Bar Graph Tutorial

Myat Phyo, Melissa Lavides, John Wolters

#Bar Graphs are a generically useful tool for the comparison of the quantities of one variable, or in more complex cases, multiple variables

#To create a Bar Graph in R the barplot() function is used

?barplot

# To get a basic idea about how the barplot function works in the R use this to look at the function on the R website

setwd("C:/r")

# First you need to set the directory to the one that you want to be working from.

# In this case, it is set to local disk at the folder named "r".

#For this example we will use a dataset called autos.

#Please download the autos.txt file into your working directory

#To read it into R do the following:

autos=read.table("autos.txt",header=T)

autos

```
  cars trucks suvs
1    1      2    4
2    3      5    4
3    6      4    6
4    4      5    6
5    9     12   16
```

# Now you are retrieving the text file named car from the directory "r".

# And at the same time you change the name of it to "autos"

# header=T is added at the end to remove the V1 V2 V3 column name.

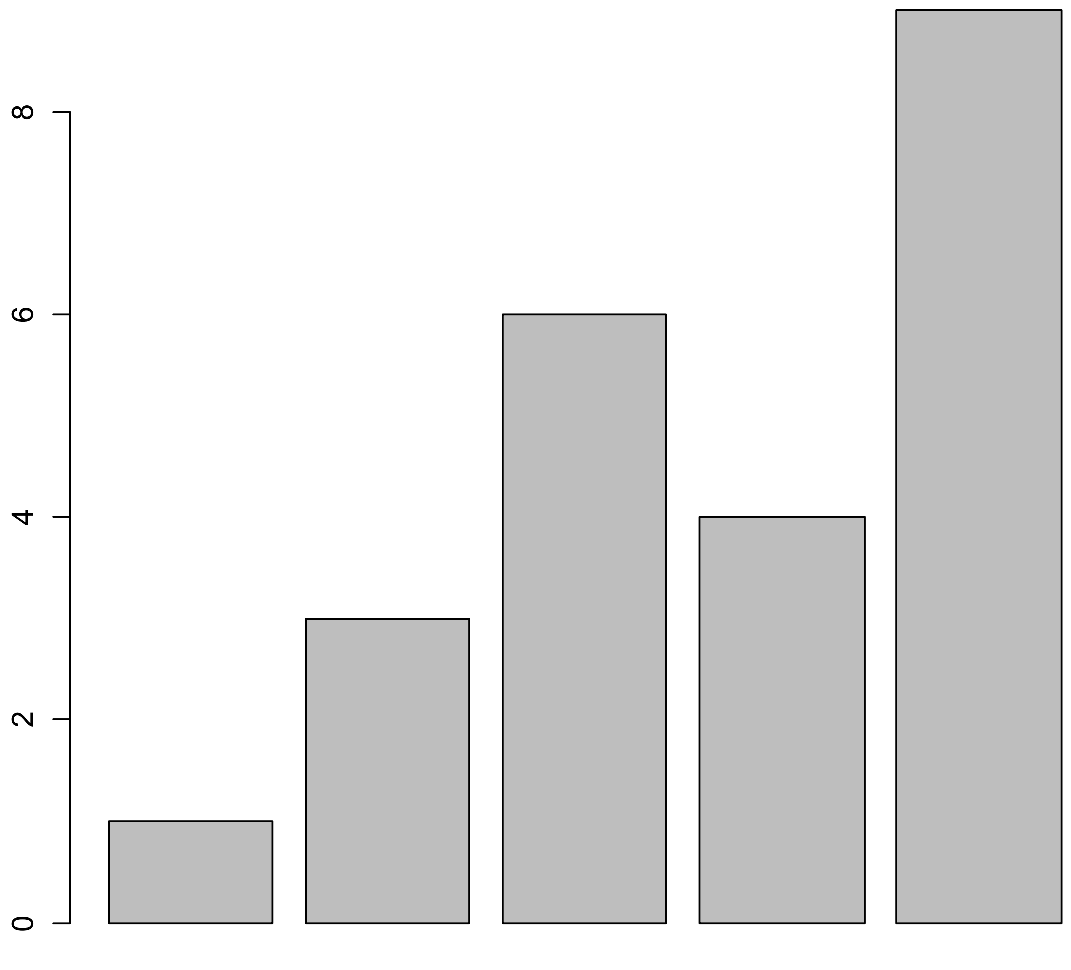
#Bar graphs mostly allow only one vector at a time to be plotted. For example, in our data set of "auto", we can only plot one vector: either car or suv or trucks. However using R it is possible to generate bar graphs showing an entire data set with multiple variables at once.

