9 Jan 2022, Monday

R Programming Lab
Assignment 1
Vectors and List

1. Type the following code, which assigns numbers to objects x and y.
   x <- 10
   y <- 20
   a. Calculate the product of x and y.
      Store the result in a new object called z.
   b. Inspect your workspace by clicking the Environment tab in Rstudio, and find the three objects you created.
   c. Make a vector *myvec* of the objects x, y, and z.
   d. Find the minimum, maximum, and length of *myvec*.

2. The numbers below are the first ten days of rainfall amounts in 1996. Read them into a vector using the c() function.
   0.1, 0.6, 33.8, 1.9, 9.6, 4.3, 33.7, 0.3, 0.0, 0.1
   a. What was the mean rainfall? How about the standard deviation?
   b. Calculate the cumulative rainfall (’running total’) over these ten days. Confirm that the last value of the vector that this produces is equal to the total sum of the rainfall.
   c. Which day saw the highest rainfall (which.max() function)?

3. Read in a vector that contains "A", "B", "C" and "D" (use the c() function). Using rep, produce this:
   a. "A" "A" "A" "B" "B" "B" "C" "C" "C" "D" "D" "D"
   b. "A" "B" "C" "D" "A" "B" "C" "D" "A" "B" "C" "D"

4. 26 letters of the Roman alphabet are conveniently accessible in R via letters and LETTERS. These are not functions but vectors that are always loaded.
   a. Draw 10 random letters from the lowercase alphabet, and sort them alphabetically (Hint: use sample and sort).
   b. Draw 5 random letters from each of the lowercase and uppercase alphabets, incorporating both into a single vector, and sort it alphabetically.
   c. Repeat the above exercise but sort the vector alphabetically in descending order.

5. Make two vectors:
   x <- c(1,2,5,9,11)
   y <- c(2,5,1,0,23)
   Experiment with the three functions(*union, setdiff, and intersect*) to find solutions to these questions.
   a. Find values that are contained in both x and y.
   b. Find values that are in x but not y (and vice versa).
   c. Construct a vector that contains all values contained in either x or y, and compare this vector to c(x,y).

6. Merge two given vectors into one list.
   n1 = 1,2,3
   c1 = "Red", "Green", "Black"

7. Sort a Vector in ascending and descending order.
   X = 10, 20, 30, 25, 9, 26

8. Find all elements of a given list that are not in another given list.
   l1 = "x", "y", "z"
   l2 = "X", "Y", "Z", "x", "y", "z"

9. Add a new item, "Python" to a given list.
   Sample list: (1:10, "R Programming", "HTML")

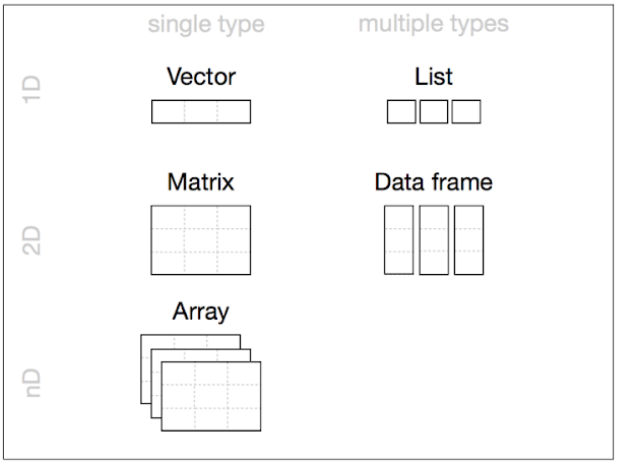
10. Create a list containing a sequence of 15 capital letters, starting with ‘E’. (Hint: use match() function)

14 Jan 2022, Saturday

R Programming Lab
Assignment 2 with Concepts
Vectors and List Cont.

Dimension	Homogenous	Heterogenous
1	Atomic vectors	Lists
2	Matrix	Data frame
≥ 1	Array	

- *Vector*: All elements *must* have the same data type.
- *List*: Different elements can have different types, including vectors, matrices, and lists.



