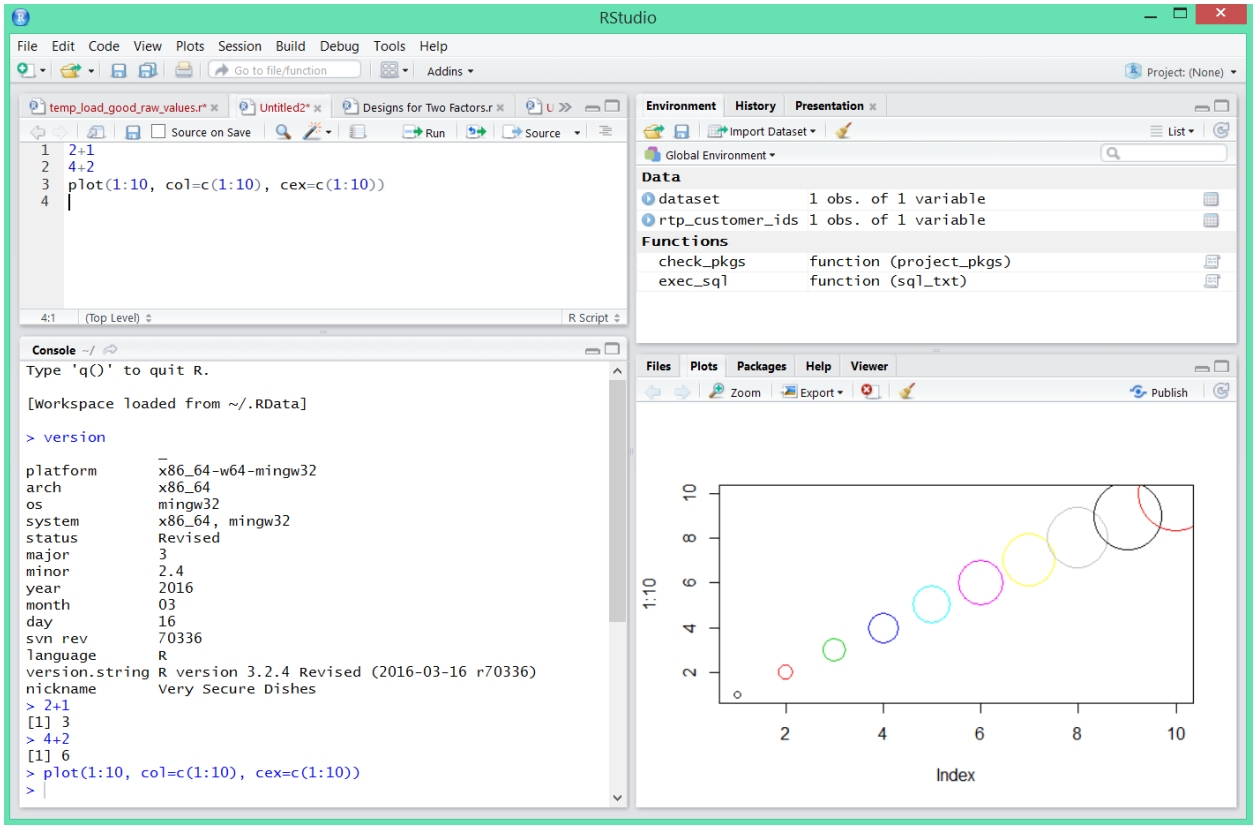
https://www.rstudio.com/wp-content/uploads/2016/01/rstudio-IDE-cheatsheet.pdf

