R Markdown for HW1 Submission

Kritika Raghuwanshi, Anandita Jain, Raj Shah

2023-01-29

Problem 1

Answer 1a:

To summarize the main statistics of all the variables, I used the summary() function:

```
summary(attitude)
```

```
##      rating        complaints     privileges       learning         raises
##  Min.   :40.00   Min.   :37.0   Min.   :30.00   Min.   :34.00   Min.   :43.00
##  1st Qu.:58.75   1st Qu.:58.5   1st Qu.:45.00   1st Qu.:47.00   1st Qu.:58.25
##  Median :65.50   Median :65.0   Median :51.50   Median :56.50   Median :63.50
##  Mean   :64.63   Mean   :66.6   Mean   :53.13   Mean   :56.37   Mean   :64.63
##  3rd Qu.:71.75   3rd Qu.:77.0   3rd Qu.:62.50   3rd Qu.:66.75   3rd Qu.:71.00
##  Max.   :85.00   Max.   :90.0   Max.   :83.00   Max.   :75.00   Max.   :88.00
##     critical        advance
##  Min.   :49.00   Min.   :25.00
##  1st Qu.:69.25   1st Qu.:35.00
##  Median :77.50   Median :41.00
##  Mean   :74.77   Mean   :42.93
##  3rd Qu.:80.00   3rd Qu.:47.75
##  Max.   :92.00   Max.   :72.00
```

Answer 1b:

There are 30 observations for 7 variables in the "attitude" dataset. I used the View() function in R to display this information

```
#View(attitude)
```

Alternatively, I also used the str() function, which displays the internal structure of the dataset and is an alternate to the summary() function, to display the same information

```
str(attitude)
```

```
## 'data.frame':    30 obs. of  7 variables:
##  $ rating    : num  43 63 71 61 81 43 58 71 72 67 ...
##  $ complaints: num  51 64 70 63 78 55 67 75 82 61 ...
##  $ privileges: num  30 51 68 45 56 49 42 50 72 45 ...
##  $ learning  : num  39 54 69 47 66 44 56 55 67 47 ...
##  $ raises    : num  61 63 76 54 71 54 66 70 71 62 ...
##  $ critical  : num  92 73 86 84 83 49 68 66 83 80 ...
##  $ advance   : num  45 47 48 35 47 34 35 41 31 41 ...
```

Finally, I also explored the option of displaying the same using the nrow() function, which returns the number of rows and ncol(), which displays the number of columns, as follows:

```
nrow(attitude)
```

```
## [1] 30
```

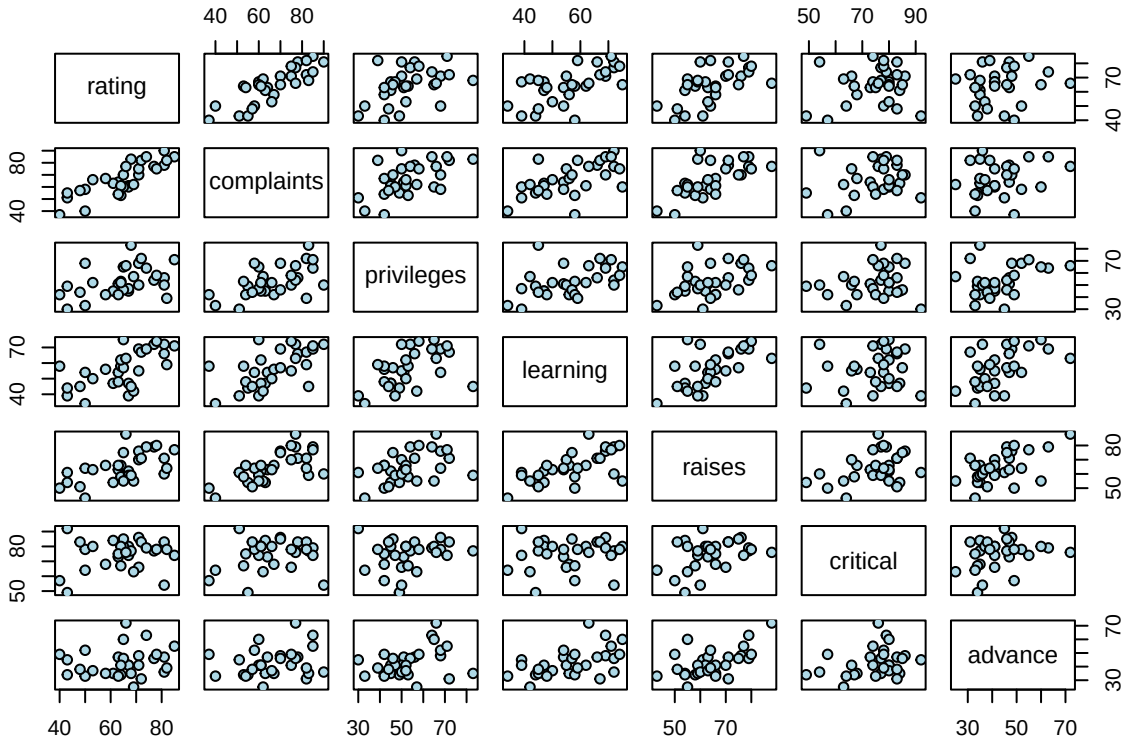
```
ncol(attitude)
```

```
## [1] 7
```

Answer 1c:

Scatterplot matrix can be created using the plot() function. Per the scatterplot, the most correlated variable with "rating" is "complaints" because the data in both columns is quite similar, making the plot points in the scatterplot also similar for both.

```
plot(attitude, pch = 21, bg = "lightblue")
```



Another way to create a scatterplot matrix is to use the pairs() function as shown below. Here attitude[1:7] covers all the variables (1 to 7) of the attitude dataset, pch is used for plotting 'character', and bg fills the background color for pch values 21:25, with Pch 21 corresponding to a "filled circle".

```
#pairs(attitude, pch = 21, bg = "lightblue")
```

Answer 1d:

To produce a scatterplot of rating (y-axis) v/s learning (x-axis), the plot() function can be used as shown below (commented). Here we're assigning the "learning" variable to the x-axis and the "rating" variable to the y-axis, with xlab and ylab being used to give labels to the x and y-axis respectively.