1. Make two vectors of **equal** length:
   x <- c(500, 400, 600)
   y <- c(10, 5, 100)
   a. Call x + y (addition)
   b. Call x / y (division)

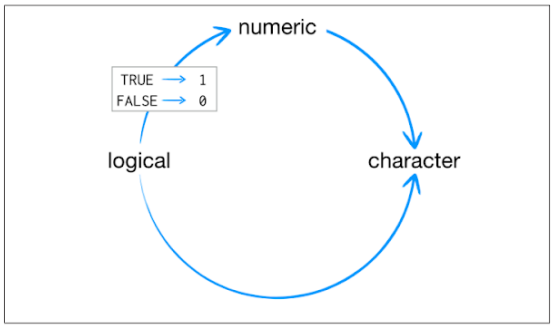
2. Make two vectors of **unequal** length:
   x <- c(500, 400, 600, 800)
   y <- c(100, 2)
   a. Call x + y (addition)
   b. Call x / y (division)

Observe the difference between ques 1 and 2.

3. Create a vector
   v <- c(10,20,30,40,50,60,70,80,90,100)

a. is.vector(v)
b. length(v)
c. typeof(v) #R will save any number that you type in R as a double.
d. v[1:3]
e. v[8:10]
f. head(v, n = 3)
g. tail(v, n = 3)

4. **Coercion:** If you try to put multiple types of data into a vector, R will convert the elements to a single type of data.
   - If a character string is present in an atomic vector, R will convert everything else in the vector to character strings.
   - If a vector only contains logicals and numbers, R will convert the logicals to numbers; every TRUE becomes a 1, and every FALSE becomes a 0.



   a. Create vectors and check their type.
   b. Convert data from one type to another with the **as** functions:
      i. as.character(1)
      ii. as.logical(1)
      iii. as.numeric(FALSE)

5. Create a **named vector**
   x <- c(age = 35, height = 1.72, zipcode = 6020)
   a. x["age"]
   b. x[age]
   c. x[c("age", "zipcode")]
   d. x[1] # Single brackets
   e. x[[1]] # Double brackets

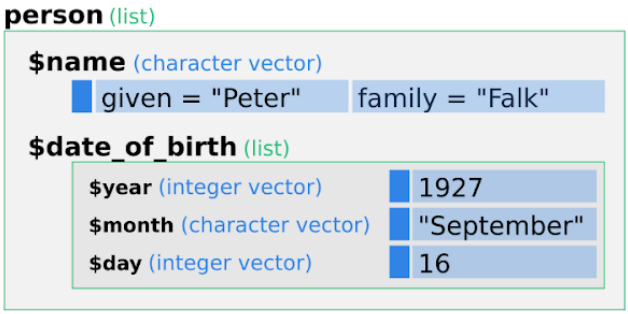
6. Merge two given vectors into one list L1.
   n1 = 1,2,3
   c1 = "Red", "Green", "Black"
   a. L1[1]

b. L1[2]
c. L1[[1]]
d. L1[[1]][3]

7. Create a named list L2.
   L2 <- list(number = c(1,2,3), color = c("Red", "Green", "Black"))
   a. L2
   b. L2[1]
   c. Try L2[[1]] and L2$number
   d. Try L2[[1]][3] and L2$number[3]

8. **Recursive named List**
   person <- list(name = c(given = "Peter", family = "Falk"),
   date_of_birth = list(year = 1927, month = "September", day = 16))
   a. typeof(person)
   b. length(person)
   c. class(person)
   d. names(person)
   e. person$name
   f. person$date_of_birth
   g. person$date_of_birth$month
      How to read:
      - Right to left: Return month from date_of_birth of the object person.
      - Left to right: Inside object person access the element date_of_birth, inside date_of_birth access element month.

   h. str(person) function returns us a text representation similar to the image shown below.



   i. person[[c(1, 2)]]
   j. person[[c("name", "last_name")]]

9. The following object demo is a list with information about two persons, Frank and Petra.

```
demo <- list(
          "Petra" = list(location = "Birmingham", kids = NULL, job = "Programmer"),
          "Frank" = list(location = "Kufstein",   kids = c("Peter", "Paul"))
         )
```

a. How do we get Franks location?
b. Try demo["Frank"]$location (will return NULL). Why doesn't this work?
c. How many kids does Frank have?
d. Our friend Petra moves from Birmingham to Vienna. Change her location (inside demo) to Vienna.

14 Jan 2022, Saturday

R Programming Lab
Assignment 3 with Concepts
Matrices

Dimension	Homogenous	Heterogenous
1	Atomic vectors	Lists
2	Matrix	Data frame
≥ 1	Array	

- Matrices can only contain data of **one type** (like vectors).
- A matrix is a special array with **two dimensions**.