R Software

A: Without R Studio



B: With RStudio (

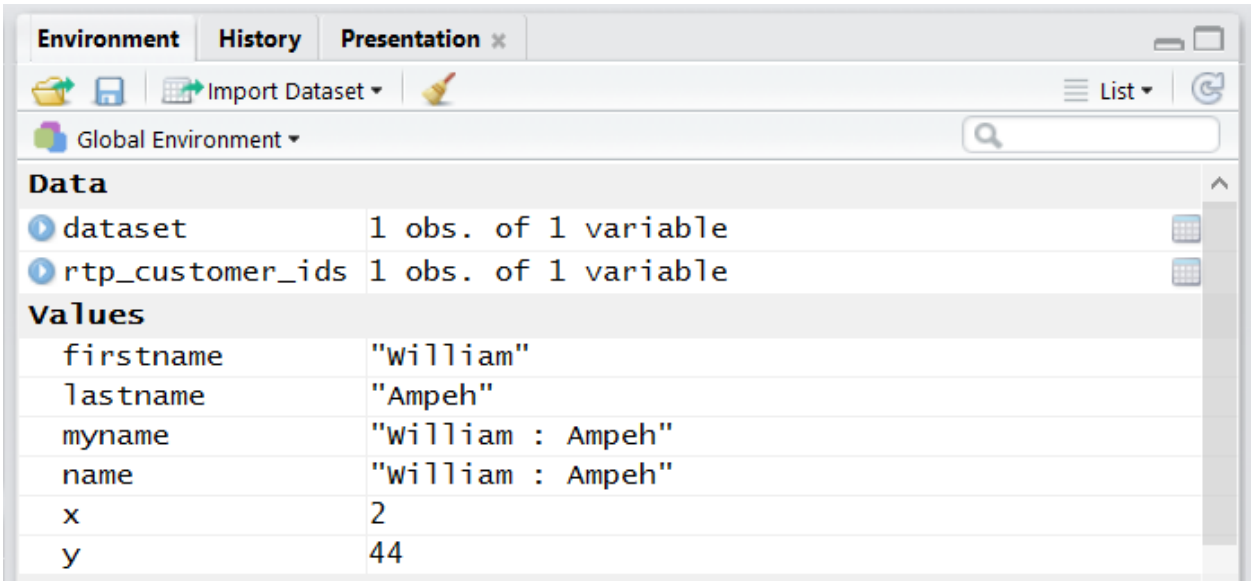


1: How to use R as a calculator

Start RStudio

- Start RStudio (Windows key, RStudio <Enter>)
- Key in as many mathematical expressions as can and note the results. You may use the following example:
    - 1 + 1
    - 2 + 5
    - sqrt(25)
    - x = 2
    - x + 3
    - factorial(6)
    - 6*5*4*3*2*1
    - 45 + <Press the Enter key>
      10
    - 5^3
    - y = 34 + 10
    - firstname = 'William'
    - lastname = 'Ampeh'
    - myname = paste(firstname, ' : ' , lastname)
    - myname

- Observe the variables x and y from the **Environment** window
- Also take note of the data type of ***x***, ***y*** and ***myname***



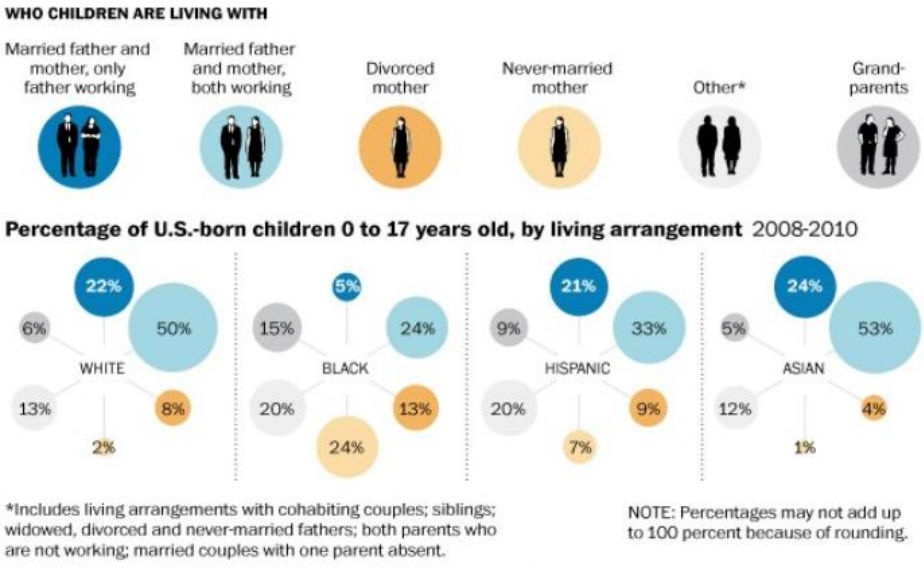
Note: Your window entries may differ from the one show above

2: Use Washington post article to describe the following concepts of observations and variables

Who U.S.-born children are living with

+ More

A study by Ohio State University's Department of Sociology found that more white and Asian children live among dual-income households than black and hispanic children, who have a higher percentage living with grandparents. Almost one-quarter of black children live with their single, never-married mother. Read related article.



Source: "Divergent Paths of American Families" report, Department of Sociology at Ohio State University | Cristina Rivero/The Washington Post September 11, 2013

Class discussion (5 minutes):

List what you thing will be the observation and the variables in the above picture

Observation:

Variables:

Values:

The two factors in the above plot are:

- race with 4 levels and
- family type with 6 levels.

An example of a family type is "Married, both working".

There are 24 values (***percents***) that correspond to the product set of:

- 4 races and
- 6 family types.

The family type (***percents***) are percent of the total for each race.

Note: The percent totals for each race turn out to be 99 or 101 due to rounding to integers.

3: R data types: vectors, arrays, lists, matrices, data frame and factors

A: Vectors: A vector is the most basic data structure in R. All values are vectors.