#For a few quick examples of that can be done with a bar graph let's look just at the cars variable in autos

attach(autos)

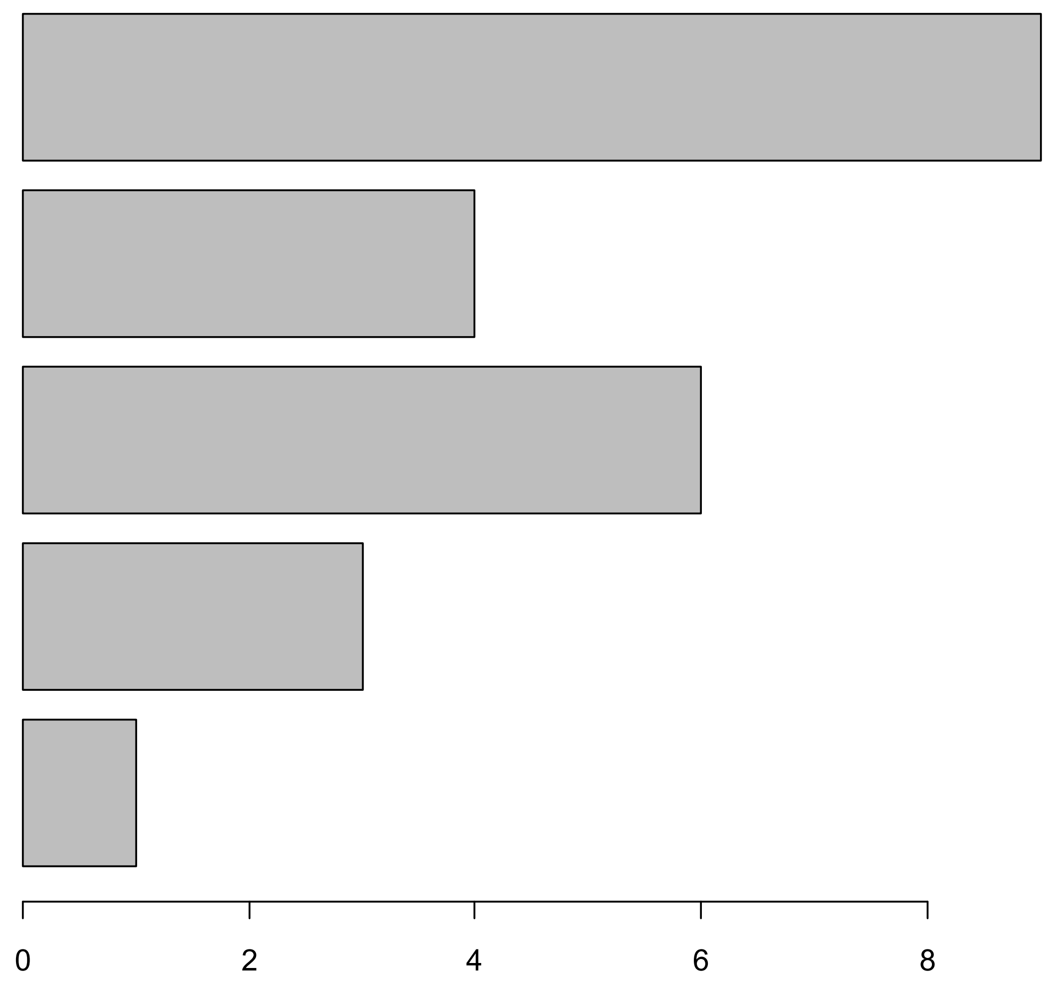
#Note that this will mask the cars variable that comes preinstalled in R. That is nothing to be concerned about. It simply means if you type cars while autos is attached, you will be referring to cars within autos, not the other variable.

barplot(cars)