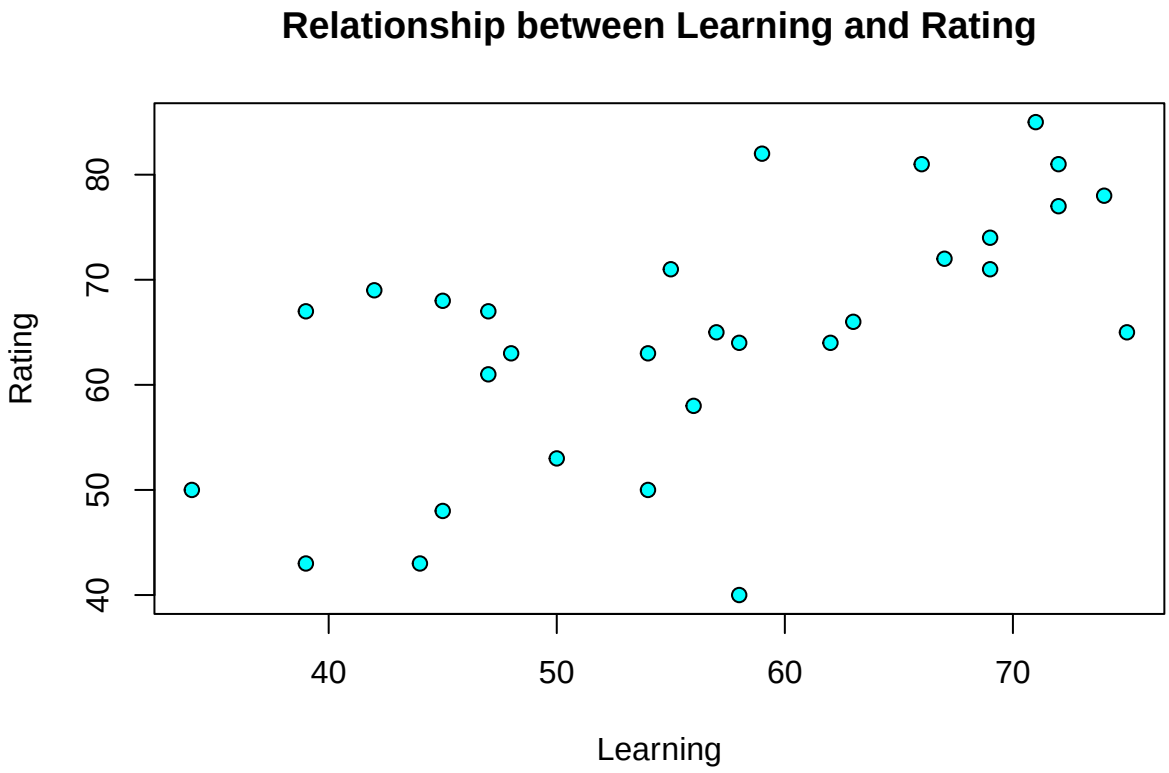
```
#plot(x=attitude$learning, y=attitude$rating, xlab = "Learning", ylab = "Rating")
```

To give a title to this plot, we can use two different methods, where we can either include the title using "main=" in the plot() function or we could use the title() function to do the same. For adding the title using the title() function, here's the code (commented):

```
#plot(x=attitude$learning, y=attitude$rating, xlab = "Learning", ylab = "Rating", pch = 21,
#bg = "cyan")
#title(main = "Relationship between Learning and Rating")
```

For adding title using the "main=" option, here's the code:

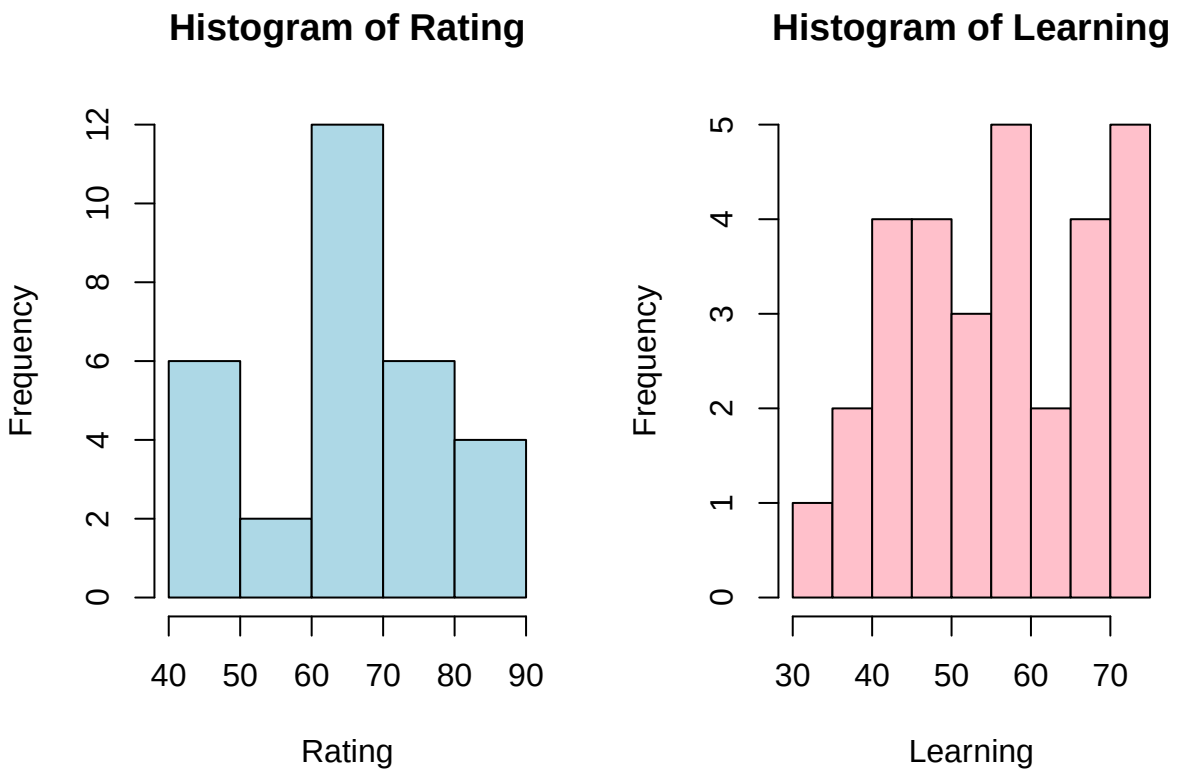
```
plot(x=attitude$learning, y=attitude$rating, xlab = "Learning", ylab = "Rating", pch = 21,
bg = "cyan", main = "Relationship between Learning and Rating")
```



Answer 1e:

To produce the histograms for "rating" and "learning" variables, we can use the hist() function. However, to show them side by side, the par(mfrow=...) function needs to be used as shown below. mfrow=c(1,2) displays the figures in 1 row and 2 columns. The hist(...) command is used to display the histogram for the "rating" and "learning" variables. "col =" is used to add color to the graph.

```
par(mfrow=c(1,2))
hist(attitude$rating, xlab="Rating", main = "Histogram of Rating", col = "lightblue")
hist(attitude$learning, xlab="Learning", main = "Histogram of Learning", col = "pink")
```



Problem 2

```
library("pacman")
library("rio")
library("tidyverse")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.1
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.3.0      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Loading the Data "hw1.xls" using the import function in rio package

```
df<-import("C:/hw1.xls")%>%
mutate(`Your gender` = as.factor(`Your gender`))%>%
rename(`Eat Fried Foods`=`How many times did you eat fried food per week during the past 7 days? (such a
head(df, n=10)
```

```
##    Your Weight in pounds Your Height in inches Your gender
## 1                    177                    73        Male
## 2                    180                    65      Female
## 3                    130                    70      Female
## 4                    123                    61      Female
## 5                    195                    72      Female
## 6                    170                    64      Female
## 7                    189                    69        Male
## 8                    175                    67      Female
## 9                    154                    69        Male
## 10                   135                    63      Female
##    On average, how many days do you exercise per week during this past month? (in days)
## 1                                                                                    5
## 2                                                                                    7
## 3                                                                                    4
## 4                                                                                    5
## 5                                                                                    1
## 6                                                                                    2
## 7                                                                                    3
## 8                                                                                    2
## 9                                                                                    4
## 10                                                                                   2
##    Have you been doing exercise regularly during this past year?
## 1                                                            Yes
## 2                                                            Yes
## 3                                                            Yes
## 4                                                             No
## 5                                                             No
```

