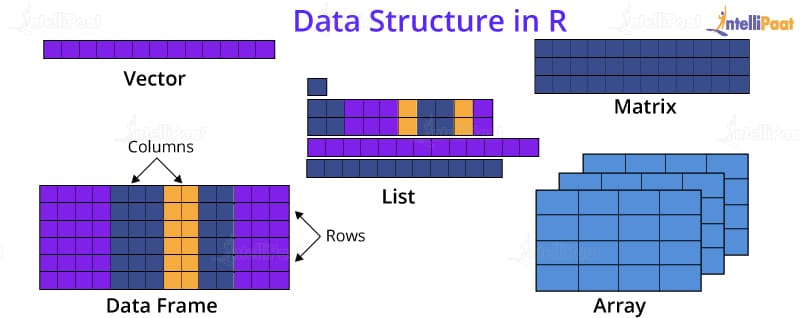
### **How many data structures R has?**

There are 5 [data structures in R](https://intellipaat.com/blog/tutorial/r-programming/data-structures-r-programming/) i.e. vector, matrix, array which are of a homogenous type and the other two are list and data frame which are heterogeneous.



Module 2.

Data Visualization using R: Reading and getting data into R (External Data): Using CSV files, XML files, Web Data, JSON files, Databases, Excel files.

Working with R Charts and Graphs: Histograms, Boxplots, Bar Charts, Line Graphs, Scatterplots, Pie Chart

Plot

Plot

The plot() function is used to draw points (markers) in a diagram.

### **Example**

Draw one point in the diagram, at position (1) and position (3):

plot(1, 3)

Draw two points in the diagram, one at position (1, 3) and one in position (8, 10):

Eg plot(c(1, 8), c(3, 10))

Eg plot(c(1, 2, 3, 4, 5), c(3, 7, 8, 9, 12))

Eg x <- c(1, 2, 3, 4, 5)  
y <- c(3, 7, 8, 9, 12)  
plot(x, y)

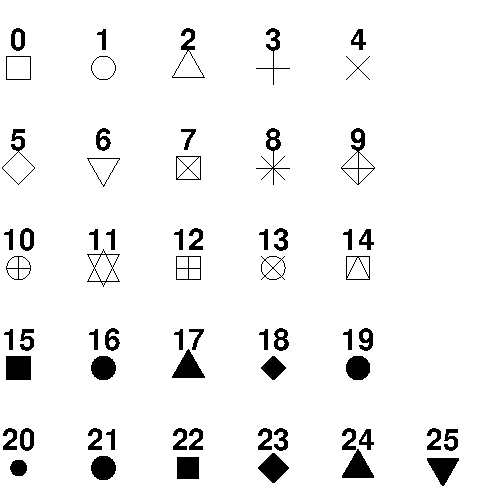
plot(1:10)

The plot() function also takes a type parameter with the value l to draw a line to connect all the points in the diagram:

plot(1:10, main="My Graph", xlab="The x-axis", ylab="The y axis")

plot(1:10, col="red")

plot(1:10, pch=25, cex=2)



**Line plotting**

plot(1:10, type="l")

plot(1:10, type="l", col="blue")

plot(1:10, type="l", lwd=2)

To change the width of the line, use the lwd parameter (1 is default, while 0.5 means 50% smaller, and 2 means 100% larger)

plot(1:10, type="l", lwd=5, lty=3)

The line is solid by default. Use the lty parameter with a value from **0 to 6** to specify the line format.

## Multiple Lines

To display more than one line in a graph, use the plot() function together with the lines() function:

line1 <- c(1,2,3,4,5,10)  
line2 <- c(2,5,7,8,9,10)  
plot(line1, type = "l", col = "blue")  
lines(line2, type="l", col = "red")

## Scatter Plots

A "scatter plot" is a type of plot used to display the relationship between two numerical variables, and plots one dot for each observation.

It needs two vectors of same length, one for the x-axis (horizontal) and one for the y-axis (vertical):

x <- c(5,7,8,7,2,2,9,4,11,12,9,6)  
y <- c(99,86,87,88,111,103,87,94,78,77,85,86)  
plot(x, y, main="Observation of Cars", xlab="Car age", ylab="Car speed")