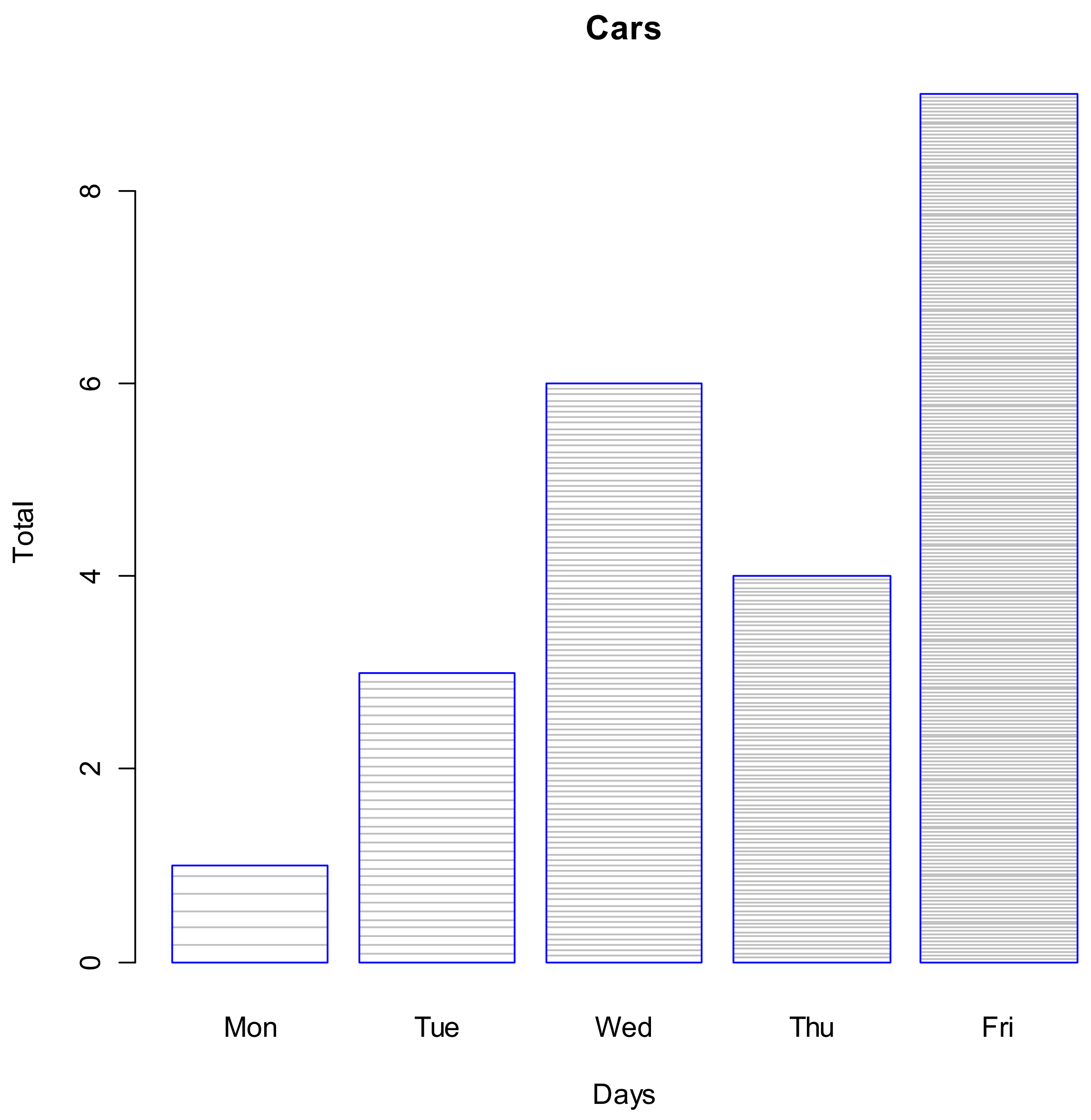


#The bar Graph can also be drawn with the bars horizontal by adding horiz=TRUE

```
barplot(cars, horiz=TRUE)
```



```
#Lets look at some of the ways you can label a bar graph

barplot(cars, main="Cars", xlab="Days",

  ylab="Total", names.arg=c("Mon","Tue","Wed","Thu","Fri"),

  border="blue", density=c(10,20,30,40,50), angle=180)

# main="Cars" allows us to give the graph an overhead title saying Cars

# the xlab and ylab allow us to set labels for the x and y axes.

# border="blue" set the border of the bars to the color blue. The border lines can also be colored
according to your taste.

# The names.arg function allows to specify names for each individual bar, in this case "monday,
tuesday...etc". Be sure when using names.arg that the number of names matches the number of bars

# Bar Graphs are extremely useful when data is separated into rows based on times such as hours, days,
or months

# The density function allows us to set different densities of diagonal lines for each individual bar. Note
how there are more lines in each subsequent bar, this is because we increased the density value as we
went across the bars. Just as with names.arg make sure you specify the same amount of densities as
there are bars

# angle allows you to change the angle of the density bars. In our case we made them into horizontal
lines by setting to 180 degrees.
```


Cars
Total
0
2
4
6
8
Mon
Tue
Wed
Thu
Fri
Days

#As opposed to names.arg we could also identify our graph by using colors

```
barplot(cars, main="Cars", xlab="Days",

  ylab="Total", col=rainbow(5))
```

# By adding col=rainbow(5) we have set the colors to a repeating pattern of the colors of the rainbow, for 5 colors

# Now we have to add a legend so that the reader will know to what these colors refer

```
legend("topleft", c("Mon","Tue","Wed","Thu","Fri"), cex=1, bty="n", fill=rainbow(5))
```