```
## 6                                                                  No
## 7                                                                 Yes
## 8                                                                  No
## 9                                                                  No
## 10                                                                Yes
##      Eat Fried Foods
## 1               None
## 2  Less than 3 times
## 3  Less than 3 times
## 4               None
## 5  Less than 3 times
## 6  Less than 3 times
## 7               None
## 8  Less than 3 times
## 9   At least 3 times
## 10              None
```

Answer 2a:

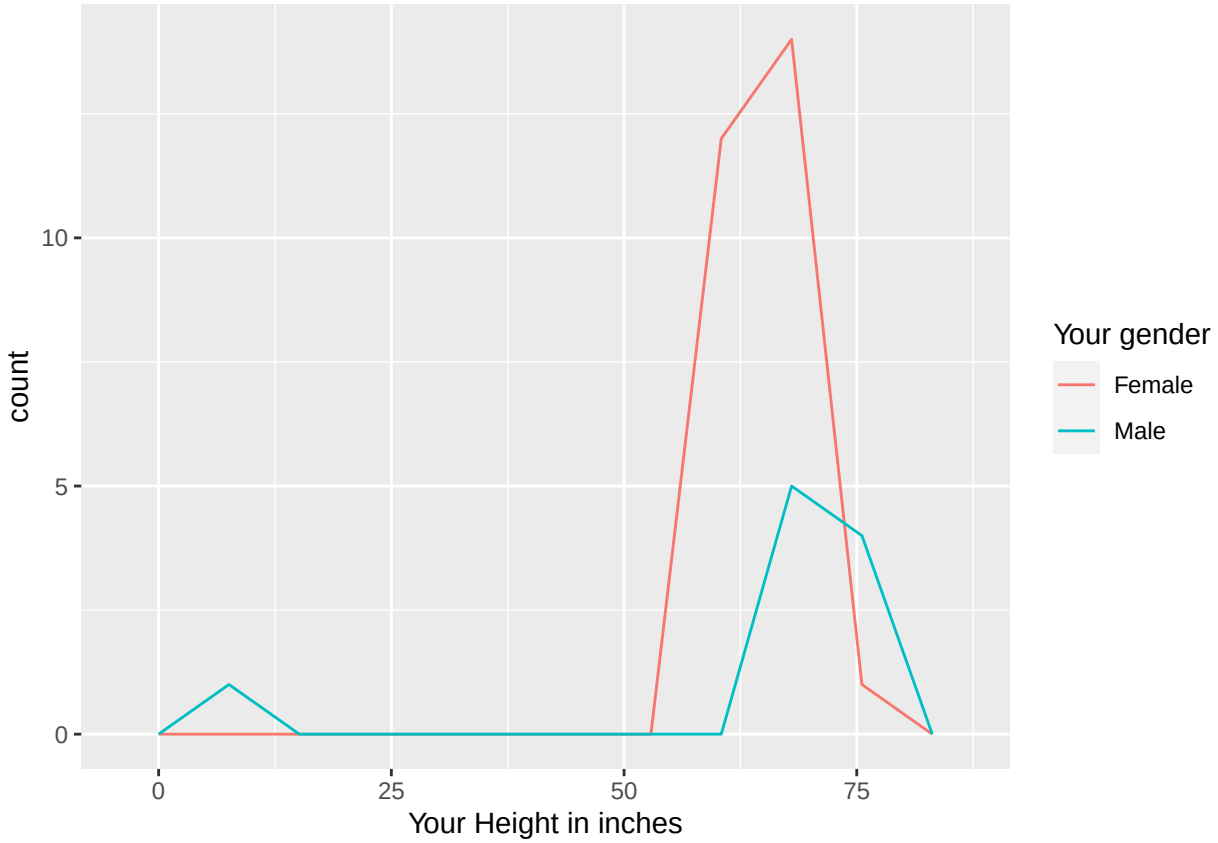
This graph shows the frequency distribution of variable 'Your Height in inches' with respect to two categorical variables Male and Female. The graph also shows that the number of Females are more than Male and also that the number of Females with higher Height are more than that of Male.

