LAB1

I. More on vectors
1. contain elements of just 1 type, either numeric, logical, or character
   a. numeric
      
      \[
      y = 18 \\
      y = c(-3.6, 1.58) \\
      y = 1:9 \\
      y = seq(from=0, to=10, by=0.02) \quad \# \text{seq() a function w/ args} \\
      \quad \quad \# \text{from, to, \ldots by argument} \\
      \quad \quad \# \text{gives distance} \\
      y = seq(0, 10, by=0.02) \quad \# \text{if know where args are,} \\
      \quad \quad \# \text{no need to specify name} \\
      \]
      \[
      y = seq(-3, 25, length=6) \quad \# \text{length option} \\
      x = c(3, 3, 9, 12, 1) \\
      y = seq(0, 10, along=x) \quad \# \text{will return seq w/} \\
      \quad \quad \# \text{length(x)} \\
      y = rep(x=3, times=7) \quad \# \text{replicate the value} \\
      \quad \quad \# \text{in } x, \text{ namely 3, seven times} \\
      y = rep(x=c(3, 5), times=7) \quad \# \text{same} \\
      y = rep(x=c(3, 5), times=c(2, 6)) \quad \# \text{repeat 3 two times,} \\
      \quad \quad \# \text{then 5 seven times} \\
   
   b. character
      
      \[
      y = "San Diego" \\
      y = c("LA", "NY", "San Fran") \\
      \]
   c. logical
      
      \[
      y = T \\
      y = c(F, F, T) \\
      x = 1:7 \\
      y = (x<5) \\
      \]

2. Accessing elements of a vector in 1 of 4 ways
   y = c(18, 32, 15, -7, 12, 19)
   a. position in vector as positive integer
      y[3:5]
   b. excluding elements, position as negative integers
      y[-c(1, 5, 6)]
   c. by element name
      names(y) = c("Joe", "Bill", "Karen", "Helen", "Ray", "Paul")
      y[c("Helen", "Ray")]
   d. by logical conditions
      y[y<15]

3. Matrices, arrays, and factors
   a. Matrices
      y = c(18, 32, 15, -7, 12, 19)
x = matrix(data=y,nrow=2,ncol=3)       #fill columns first
x = matrix(data=y,nrow=2,ncol=3,byrow=T)  #fill rows first
dimnames(x) = list(c("r1","r2"),c("a","b","c"))

b. Arrays
   # creates a 2 by 4 by 3 array
y = c(1:8, 11:18, 111:118)
x = array(y, dim = c(2,4,3))    #first dimension changes
   #fastest

c. Factors
   color = c("red", "red", "red", "green", "blue")
   #character valued vector
   colors = factor(color, c("red","green","blue"))
table(colors)  #table counting occurrences of colors
   #one use: when building models where
   #"dummy" variables needed

4. Vector attributes
   every vector (and object) has 2 "implicit" attributes, mode
   and length. Some modes, e.g. "numeric", "list", "logical",
   "NULL"

   other attributes- name, dimension, levels, class
   attributes(y)   #for "plain" vector doesn't report anything
   attributes(x)
   attributes(colors)

II. More on lists
x = list(one=c(18:36),two=c("AK","AL","AZ"),three=c(T,T,F,T),
        four=matrix(1:12,3,4))

1. access to components
   x[[1]]          #by order
   x$one           #by name

2. access to elements within components
   x[[1]][3:6]
   x$one[3:6]

3. unlist()  #convert a list to a vector
   unlist(x)  #handy for printing out returned values from
   #function

4. attach(#makes components of list directly
   #accessible
   attach(what=x)
   one
detach(what="x")   #no longer make this direct access
   #possible

5. data frames- a special kind of list object; number of
   elements must be the same for all components
   - handy for regression modeling

   muscle = rnorm(n=10,mean=3,sd=1)
   sex = factor(rep(c("M","F"),c(6,4)))
   speed = rep(0,10)
   speed[1:6] = rnorm(6,30-2*muscle[1:6],2)
speed[7:10] = rnorm(4,40-2*muscle[7:10],2)
mydata = data.frame(y=speed,x1=muscle,x2=sex)
temp = lm(y~x1+x2,data=mydata)
summary(temp)

III. More on functions
1. specified or unspecified arguments
   a. unspecified   c = function(...) 
   b. specified
      sd = function(x) {return(sqrt(var(x)))}

      dumplot = function(x,y) {
        plot(x,y)
        abline(0,1)
      }

2. argument passing for specified case 
   if don't use names, call function by passing arguments in proper order. E.g. dumplot(my.x,my.y) 
   if use names, then any order ok 
   dumplot(y=my.y, x = my.x)

3. default values for specified case 
   dumplot = function(x,y,my.line=T) {
     plot(x,y)
     if(my.line) {
       abline(lm(y~x))
     }
   }