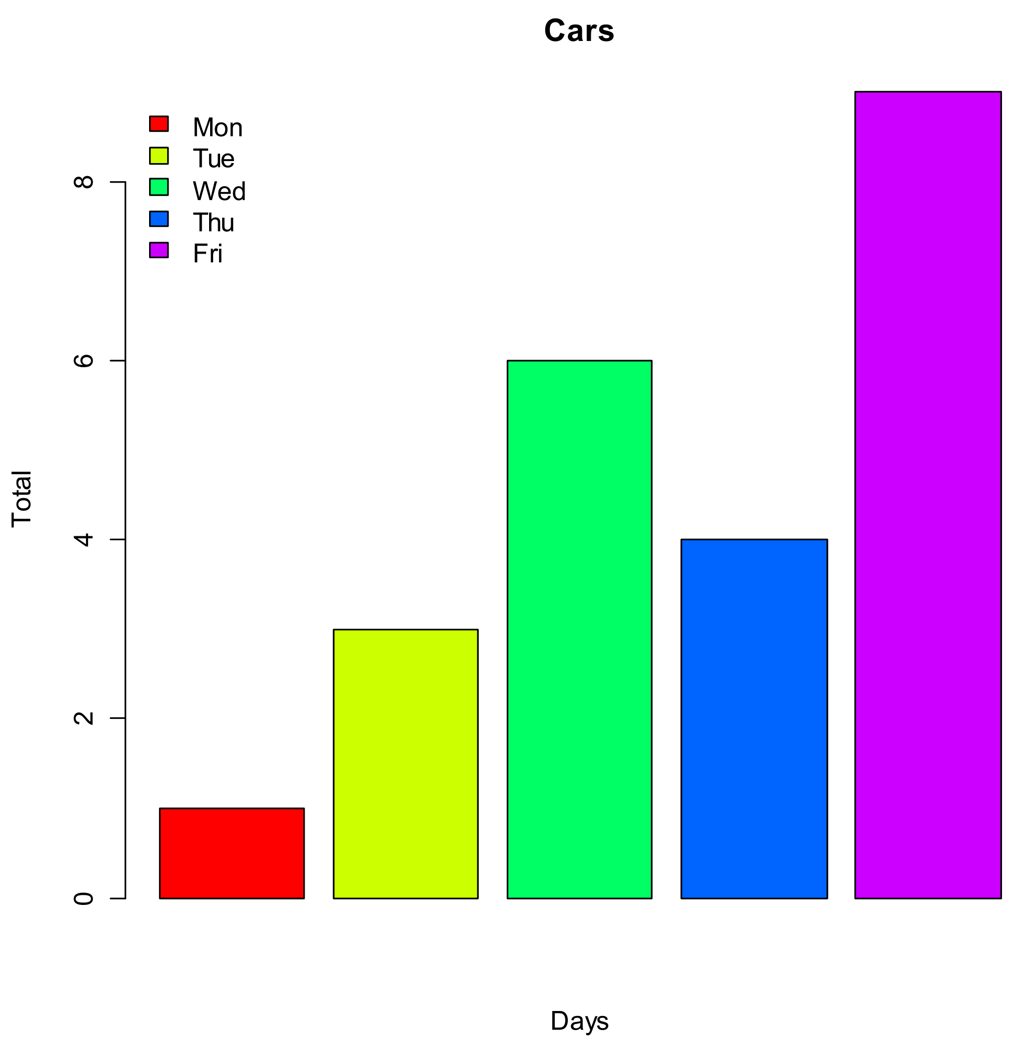
# The legend function will add a legend to your function