```
gender<-df$`Your gender`

p1_df<-as.data.frame(gender)
p1_df
```

```
##    gender
## 1    Male
## 2  Female
## 3  Female
## 4  Female
## 5  Female
## 6  Female
## 7    Male
## 8  Female
## 9    Male
## 10 Female
## 11 Female
## 12 Female
## 13 Female
## 14 Female
## 15 Female
## 16 Female
## 17 Female
## 18 Female
## 19   Male
## 20 Female
## 21 Female
## 22 Female
## 23 Female
## 24 Female
## 25 Female
## 26 Female
## 27 Female
## 28   Male
## 29 Female
## 30 Female
## 31 Female
## 32   Male
## 33   Male
## 34 Female
## 35   Male
## 36   Male
## 37   Male
```

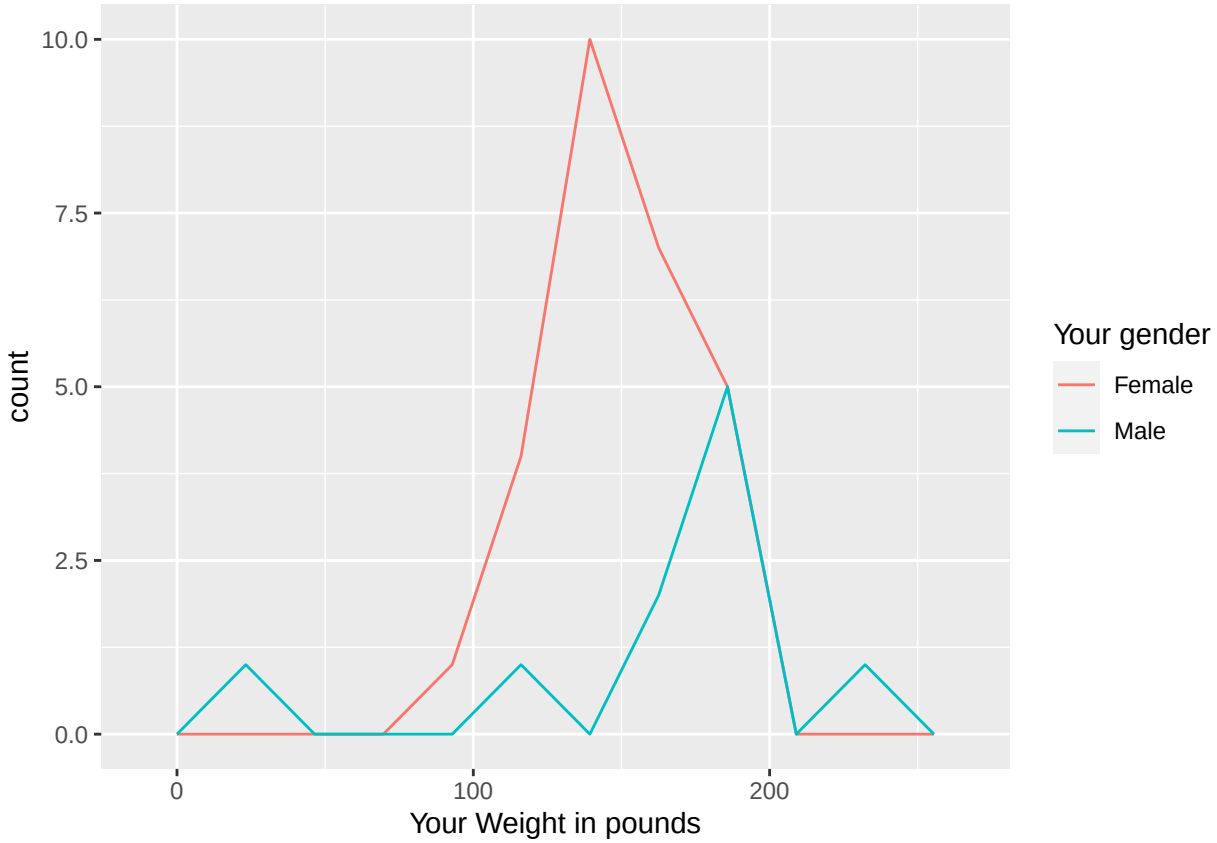
```
ggplot(df, aes(x=`Your Height in inches`, color = `Your gender`)) +
  geom_freqpoly(bins=10)
```



The below graph shows the frequency distribution of variable 'Your Weight in pounds' with respect to two categorical variables Male and Female. As it is seen in the graph, there are more number of Female with more Weight than Male.

To get these graphs, a ggplot function is used from the package ggplot2 in addition to function geom_freqpoly() which is used to plot a frequency distribution.

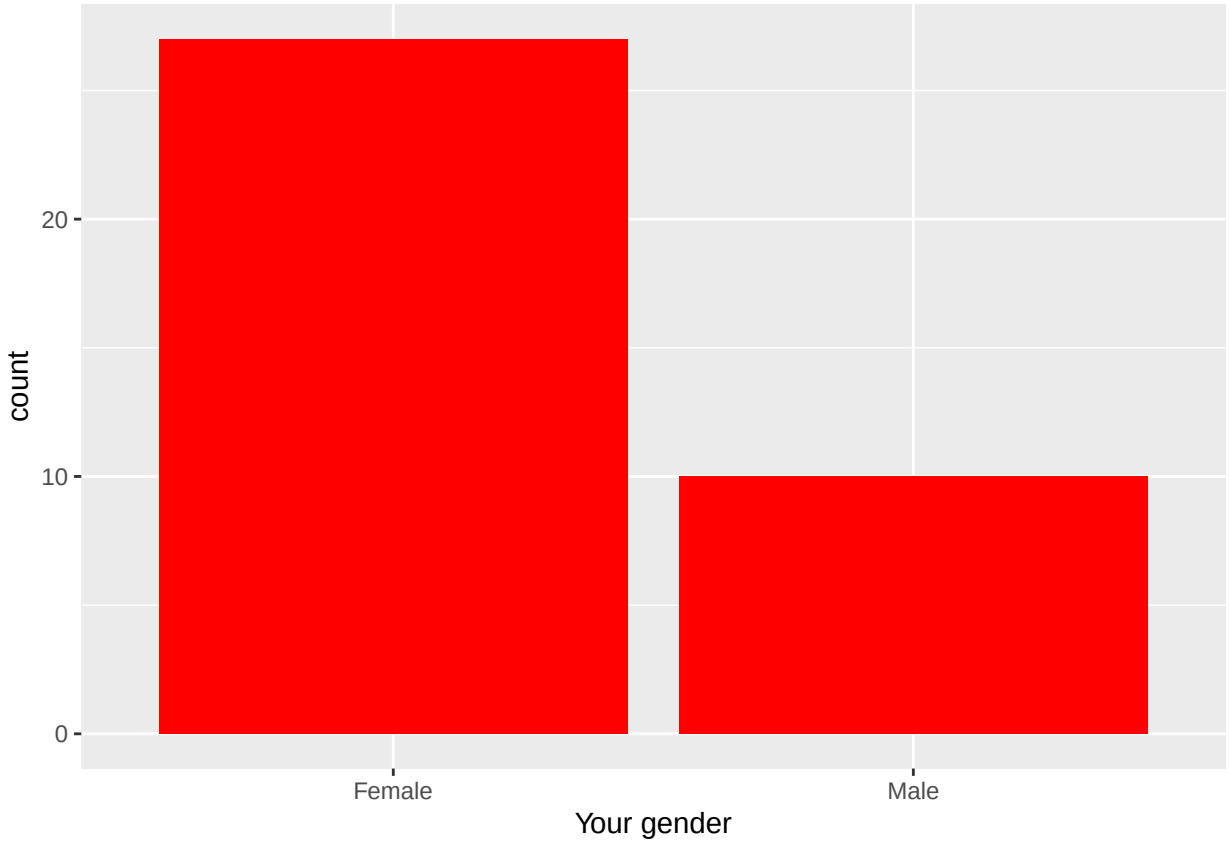
```
ggplot(df, aes(x=`Your Weight in pounds`, color = `Your gender`)) +
  geom_freqpoly(bins=10)
```


10.0
7.5
5.0
2.5
0.0
count
0
100
200
Your Weight in pounds
Your gender
Female
Male

Answer 2b:

The below graph shows the bar plot of categorical variable 'Your Gender'. According to the plot, the number of Female are 27 and number of Male are 10. So, there are more number of Females than Male.

```
p2<-ggplot(df, aes(x=`Your gender`)) + geom_bar(fill='red')
p2
```

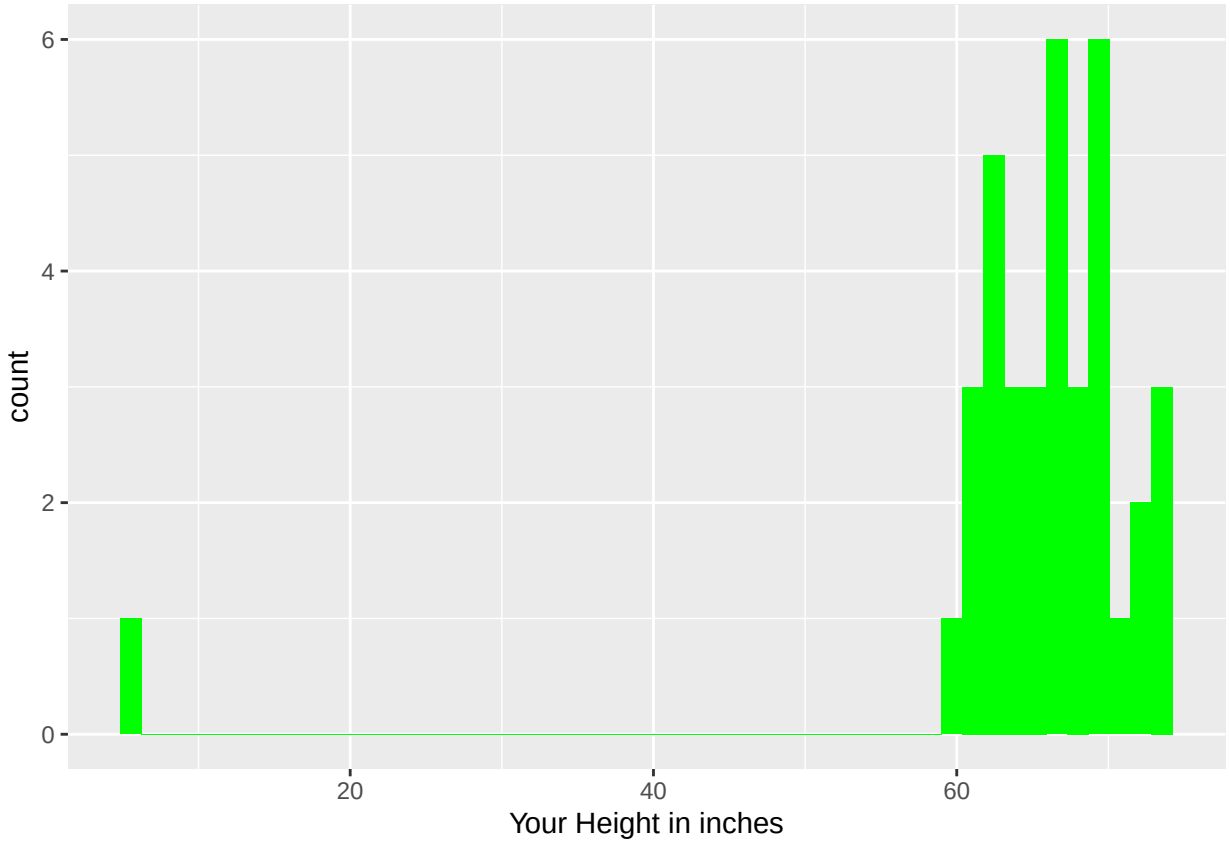


Answer 2c:

The below plot shows the Histogram of Variable 'Your Height in inches'. As it can be seen, the data for this variable is left skewed as most of the data points are above 60 inches. There is one outlier on the left hand side of graph with height less than 20 inches.

To get this plot, ggplot function from ggplot2 package was used with geom_histogram() function as it plots the histogram for the given variable.