1. Try usage of **?matrix** or **help("matrix")**

2. Create a vector x=(1,2,3,4,5,6,7,8,9)
    a. Create a 3X3 matrix m using vector x.
    b. Fill elements of x by row.
    c. Check dimension of matrix m (hint: dim())
    d. Find the number of rows and number of columns (hint: nrow(), ncol())
    e. Find the number of elements in m

3. Create four matrices based on vectors of different types (double, integer, character, and logical). Investigate the objects.
    a. x1 <- matrix(seq(0, 4.5, length.out = 9), nrow = 3) # double
    b. x2 <- matrix(1:9, nrow = 3) # integer
    c. x3 <- matrix(LETTERS[1:9], nrow = 3) # character
    d. x4 <- matrix(TRUE, nrow = 3, ncol = 3) # logical

4. Try to create the following matrices. To do so, we need to specify data, the number of rows, and the number of columns.
    a. A matrix of dimension 5X5 which contains 5L (integer) everywhere.
    b. A matrix of dimension 10X1 which contains -100 (numeric) everywhere.
    c. Check that the class of your result is c("matrix", "array")

5. Generate 50 random numbers using rnorm() with mean=50, sd=10.
    a. Create a large matrix called mat (dimension 10X5) with generated random numbers.

b. Call head(mat) and head(mat, n = 2) to get the first 6 (default) or 2 rows only.
c. Call tail(mat) and tail(mat, n = 3) to get the last 6 or 3 rows.
d. Call summary(mat) and try to interpret the output. Do you see what happens? (Hint: numerical summary for each column individually)
e. Try to answer the following questions:
    i. Get the largest and smallest value in the matrix (minimum and maximum).
    ii. What is the arithmetic mean (average), what the standard deviation of the entire matrix?
    iii. What is the sum of the entire matrix mat?

6. Arithmetic using Matrices and scalars:
   x <- matrix(1:4, ncol = 2)

    a. x + 2
    b. x * 2
    c. x / 2
    d. x ^ 2

7. Arithmetic using Matrices and Vectors:
   x <- matrix(1:4, ncol = 2)
   y <- c(10, 100)

    a. Compute x*y

8. Arithmetic using Matrices and Matrices:
   x <- matrix(c(1, 2, 3, 4), ncol = 2, nrow = 2)
   y <- matrix(c(10, 20, 30, 40), ncol = 2, nrow = 2)

    a. x + y
    b. x * y
    c. x / y

9. Create a matrix x
   x <- matrix(c(1, 2, 3, 4), ncol = 2, nrow = 2)

    a. Transpose of x using trans(x).
    b. Diagonal elements of x using diag(x).
    c. Matrix multiplication using x %* x. Check how is this different from x*x.

10. Set dimension names:
    x <- matrix(data = 1:9, nrow = 3, ncol = 3)

    a. Check rownames(x); colnames(x); dimnames(x)
    b. Run
       rownames(x) <- c("Row 1", "Row 2", "Row 3")
       colnames(x) <- c("Col A", "Col B", "Col C")
       Print the matrix m.
       Check rownames(x); colnames(x); dimnames(x)

11. Create named matrix nm
    nm <- matrix(data = 1:9, nrow = 3, ncol = 3,
                 dimnames = list(c("Row 1", "Row 2", "Row 3"),
                                  c("Col A",   "Col B", "Col C")   )
)

12. Create matrix by combining vectors.
       # Generate vectors
       x <- c(5, 5, 5)
       y <- c(11, 22, 33)

       # Combine
       z <- cbind(x, y)
       z <- rbind(x, y)

13. Let's see if we try to column-bind vectors of different lengths. For simplicity, only use integer vectors of the form 1:2 (vector of length 2) or 1:5 (vector of length 5).
    Try to row-bind and/or column-bind vectors of length:

    a. 4 and 4
    b. 8 and 4
    c. 4 and 8
    d. 4 and 1
    e. 5 and 3
    f. 9 and 10

14. Combine matrices:
     x1 <- matrix(1:6,   ncol = 2)
     x2 <- matrix(101:106, ncol = 2)

    a. cbind(x1, x2)
    b. rbind(x1, x2)

15. Hands on matrix subsetting. Try to answer the questions based on the following numeric matrix mat.
mat <- matrix(c(270, 100, 330, 340, 260, 160, 10, 310, 80, 50, 60, 190, 150, 110, 290, 220, 10, 350, 100, 0), nrow = 5)

    a. What is the value of element mat[3, 2]?
    b. What is the value of element mat[2, 4]?
    c. What is the value of element mat[7], and how can we extract the same element using row and column indices?
    d. What is the value of element mat[15], and how can we extract the same element using row and column indices?
    e. mat[1,]
    f. mat[, 3]
    g. mat[mat>100] : get all elements in the matrix which are larger than 100