# The first part dictates a location, the second dictates the name

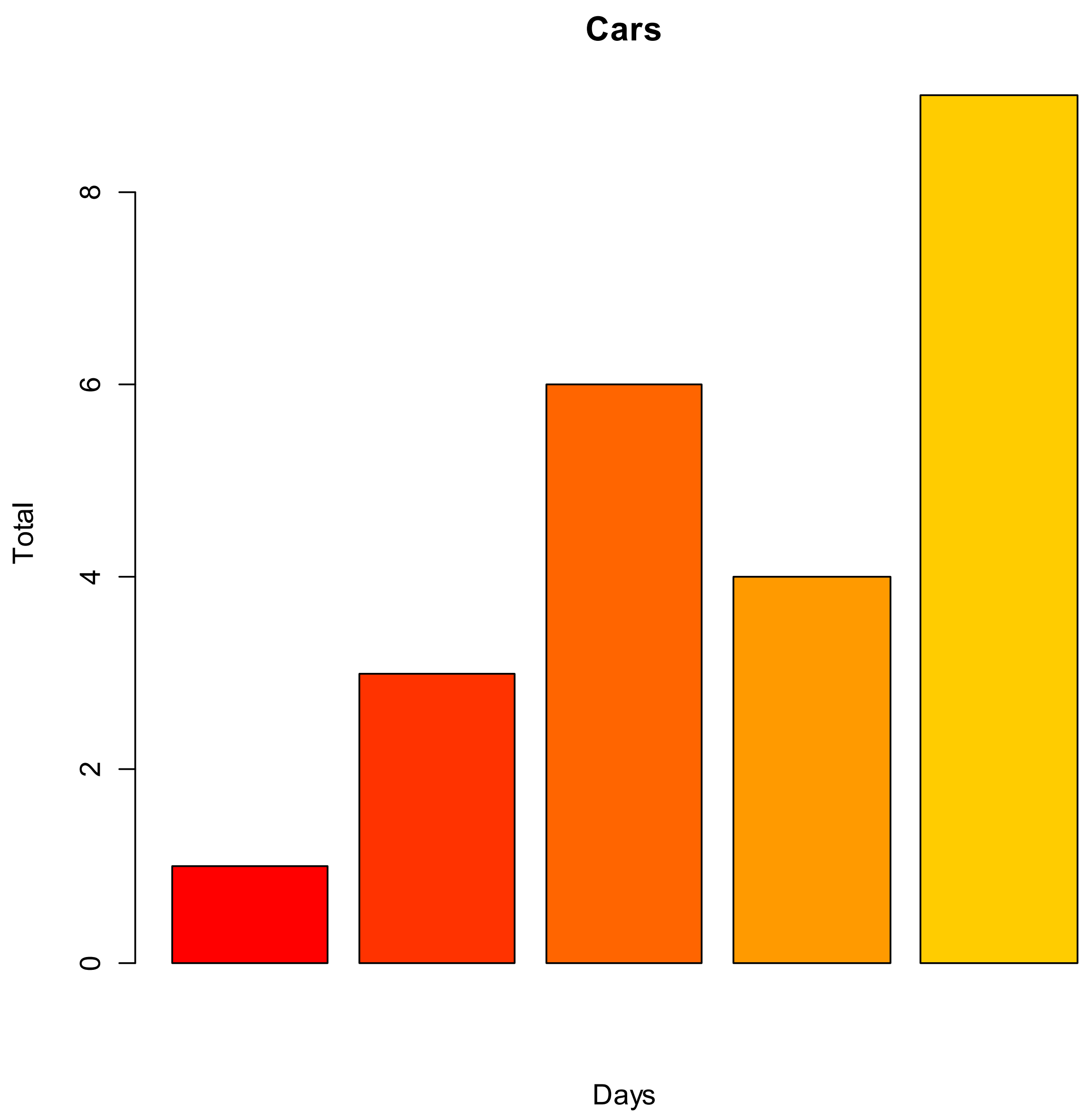
# cex dictates the size of the legend compared to the graph, make this number bigger or smaller to change the size of the legend

# bty can equal "n" or "o". "n" means no box around the legend while "o" adds a box around it.

# fill=rainbow(5) fills the colors for the different names. Make sure you use the same number you used in your bar graph


Cars
Mon
Tue
Wed
Thu
Fri
Total
0
2
4
6
8
Days

```
# The number you can put into colors is highly variable, anywhere from 5 to 10 to 100 to even 1000

# If you are using it for labeling purposes it is best to use the same number as the number of bars

# By changing the number you can create some cool gradients like the following

barplot(cars, main="Cars", xlab="Days",

  ylab="Total", col=rainbow(30))
```



# Now let's learn how to graph the entire dataset of autos at once

#When making bar graphs for multiple vectors on one plot there are two basic ways to go about it