```
p3<-ggplot(df,aes(x=`Your Height in inches`)) +
geom_histogram(fill='green', bins=50)
p3
```



Answer 2d:

The below graph is a side by side bar chart between variables 'Your Gender' and 'Eat fried Foods'. As we can see from the graph, maximum Male and Female do not Eat Fried Foods and stay healthy. The number of Male and Female eating fried food are same for the category 'Less than 3 times'. Apart from it, there are some Male that Eat fried foods more than 3 times.

To plot this graph, ggplot function was used with aes parameter where a groupby of two variables 'Your Gender' and 'Eat Fried Foods' is passed. This essentially counts the number of Male and Female in each category of 'Eat Fried Foods'.

```
p4<-ggplot(df,aes(`Your gender`,`Eat Fried Foods`)) +
  geom_bar(aes(fill=`Eat Fried Foods`),stat='identity', position = 'dodge')
p4
```

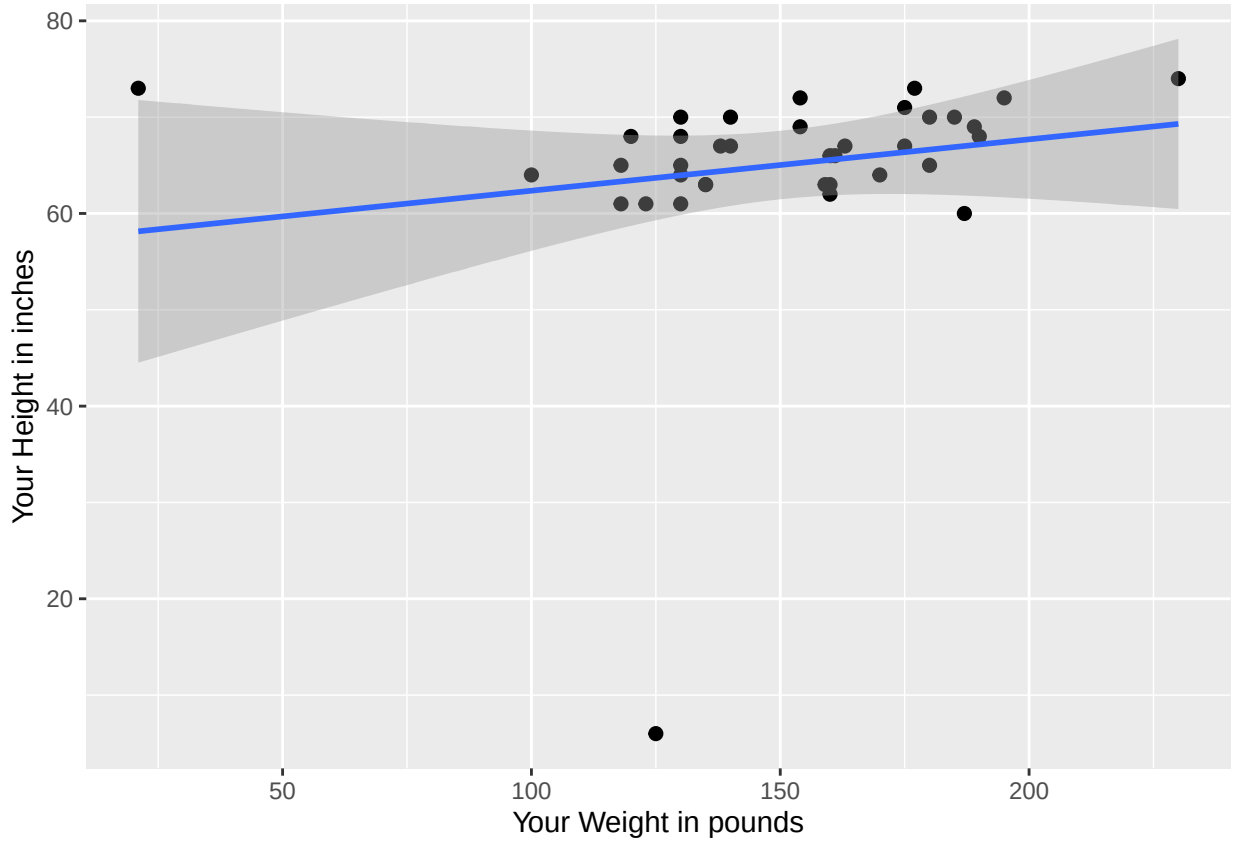
Answer 2e:

Below is the Scatter plot of 'Your Weight in Pounds' vs 'Your Height in Inches'. There is also a line passing through the graph that shows the strength of this relation. This is a positive correlation that with an increase in Weight, the Height also increases as data points are closely wounded around the line.

To plot this graph, ggplot was used with groupby feature used in aes parameter. In addition to it, geom_point() function was used to plot the scatter plot and to add the regression line geom_smooth() function with method=lm was used that gives the regression line between two numerical variables.