LAB 2

# Arithmetic and logical expressions

#0. Precedence
#See Table 2.1 on p. 33 of V&R, but some common situations, highest to lowest:

  function(..)
  ^            #exponentiation
  * /          #multiplication, division
  + -          #addition, subtraction
  > < <= == != #comparison operators

#but can always use parentheses to force a precedence

  x = 36
  y = 7
  sqrt(x)*7-2 < x+y^2
  (((sqrt(x)))*7)-2) < (x+(y^2))  #identical to above but "clear" precedence

#1. element by element operations on two vectors
#2. recycling: when operating on 2 vectors with unequal length
#   length of output = length of longer vector, recycling the
#   shorter vector's values till match longer vector's length

x = 2
y = 1:6
x+y   #2+1, 2+2, 2+3, 2+4, 2+5, 2+6

x = c(2,9)
x+y   #2+1, 9+2, 2+3, 9+4, 2+5, 9+6

#3. Integer division and modulo reduction
7 %/% 3       #answer 2, number of times 3 goes into 7
7 %% 3       #answer 1, i.e., 7 (mod 3), or 7 - (7 %/% 3)*3

#4 Logical expressions and functions with logical arguments
x = c(1,3,5)
y = c(2,2,4)

x < y
x <= y
x != y

any(x<y)  #if any term in vector x<y is T, return T
all(x<y)  #if all terms in vector x<y is T, return T

(x==3 | y==2)  # union of vector x==3 with vector y==2
              # T|T or T|F or F|T returns T
              # only F|F returns F

(x==3 & y==2)  # intersection of vector x==3 with vector y==2
               # only T&T returns T

LAB 3

# Simple arithmetic functions

#1. element by element
x = 3:12
abs(x)
log(x)
log10(x)
sqrt(x)
exp(x)
sin(x)

#2. operate on entire vector:
# returns a scalar value
sum(x)  #multiply each term
prod(x)  #return vector of same length
cumprod(x)  #x[1], x[1]*x[2], x[1]*x[2]*x[3]

#3. conversion to integers, rounding,..
x = 103.652
round(x)
round(x,1)
round(x,2)
trunc(x)  #nearest integer in direction of 0
trunc(-0.8)
floor(x)  #nearest integer in direction of negative infinity
floor(-0.8)
ceiling(x)  #nearest integer in direction of positive infinity
ceiling(-0.8)

#4. max, min related
x = c(3,5,5,9)
y = c(5,3,5,11)
max(x)
min(x)
rangle(x)  #useful for setting limits on graph axes
pmax(x,y)  #per position [i] return maximum of x[i] and y[i]
pmin(x,y)

#5. ordering
x = c(12,5,6,6,-1)
sort(x)
rev(x)
order(x)  #returns position of smallest, next smallest, ...
rev(order(x))  #returns position of largest, next largest,...
rank(x)  #rank of x[1], then rank of x[2]; useful for
ranksum statistics

#6. simple statistics
x = c(8,12,5,4,19,2,1)
y = rnorm(length(x),mean=3+2*x,sd=1)
mean(x)
var(x)
median(x)
quantile(x,probs=c(0.25,0.75))
summary(x)  #a generic function used on lots of classes of data

LAB 4

#Random variables- generating, distribution functions

#0. random number seed; fix for reproducible simulations
i = 5
set.seed(i)

#1. random sampling
z = c(18, 22, 15, 25, 36)
sample(x=z,size=3,replace=F)# take a SRS w/o replacement from z
sample(x=z,size=3,replace=T)# now w/ replacement; as for #bootstrap
sample(x=z,size=3,replace=T,prob=c(0.1, 0.3, 0.2, 0.1, 0.3))
# unequal prob sampling

#2. univariate random variable generation (see p. 164, Table 5.1)
# beta, binomial, Cauchy, normal, lognormal, gamma, F, poisson, Chi-square
rbeta(n=10,shape1=3,shape2=4)    # ten obs'ns from Beta(3,4)
rbinom(n=10,size=100,prob=0.2)   # ten obs'ns from Binomial(100,0.2)
rnorm(n=10,mean=20,sd=5.1)       # ten obs'ns from Normal(20,5.1^2)
rpois(n=30,lambda=3)             # ten obs'ns from Poisson(3)

#3. percentages- for P-values in particular; can discard tables in back of stat texts...
pnorm(q=1.645,mean=0,sd=1)
1-pf(q=5.7,df1=3,df2=15)

#4. quantiles- for test statistics, for example
qt(p=0.975,df=21)

#5. density function value; or probability for discrete valued rv
dnorm(x=0.5,mean=0,sd=1)
dpois(x=1,lambda=2)

Acknowledgement: The above are adapted from professor Ken Newman’s Splus notes.