#The first is to graph the different vectors side by side

#Lets graph the entire autos dataset with colors and a legend

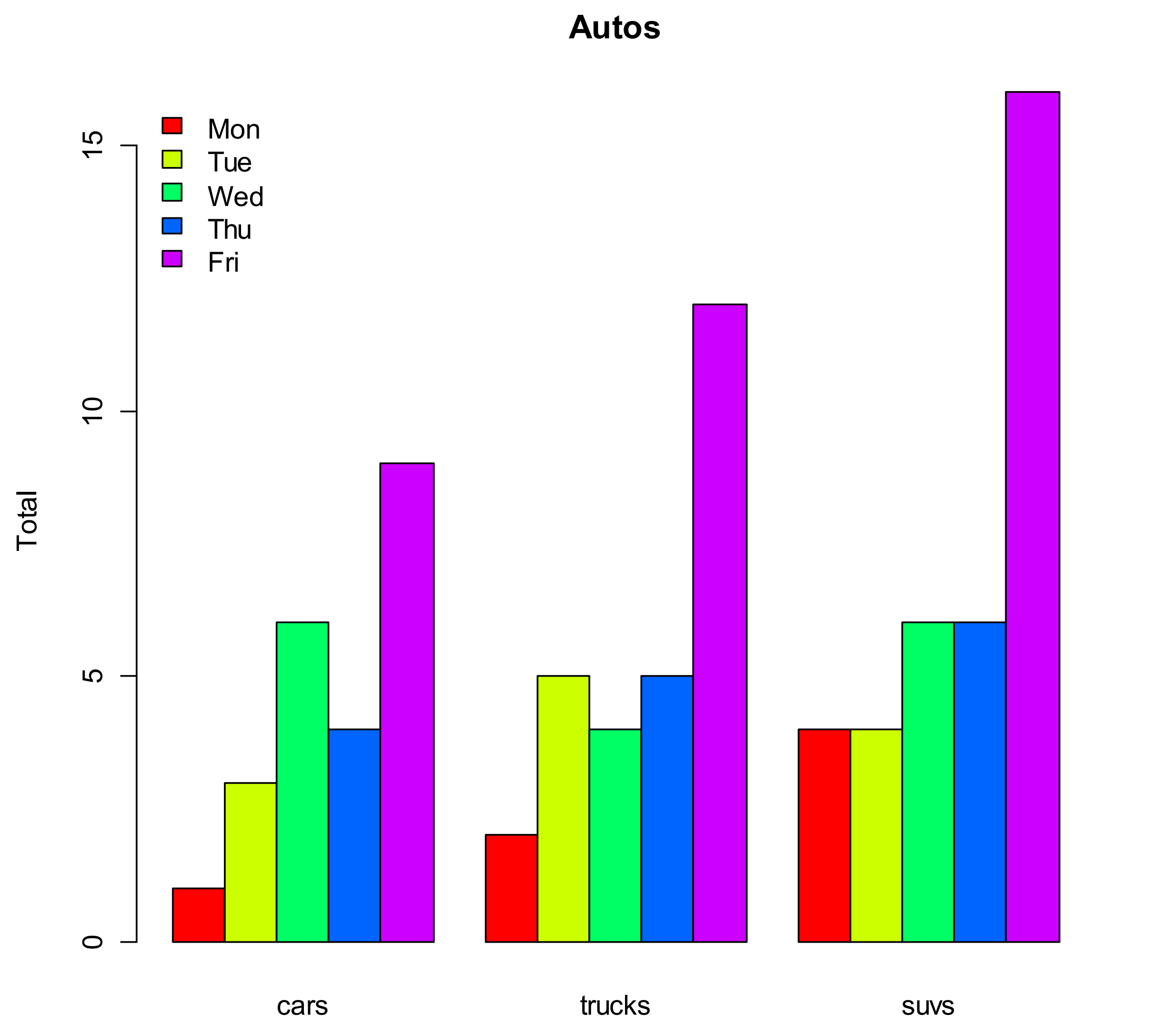
```
barplot(as.matrix(autos), main="Autos", ylab= "Total", beside=TRUE, col=rainbow(5))

legend("topleft", c("Mon","Tue","Wed","Thu","Fri"), cex=1, bty="n", fill=rainbow(5))
```