```
p5<-ggplot(df,aes(`Your Weight in pounds`,`Your Height in inches`)) +
geom_point(size=2) +
geom_smooth(method = lm)
p5
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

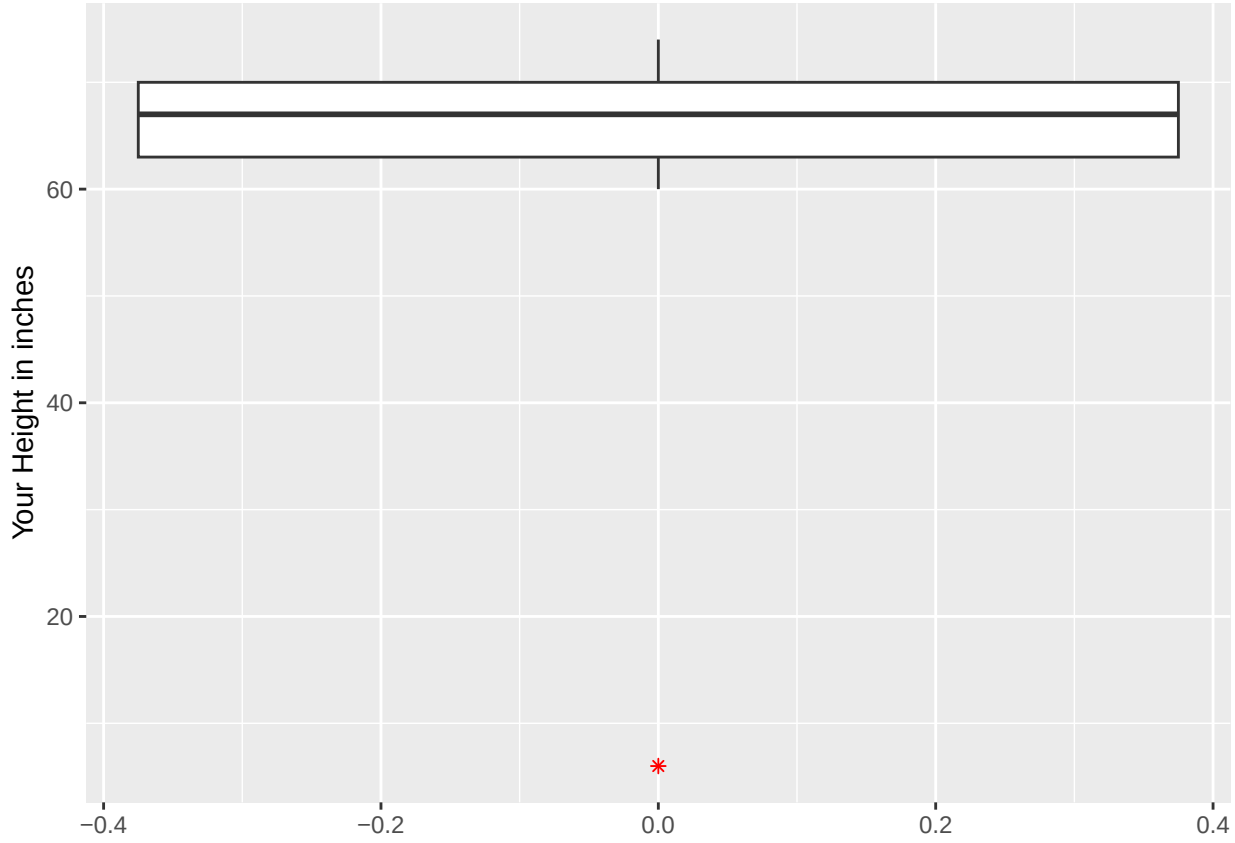


Answer 2f:

The Below graph depicts the Box Plot of variable 'Your Height in Inches'. Box plot is the Five Number summary of the variable and it also shows the outliers if present. In our graph, there is an outlier at the bottom of the graph with height less than 20. The Five number summary for this boxplot is as follows: Min = 60 inches, Max = 75 inches, Q1 = 63 inches, Q3 = 70 inches and Median = 67 inches. The figures are taken approximately from the graph.

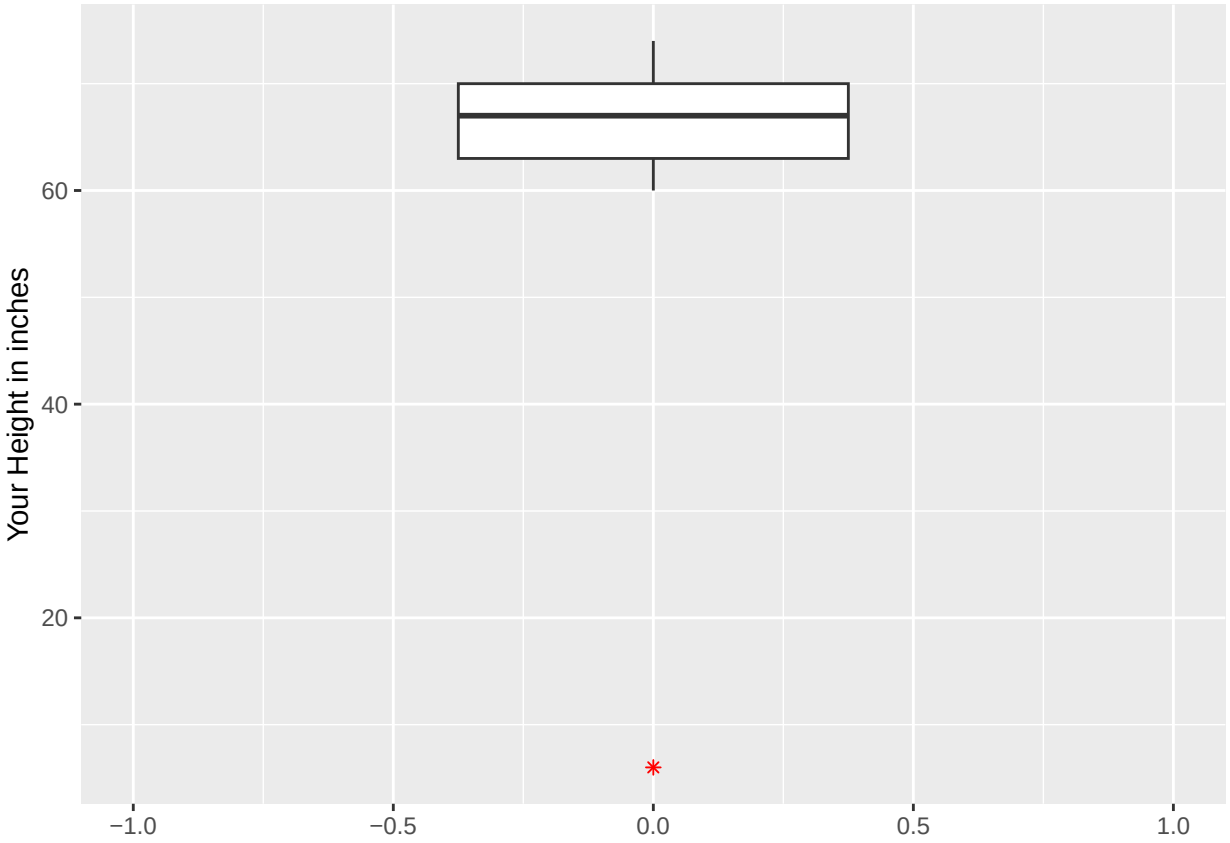
To plot this, ggplot function was used in addition to geom_boxplot() function. A parameter outlier.color was also used to show the outlier in different colour so that it is distinguished from others. The final graph is the last graph which we get by moderating the graphs limits so that we get the proper boxplot shape.

```
p6<-ggplot(df,aes(`Your Height in inches`)) +
  geom_boxplot(size=0.5,outlier.color = 'red',outlier.shape = 8,notch = FALSE) +
  coord_flip()
p6
```




```
p6+xlim(-1,1)
```


```
p6+ylim(-1,1)
```


Your Height in inches
60
40
20
−1.0
−0.5
0.0
0.5
1.0

Problem 3

Answer 3a:

