```
###################################
####  entering data            ####
###################################
#variables
x=3
y=4
3*x
x+y
y^2

#using c() function
my.data=c(11,14,22,15)    #this code will create your dataset and assign it the variable name
my.data
my.data    #If you want to know what a variable holds, you must ask for it by name
my.data+500
2*my.data
length(my.data)

#using :
stats.data=1:5   #this code will create your sequence of data from 1 to 5, by increments of 1
stats.data

decreasing.data=5:-6   #this code will create your sequence of data from 5 to -6, by increments
of -1
decreasing.data

#using seq() function
seq(0,3,by=2)   # seq(low.number, high.number, increment.size)
seq(0,3,length=5)  # seq(low.number, high.number, sequence.length)

#using rep() function
rep(c(1,2,3), each=3)
rep(c(1,2,3), times=3)

###################################
####  math functions           ####
###################################
2 + 4 * 5      # Order of operations
log (10)       # Natural logarithm with base e=2.7182
log10(5)      # Common logarithm with base 10
5^2             # 5 raised to the second power
5/8             # Division
sqrt (16)      # Square root
abs (3-7)     # Absolute value
pi                # 3.14
exp(2)         # Exponential function
round(pi,0)  # Round pi to a whole number
round(pi,1)  # Round pi to 1 decimal place
round(pi,4)  # Round pi to 4 decimal places
floor(15.9)   # Rounds down
ceiling(15.1)  # Rounds up
cos(.5)        # Cosine Function
sin(.5)        # Sine Function
tan(.5)       # Tangent Function
acos(0.8775826)        # Inverse Cosine
asin(0.4794255)       # Inverse Sine
atan(0.5463025)      # Inverse Tangent



###################################
```

```
####  data manipulation        ####
#####################################
x=c(1,3,2,10,5)    #create a vector x with 5 components
### Sort your data
sort(x)                # increasing order
sort(x, decreasing=T)  # decreasing order

### Component extraction is a very important part of vector calculation.
x
length(x)           # number of elements in x
x[4]                # the fourth element of x
x[2:5]              # the second to fifth element of x, inclusive
x[-2]               # all except the second element
x[-c(1,2)]          # all except the first and the second elements
x[x>3]              # list of elements in x greater than 3

### Logical vector can be handy:
x>3                   # Which x values are greater than 3
sum(x>3)            # number of elements in x greater than 3

#####################################
####  numerical summaries      ####
#####################################
x=c(34, 24, 10, 16, 52, 76, 33, 31, 46, 24, 18, 26, 57, 32, 25, 48, 22, 48, 29, 19)
x
max(x)
min(x)
which.min(x)
which.max(x)
mean(x)
mean(x, trim = 0.1)
median(x)
sd(x)
var(x)
range(x)
sum(x)
summary(x) # Calculate five-number summary: min,Q1,median,Q3,max, return an object
my.q=fivenum(x) # calculate five-number summary, return a numeric vector
quantile(x, c(.25, .5, .75, 1)) ### Calculates the numbers associated to defined percentiles
quantile(x,0.99)
iqr=IQR(x)

q1=my.q[2]
q3=my.q[4]
q1-1.5*iqr    #lower fence for outliers
q3+1.5*iqr    #upper fence for outliers
q1-3*iqr      #lower fence for extreme outliers
q3+3*iqr      #upper fence for extreme outliers

#####################################
####  Built-in data frames    ####
#####################################
### We need to load a library to work with the battery dataset
install.packages('EngrExpt')
library(EngrExpt);
data(battery);
attach(battery)

### Let's view the actual dataset
battery

### The following 3 lines of code will create three separate datasets
```

```
### by battery type
type.a=subset(battery, type=="A")

type.b=subset(battery, type=="B")

type.c=subset(battery, type=="C")


### Let's take a look at dataset type.a
type.a

### Let's look at just the lifetime variable within the type.a dataset
type.a$lifetime      ### DataSetName$VariableName