# When graphing a dataset with multiple vectors it is necessary to specify to graph the matrix using as.matrix

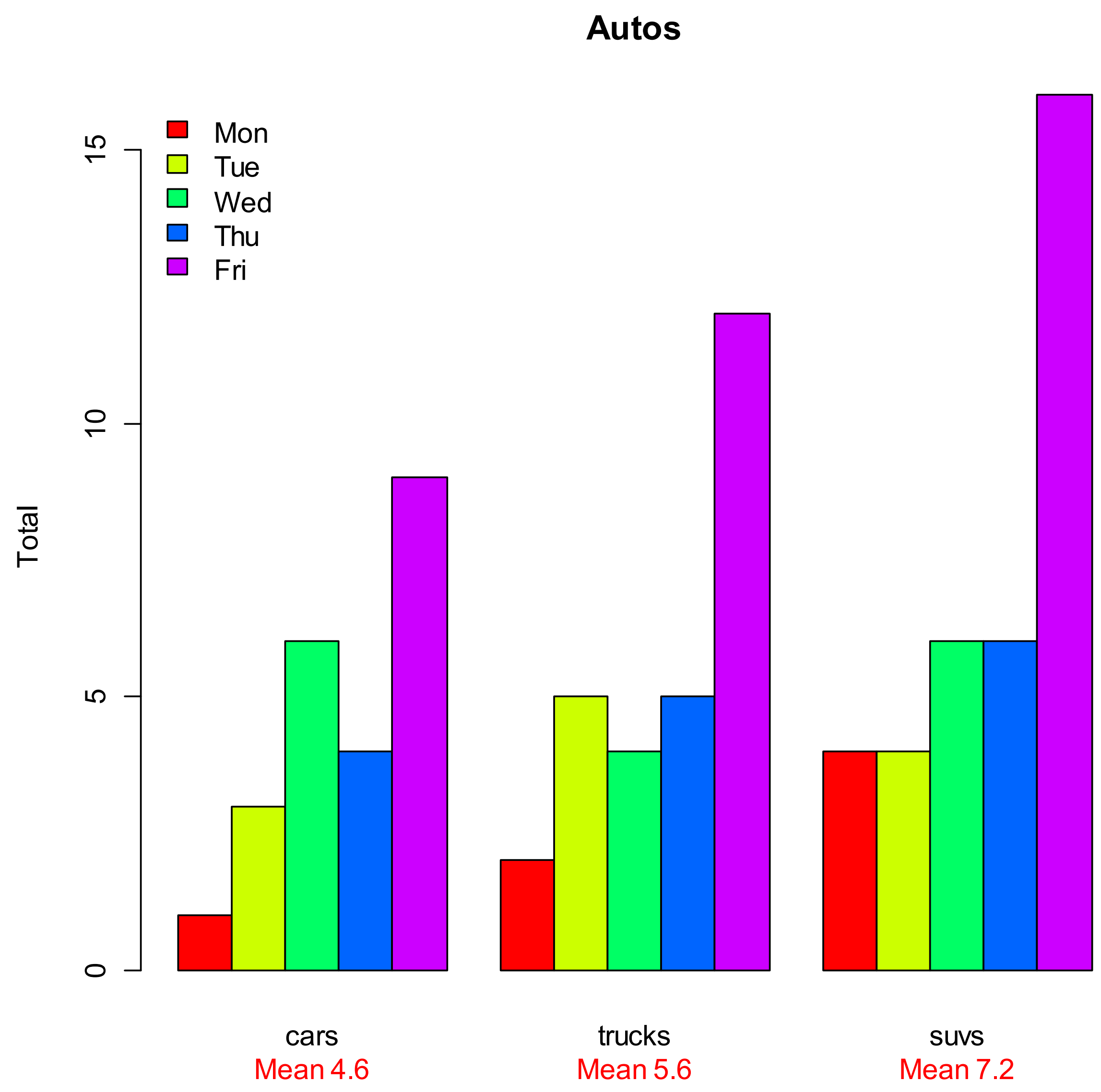
# names.arg can be used to specify names for each group of bars but if omitted the function just takes the column names from the columns of your data set.

# Note how we still used five for the rainbow. Because the number of bars was identical for each group, and we wanted each group to have the same pattern, a repeating set of 5 worked perfectly.

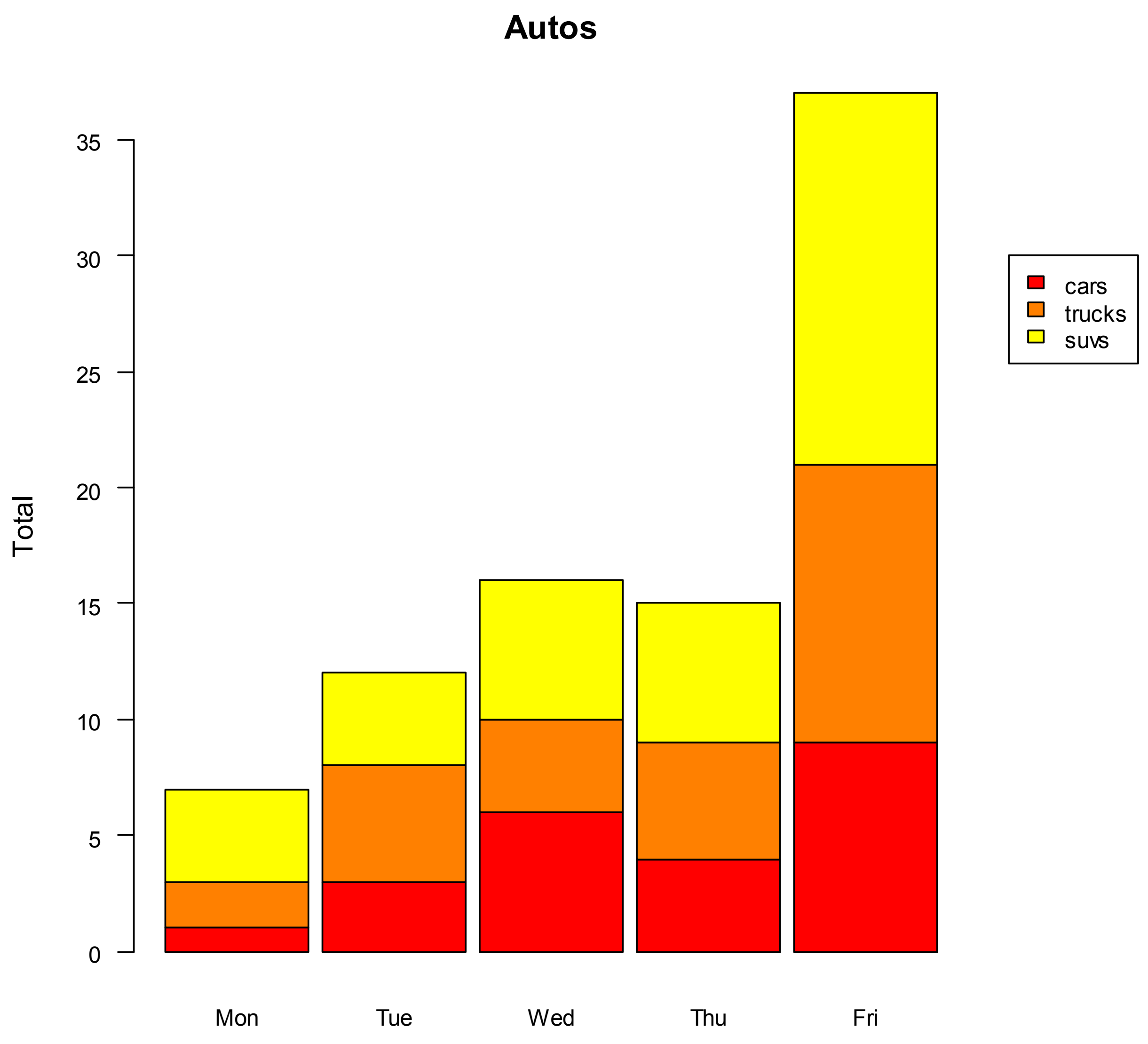
# For your own graphs this may take some experimentation, but be aware that the rainbow function follows this kind of repeating pattern.