For this problem, we've used the lapply() function, which allows you to apply a function (FUN) to the corresponding element of the dataset or all variables of the dataset - lapply(X, FUN, ...). First, we've loaded the MASS library and then used the "birthwt" dataset, followed by checking the structure of the dataset pre and post converting numericals to factors.

```
library(MASS)
library(tibble)
birthwt <- as_tibble(birthwt)
head (birthwt)
```

```
## # A tibble: 6 x 10
##     low   age   lwt  race smoke   ptl    ht    ui   ftv   bwt
##   <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
## 1     0    19   182     2     0     0     0     1     0  2523
## 2     0    33   155     3     0     0     0     0     3  2551
## 3     0    20   105     1     1     0     0     0     1  2557
## 4     0    21   108     1     1     0     0     1     2  2594
## 5     0    18   107     1     1     0     0     1     0  2600
## 6     0    21   124     3     0     0     0     0     0  2622
```

```
birthwt_factor <- data.frame(lapply(birthwt, factor)) #using lapply function
birthwt_factor <- as_tibble(birthwt_factor)
head (birthwt_factor)
```

```
## # A tibble: 6 x 10
##   low   age   lwt   race  smoke ptl   ht    ui    ftv   bwt
##   <fct> <fct> <fct> <fct> <fct> <fct> <fct> <fct> <fct> <fct>
## 1 0     19    182   2     0     0     0     1     0     2523
## 2 0     33    155   3     0     0     0     0     3     2551
## 3 0     20    105   1     1     0     0     0     1     2557
## 4 0     21    108   1     1     0     0     1     2     2594
## 5 0     18    107   1     1     0     0     1     0     2600
## 6 0     21    124   3     0     0     0     0     0     2622
```

Answer 3b:

Mutate function:

To use the function mutate, we installed the dplyr and MASS libraries, Followed by converting the "birthwt" dataset into a tibble data frame (for better clarity), and finally using the "as.factor()" function to mutate all the variables at the same time.

```
library(MASS)
library(tibble)
library(dplyr)
head(birthwt) #checking the dataset pre-conversion to tibble data frame
```

```
## # A tibble: 6 x 10
##     low   age   lwt  race smoke   ptl    ht    ui   ftv   bwt
##   <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
## 1     0    19   182     2     0     0     0     1     0  2523
## 2     0    33   155     3     0     0     0     0     3  2551
## 3     0    20   105     1     1     0     0     0     1  2557
## 4     0    21   108     1     1     0     0     1     2  2594
## 5     0    18   107     1     1     0     0     1     0  2600
## 6     0    21   124     3     0     0     0     0     0  2622
```

```
birthwt <- as_tibble(birthwt)
head(birthwt) #checking the dataset post-conversion to tibble data frame
```

```
## # A tibble: 6 x 10
##     low   age   lwt  race smoke   ptl    ht    ui   ftv   bwt
##   <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
## 1     0    19   182     2     0     0     0     1     0  2523
## 2     0    33   155     3     0     0     0     0     3  2551
## 3     0    20   105     1     1     0     0     0     1  2557
## 4     0    21   108     1     1     0     0     1     2  2594
## 5     0    18   107     1     1     0     0     1     0  2600
## 6     0    21   124     3     0     0     0     0     0  2622
```

```
birthwt %>% mutate(birthwt, low=as.factor(low), age=as.factor(age),
lwt=as.factor(lwt), race=as.factor(race), smoke=as.factor(smoke),
ptl=as.factor(ptl), ht=as.factor(ht), ui=as.factor(ui),
ftv=as.factor(ftv), bwt=as.factor(bwt))
```

```
## # A tibble: 189 x 10
##    low   age   lwt   race  smoke ptl   ht    ui    ftv   bwt
##    <fct> <fct> <fct> <fct> <fct> <fct> <fct> <fct> <fct> <fct>
##  1 0     19    182   2     0     0     0     1     0     2523
##  2 0     33    155   3     0     0     0     0     3     2551
##  3 0     20    105   1     1     0     0     0     1     2557
##  4 0     21    108   1     1     0     0     1     2     2594
##  5 0     18    107   1     1     0     0     1     0     2600
##  6 0     21    124   3     0     0     0     0     0     2622
##  7 0     22    118   1     0     0     0     0     1     2637
##  8 0     17    103   3     0     0     0     0     1     2637
```

```
##  9 0     29    123   1     1     0     0     0     1     2663
## 10 0     26    113   1     1     0     0     0     0     2665
## # ... with 179 more rows
```

Mapvalues function:

Since mapvalues() function is used to replace values, we could only change the following variables - low, race, smoke, ptl, ht, ui, and ftv. These are variables with values in the range of 0 to 4, which allows us to label them accordingly. In order to do this, we used the following commands:

```
library(plyr)
library(knitr)
birthwt$low <- mapvalues(birthwt$low, from = c("0", "1"),
to = c("Weight < 2.5kg", "Weight > 2.5kg"))
birthwt$race <- mapvalues(birthwt$race, from = c("1", "2", "3"),
to = c("White", "Black", "Other"))
birthwt$smoke <- mapvalues(birthwt$smoke, from = c("0", "1"),
to = c("Mother Does Not Smoke", "Mother Smokes"))
birthwt$ptl <- mapvalues(birthwt$ptl, from = c("0", "1"),
to = c("No Premature Labor", "Premature Labor"))
birthwt$ht <- mapvalues(birthwt$ht, from = c("0", "1"),
to = c("No Hypertension", "Have Hypertension"))
birthwt$ui <- mapvalues(birthwt$ui, from = c("0", "1"),
to = c("No UI", "Have UI"))
birthwt$ftv <- mapvalues(birthwt$ftv, from = c("0", "1", "2", "3", "4"),
to = c("No visits", "1st Visit", "2nd Visit", "3rd Visit", "4th Visit"))
kable(head(birthwt, n=7))
```

low	age	lwt	race	smoke	ptl	ht	ui	ftv	bwt
Weight < 2.5kg	19	182	Black	Mother Does Not Smoke	No Premature Labor	No Hypertension	Have UI	No visits	2523
Weight < 2.5kg	33	155	Other	Mother Does Not Smoke	No Premature Labor	No Hypertension	No UI	3rd Visit	2551
Weight < 2.5kg	20	105	White	Mother Smokes	No Premature Labor	No Hypertension	No UI	1st Visit	2557
Weight < 2.5kg	21	108	White	Mother Smokes	No Premature Labor	No Hypertension	Have UI	2nd Visit	2594
Weight < 2.5kg	18	107	White	Mother Smokes	No Premature Labor	No Hypertension	Have UI	No visits	2600
Weight < 2.5kg	21	124	Other	Mother Does Not Smoke	No Premature Labor	No Hypertension	No UI	No visits	2622
Weight < 2.5kg	22	118	White	Mother Does Not Smoke	No Premature Labor	No Hypertension	No UI	1st Visit	2637

Answer 3c:

Post loading the necessary packages - plyr, dplyr, tibble, and knitr, we created a few vectors to represent the bwt (BirthWeight), race (Race), and smoke (Smoke) variables. Converted Race and Smoke to factors, as race 1, 2, and 3 corresponds to some races, and similarly smoke values 0 an 1 also represent whether the mom smokes or not. Finally, using the tapply() function, calculated the mean / average birth weight of a child as requested.

```
library(MASS)
library(plyr)
library(dplyr)
library(tibble)
library(knitr)
remove(birthwt)#removing dataset to start afresh
BirthWeight <- birthwt$bwt
Race <- as.factor(birthwt$race)
Smoke <- as.factor(birthwt$smoke)
Output <- tapply(X = BirthWeight, INDEX = list(Race, Smoke), FUN = mean)
Output
```

```
##          0        1
## 1 3428.750 2826.846
## 2 2854.500 2504.000
## 3 2815.782 2757.167
```

* Per the data, we can see that yes, smoking does have an adverse effect on the weight of the baby.