### Now, we can analyze the lifetime variable within the type.a dataset
mean(type.a$lifetime)
sd(type.a$lifetime)

## more built-in dataframes
## https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html

###################################
####  Graphics: categorical    ####
###################################
#### EXAMPLE OF THE table FUNCTION
x=c("Yes", "No", "Yes", "Yes", "No")  # Create categorical data
x

table(x)  # table function returns the frequency for each category of data

### Now we will use storm/hurricane data
# You have to download your data in some folder and set up this folder as working directory
setwd('D:\\Dropbox\\classes\\324R') #set working directory
getwd() #get working directory
sh = read.table("u.s._hurricane_data2011.txt",header=T)
cat = sh$cat
year = sh$year
head(sh)
# Create a table for the category of storm/hurricane
cat.counts=table(cat)

# Create a table for the year of the storm/hurricane
year.counts=table(year)

# View the respective tables
cat.counts

year.counts

###### CREATING BAR GRAPHS

# Create a barplot of storm/hurricane categories
barplot(cat.counts, col="red")
# Create a barplot using percentages for the years
barplot(year.counts/length(year), col="green")

###### CREATING PIE CHARTS

# Basic pie chart
pie(cat.counts)

# Add names to your piechart
```

```
names(cat.counts)=c("2005", "2006", "2007", "2008", "2009", "2010")


# Pie chart with names printed out
pie(cat.counts)

# Now, add a little color and a main :)
pie(cat.counts, col=rainbow(12), main="Atlantic Hurricanes by Year")

###################################
####  Graphics: quantitative   ####
###################################
## stem-and-leaf plot
### Lets work with the pre-installed cars data set found in R
cars; attach(cars)

### default stem, doesn't represent data so well
stem(dist)

### The parameter scale can be used to expand the scale of the plot.
### A value of scale=2 will cause the plot to be roughly twice ### as long as the default.
stem(dist, scale=2)

### experiment with a scale of 4
stem(dist, scale=4)

### dot plot
## The following dataset is the time required for a
## group of students to complete a statistics quiz
quiz.time=c(15, 12, 13, 19, 18, 17, 16, 18, 22, 21, 11, 14, 13, 17, 17)

stripchart(quiz.time,
           method="stack",
           pch=19,
           main="Student Quiz Completion TIme",
           col="Red",
           offset=0.5,
           xlab="Time (min)")
# or
source("dotplot.R")
dotplot(quiz.time)
###histogram
glucose = c(81, 85, 93, 93, 99, 76, 75, 84, 78, 84, 81, 82, 89,
            + 81, 96, 82, 74, 70, 84, 86, 80, 70, 131, 75, 88, 102, 115,
            + 89, 82, 79, 106)

hist(glucose)
hist(glucose, breaks = 14)    # here breaks is a single number giving the number of cells for
the histogram
hist(glucose, breaks = seq(60, 140, by = 8), main="Equal Intervals") # here breaks is a vector
giving the breakpoints between histogram cells
hist(glucose, breaks = c(60,75,80,100,140), main="Unequal Intervals")



###single box-plot#######################
green = c(34, 24, 10, 16, 52, 76, 33, 31, 46, 24, 18, 26, 57,
          + 32, 25, 48, 22, 48, 29, 19, 100, -20)

### Boxplot A
boxplot(green, col = "blue") ### Default, vertical boxplot
```

```
### Boxplot B
### Create a horizontal boxplot
boxplot(green, horizontal=TRUE, col="blue")

###box-plot comparison################################
male = c(6, 0, 2, 1, 2, 4.5, 8, 3, 17, 4.5, 4, 5)
female = c(5, 13, 3, 2, 6, 14, 3, 1, 1.5, 1.5, 3, 8, 4)
boxplot(list(male = male, female = female), col="red")


growthDark = c(15, 20, 11, 30, 33, 22, 37, 20, 29, 35, 8, 10, 15, 25)
growthLight = c(10, 15, 22, 25, 9, 15, 4, 11, 20, 21, 27, 20, 10, 20)
boxplot(list(dark = growthDark, light = growthLight),col=c("red","blue"))

# Multivariate data
library(MASS)
data(UScereal)
head(UScereal)
summary(UScereal)

UScereal$shelf=as.factor(UScereal$shelf)
summary(UScereal)

attach(UScereal)
hist(sodium)
abline(v=mean(sodium),lty=3)
plot(potassium, protein)

boxplot(sugars)
```