- Numeric: Vector of length 1

  x = 22

  y = 50

  z = 3.14

  typeof(x)

  typeof(y)

```
lapply(c(value, myPi, pi), typeof )   #see the class of each of the variables
```

- Character: Non-numeric values

  string = "FRB 1801K-street"

  Logic: True or False

  t = 2 <4

  typeof(t)

  class(t)

- Factors Factors (categorical data) and dates are built on top of integers

```
x <- factor( c("Yes", "No", "No", "Yes", "Yes", "Maybe") )
x
```

classof(x)

```
s <-  factor( rep(c("Male", "Female"), times=c(3, 4)) )
s
```

B: List: Generic vector containing other objects of the same type

```
xList <- list(1, 2, 3)
```

```
v = c("a","b",1,2,3,TRUE)
```

Discussion: 1: Display the value of v

2: What is the ***type*** and ***class*** of v?

C: Matrix: A collection of data elements arranged in a 2-D rectangular layout

The data elements of a matrix must be of the same basic type.

```
A = matrix(
   c(1:12),             # the data elements
   nrow=3,              # number of rows
   ncol=4,              # number of columns
   byrow = TRUE         # fill matrix by rows
   )
```

```
A                       # print A
```

Another Matrix

column_1 <- c(22, 50, 8, 2, 13, 6)
column_2 <- c(5, 24, 13, 24, 20, 15)
column_3 <- c(21, 33, 9, 7, 20, 9)
column_4 <- c(24, 53, 4, 1, 12, 5)

mat <- cbind(column_1, column_2, column_3, column_4)
mat

An element at the mth row, nth column of A can be accessed by the expression A[m, n].

```
mat[3, 1]          # element at 3nd row, 1rd column

mat[1,  ]          # all elements of the first row

mat[ , 2]          # second column elements
```

We can also extract more than one rows or columns at a time.

```
mat[ ,c(1,4)]    # the 1st and 4rd columns
```

Discussion: Create a new matrix using:

1: the first 2 columns of mat
2. last 2 rows of mat
3: the 3rd and 4th column of mat
4: extract the number **53** from the **mat** matrix
5: find the sum of the data values in row 2 of mat
6: find the mean and standard deviation of the data values 2nd column of mat

D: A data frame a list of vectors of equal length.

For example, the following variable df is a data frame containing three vectors a, b, c.

```
a = 1:3                                #same as c(1, 2, 3) or seq(from=1, to=3, b
y=1)
b = c("aa", "bb", "cc")                # 3 character sets
c = c(TRUE, FALSE, TRUE)               # 3 logic values


mydf = data.frame(a, b, c)             # mydf is a data frame


head(mydf, n=2)
```

```
m.df <- data.frame(column_1 = c(22, 50,  8,  2, 13,  6),
                   column_2 = c( 5, 24, 13, 24, 20, 15),
                   column_3 = c(21, 33,  9,  7, 20,  9),
                   column_4 = c(24, 53,  4,  1, 12,  5))

m.df
```

In RStudio

```
> m.df <- data.frame(column_1 = c(22, 50,  8,  2, 13,  6),
+                    column_2 = c( 5, 24, 13, 24, 20, 15),
+                    column_3 = c(21, 33,  9,  7, 20,  9),
+                    column_4 = c(24, 53,  4,  1, 12,  5))
> m.df
  column_1 column_2 column_3 column_4
1       22        5       21       24
2       50       24       33       53
3        8       13        9        4
4        2       24        7        1
5       13       20       20       12
6        6       15        9        5
```

Discussion: Create a data frame using the data values from the **mat** matrix

1: the first 2 columns of mat
2. last 2 rows of mat
3: the 3rd and 4th column of mat
4: extract the number **53** from the **mat** matrix
5: find the sum of the data values in row 2 of mat
6: find the mean and standard deviation of the data values 2nd column of mat

Build-in Data Frame

R has a number of built-in data frames, for example, here is a built-in data frame in R, called mtcars.

```
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

Data frame slicing using Numeric Indexing

To display the cell value from the first row, second column of mtcars.

```
mtcars[1, 2]
```

Data frame slicing using Name Indexing

Moreover, we can use the row and column names instead of the numeric coordinates.

```
mtcars["Mazda RX4", "cyl"]
```

```
## [1] 6
```

Data frame slicing using Logical Indexing

finally, we can also retrieve rows with a logical index vector

```
G = mtcars$gear == 3              #[,10]     gear    Number of forward gears

head( mtcars[G, ] )
```

```
##                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
```

R Variable names

A syntactically valid name consists of letters, numbers and the dot or underline characters and starts with a letter or the dot not followed by a number. Names such as ".2way" are not valid, and neither are the reserved words.

R's make.names() function can be used to check and/or create valid variable names

```
# ?make.names
make.names(c(".2pi"))
```

```
## [1] "X.2pi"
```

Discussion: Try forming as many variable names as you can using R's ***make.name***s function

The Mosaic package

The mosaic package is designed to help simplify the interface for R users, while allowing them to undertake sophisticated statistical analyses.

```
#install.packages("mosaic")
library(mosaic)
```

```
( x <- 1:10 )
( xmean <- mean(x) )                #new (x <- 1)    # assign and display
( xsd <- sd(x) )

# display functions in this package
ls("package:mosaic")

#
#help(mosaic)                     #Homework Q5.b

# Use the summary function
summary()
```

End of session