* Q1: Create a bar plot of the number of cylinders (cyl) in the mtcars dataset. Use different colors to represent the transmission type (am). Add appropriate title, labels, and legend to the plot.
* Q2: Create a histogram of the miles per gallon (mpg) in the mtcars dataset. Use different shades of blue to represent the frequency of each bin. Add appropriate title and labels to the plot. Calculate and display the mean and standard deviation of mpg on the plot.
* Q3: Create a box plot of the horsepower (hp) in the mtcars dataset. Use different shapes to represent the number of gears (gear). Add appropriate title, labels, and legend to the plot. Identify and label any outliers on the plot.
* Q4: Create a scatter plot of the displacement (disp) versus the weight (wt) in the mtcars dataset. Use different colors and sizes to represent the number of carburetors (carb). Add appropriate title, labels, and legend to the plot. Add a smooth line to show the trend of the relationship.
* Q5: Create a map of India using the maps package. Use different colors to represent the literacy rate of each state. Add appropriate title, labels, and legend to the map. Add a text annotation to show the name and literacy rate of Kerala.
* Q1: How can you create a scatterplot of the Sepal.Length and Petal.Length variables in the iris dataset using the plot function? Add appropriate labels and title to the plot.
* Q2: How can you create a scatterplot of the mpg and disp variables in the mtcars dataset using the ggplot2 package? Use different colors to represent the cyl variable and add a smooth line to show the trend. Add appropriate labels, title, and legend to the plot.

## Pie Charts

A pie chart is a circular graphical view of data.

Use the pie() function to draw pie charts:

x <- c(10,20,30,40)  
eg 1  
# Display the pie chart  
pie(x)

eg 2

# Create a vector of pies  
x <- c(10,20,30,40)  
  
# Display the pie chart and start the first pie at 90 degrees  
pie(x, init.angle = 90)

Eg 3

# Create a vector of pies  
x <- c(10,20,30,40)  
  
# Create a vector of labels  
mylabel <- c("Apples", "Bananas", "Cherries", "Dates")  
  
# Display the pie chart with labels

pie(x, label = mylabel, main = "Fruits")

eg 4

# Create a vector of colors  
colors <- c("blue", "yellow", "green", "black")  
  
# Display the pie chart with colors  
pie(x, label = mylabel, main = "Fruits", col = colors)

Eg 5

# Create a vector of colors  
colors <- c("blue", "yellow", "green", "black")  
  
# Display the pie chart with colors  
pie(x, label = mylabel, main = "Fruits", col = colors)

legend("bottomright", mylabel, fill = colors)

## Bar Charts

A bar chart uses rectangular bars to visualize data. Bar charts can be displayed horizontally or vertically. The height or length of the bars are proportional to the values they represent.

Use the barplot() function to draw a vertical bar chart:

# x-axis values  
x <- c("A", "B", "C", "D")  
  
# y-axis values  
y <- c(2, 4, 6, 8)  
  
barplot(y, names.arg = x)

Eg 2

Use the col parameter to change the color of the bars:

x <- c("A", "B", "C", "D")  
y <- c(2, 4, 6, 8)  
  
barplot(y, names.arg = x, col = "red")

eg 3

## Horizontal Bars

If you want the bars to be displayed horizontally instead of vertically, use horiz=TRUE:

### **Example**

x <- c("A", "B", "C", "D")  
y <- c(2, 4, 6, 8)  
  
barplot(y, names.arg = x, horiz = TRUE)

Histogram

R creates histogram using **hist()** function. This function takes a vector as an input and uses some more parameters to plot histograms

The basic syntax for creating a histogram using R is −

hist(v,main,xlab,xlim,ylim,breaks,col,border)

* **main** indicates title of the chart.
* **col** is used to set color of the bars.
* **border** is used to set border color of each bar.
* **xlab** is used to give description of x-axis.
* **xlim** is used to specify the range of values on the x-axis.
* **ylim** is used to specify the range of values on the y-axis.
* **breaks** is used to mention the width of each bar.

# Create data for the graph.

v <- c(9,13,21,8,36,22,12,41,31,33,19)

# Create the histogram.

hist(v,xlab = "Weight",col = "yellow",border = "blue")

# Create data for the graph.

v <- c(9,13,21,8,36,22,12,41,31,33,19)

# Create the histogram.

hist(v,xlab = "Weight",col = "green",border = "red", xlim = c(0,40), ylim = c(0,5),

breaks = 5)

**Boxplot**

Boxplots are a measure of how well distributed is the data in a data set. It divides the data set into three quartiles. This graph represents the minimum, maximum, median, first quartile and third quartile in the data set. It is also useful in comparing the distribution of data across data sets by drawing boxplots for each of them.

Boxplots are created in R by using the **boxplot()** function.

### **Syntax**

boxplot(x, data, notch, varwidth, names, main)

Following is the description of the parameters used −

* **x** is a vector or a formula.
* **data** is the data frame.
* **notch** is a logical value. Set as TRUE to draw a notch.
* **varwidth** is a logical value. Set as true to draw width of the box proportionate to the sample size.
* **names** are the group labels which will be printed under each boxplot.
* **main** is used to give a title to the graph.

input <- mtcars[,c('mpg','cyl')]

boxplot(mpg ~ cyl, data = mtcars, xlab = "Number of Cylinders",

ylab = "Miles Per Gallon", main = "Mileage Data")