Autos
Mon
Tue
Wed
Thu
Fri
Total
0
5
10
15
cars
trucks
suvs

```
# You can add additional material to a bar graph such as the means of the different vectors

# To do this we use the mtext function

barautos= barplot(as.matrix(autos), main="Autos", ylab= "Total", beside=TRUE, col=rainbow(5))

#By setting the graph to a variable we can refer to it as a location for a text function

mtext(side = 1, at = colMeans(barautos), line = 2, text = paste("Mean", colMeans(autos)), col = "red")

# side determines whether the text goes above, below, left, or right of the location

# at determines the location, set it equal to the same thing input into the text function

# line determines margins, if 2 does not work for your graph you can try different values

# text determines what text we are placing on the graph, first put the actualy text and then the
argument for what else you want displayed with the text

# Since we want to show the means we created a vector, a column, of the means of all the different
vectors in autos using the colMeans function

# There are similar colfunctions, do ?colMeans to take a look at other options

# col sets the color of the text, in this case to red

# It is often difficult to display a large amount of information on a bar graph so by adding things such as
the means we can add a lot of information to our bar graphs

# To double check the means simply enter them into R

mean(cars)

[1] 4.6

> mean(trucks)

[1] 5.6

> mean(suvs)

[1] 7.2
```


Autos
Mon
Tue
Wed
Thu
Fri
Total
0
5
10
15
cars
Mean 4.6
trucks
Mean 5.6
suvs
Mean 7.2

```
# The alternative to putting the groups like this is to create a stacked bar chart

# You could use a stacked bar chart:

# First you would need to expand the right margins so that there is room for a legend:

par(xpd=T, mar=par()$mar+c(0,0,0,4))

# Now this would give you the stacked bar chart with all the options described above:

barplot(t(autos), main="Autos", ylab="Total",

   col=heat.colors(3), space=0.1, cex.axis=0.8, las=1,

   names.arg=c("Mon","Tue","Wed","Thu","Fri"), cex=0.8)

# t(autos) creates a transposed version of the autos dataset so that it is grouped differently

# instead of grouping by cars, suvs, or trucks we are instead grouping by day

# heat.colors functions the same way rainbow does but with a different set of colors, these are more
focused on reddish colors

# space adds a small space between the bars, this can be increased or decreased according to your own
graph's needs

# cex.axis works just like the cex function outlined earlier but instead changes the size of the names of
the axes

# las Change the style of the axis label using las= 0: always parallel to the axis, [default], 1: always
horizontal, 2: always perpendicular to the axis, 3: always vertical

#Now add a legend

legend(6, 30, names(autos), cex=0.8, fill=heat.colors(3))

# The 6 and 30 are x and y coordinates used to move the legend to the right of the graph

# This different type of graph and grouping can be very helpful in interpretation, for example this graph
makes it clearer that the predominance of automobiles are seen on the road on Fridays.
```



# Now that you have reformatted the bar graph spacing you would need to take if off if you want it to go back to normal do the following:

par(mar=c(5, 4, 4, 2) + 0.1)

#And that is a lot ways to make simple and informative bar graphs in R!