* On an average, we see that the weight of the baby decreases if the mother smokes. This decrease in weight is also consistent among all races (1, 2, and 3).

* The main corelation between race and birth weight is that per the data, weight of the baby is more if the race of the mother is race 1 (White) compared to race 2 (Black) and race 3 (other).

- Comment: *Races 1, 2, and 3 have been denoted as White, Black, and Other per the description obtained using ?birthwt*

Answer 3d:

To display the table obtained above using the kable() function, we simply assigned the vector Output to this function as shown.

```
library(knitr)
kable(Output)
```

0	1
3428.750	2826.846
2854.500	2504.000
2815.782	2757.167

Answer 3e:

To apply the ddply function, we used the average/ mean of the birth weight (bwt), the summarise() function to summarise all observations in the input using a single row.

```
ddply (birthwt, ~ Race, summarize, Mean_Birth_Weight = mean(bwt))
```

```
##   Race Mean_Birth_Weight
## 1    1          3102.719
## 2    2          2719.692
## 3    3          2805.284
```

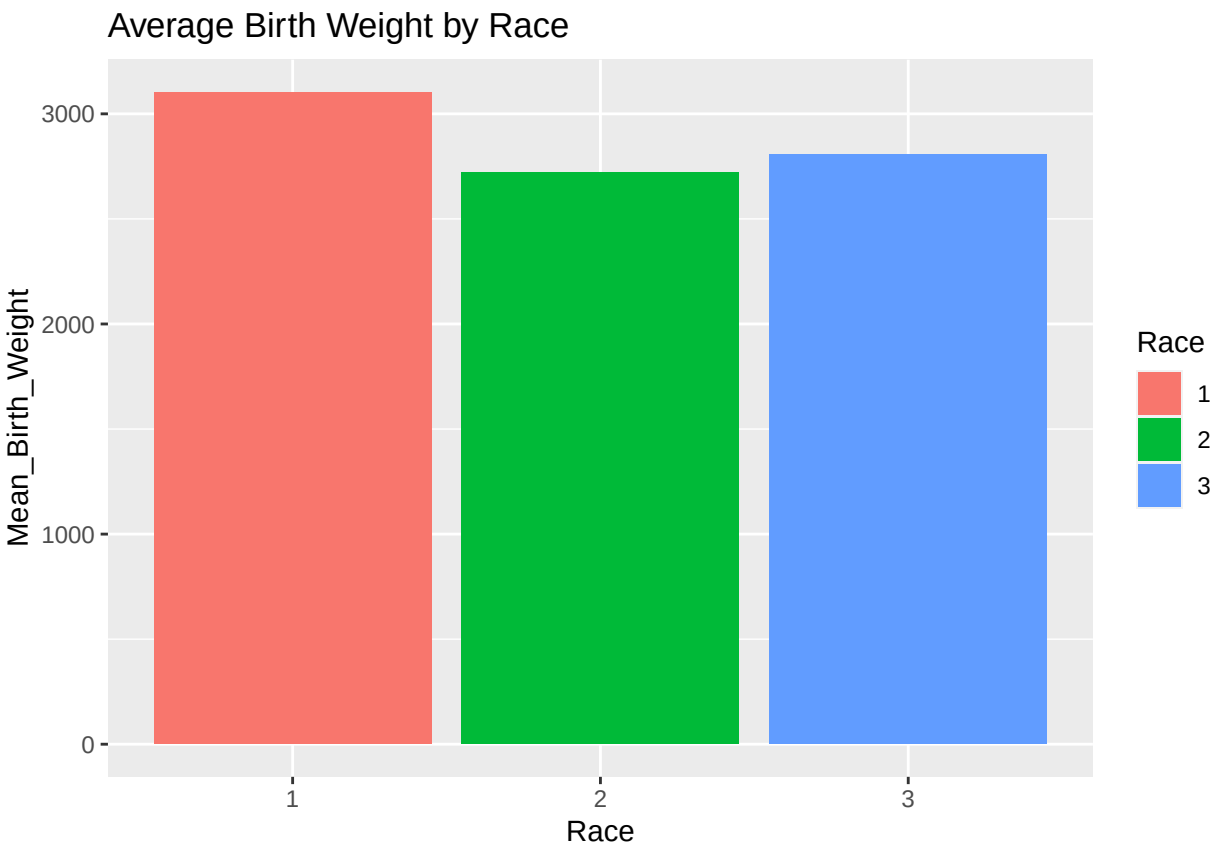
When comparing the output here with the output of the tapply() function above, we see that they're quite consistent. Babies of race 1 (White) have more weight compared to the babies of other races.

Answer 3f:

To create a ggplot of the above, first we loaded the package, followed by creating a vector called "Mean_Birth_Weight_By_Race, which holds the value of the output of the ddply() function used above. Post that, we use the geom_bar function to get a bar chart in place as shown below.

```
library(ggplot2)
Mean_Birth_Weight_By_Race <- ddply(birthwt, ~Race, summarise,
Mean_Birth_Weight = mean(bwt))

ggplot(data = Mean_Birth_Weight_By_Race, aes(x = Race, y = Mean_Birth_Weight,
fill = Race)) + geom_bar(stat = "identity")+
ggtitle("Average Birth Weight by Race")
```



Answer 3g:

In order to show the proportion of babies, we used a formula to calculate the sum of all values of "low" (so the length of the rows equals to 1) divided by the sum of count of low.

```
library(plyr)
library(knitr)
Average_By_smoking_habit <- ddply(birthwt, "Smoke", summarise,
Mean_Birth_Weight = mean(bwt), Low_Weight_Proportion = sum(low)/sum(count(low)))
kable(Average_By_smoking_habit)
```

Smoke	Mean_Birth_Weight	Low_Weight_Proportion
0	3055.696	0.25
1	2771.919	0.40

Answer 3h:

To add "Mother Smokes" to the ddply() formula used above in part (g), we factored the smoke column and labled the values 0 and 1 as "Mother Does Not Smoke" and "Mother Smokes" respectively. For this, we used the factor() function as follows:

```
library(knitr)
birthwt$smoke <- factor(birthwt$smoke,
labels = c("Mother Does Not Smoke", "Mother Smokes"))
Average_by_smoking_habit <- ddply(birthwt, "smoke",
summarise, Mean_Birth_Weight = mean(bwt),
Low_Weight_Proportion = sum(low)/sum(count(low)))
kable(Average_by_smoking_habit)
```

smoke	Mean_Birth_Weight	Low_Weight_Proportion
Mother Does Not Smoke	3055.696	0.25
Mother Smokes	2771.919	0.40

Answer 3i:

In order to understand if the mother's age is correlated with birth weight, we took the median of age and adding it to the ddply() formula used in above questions. This was done using the following commands:

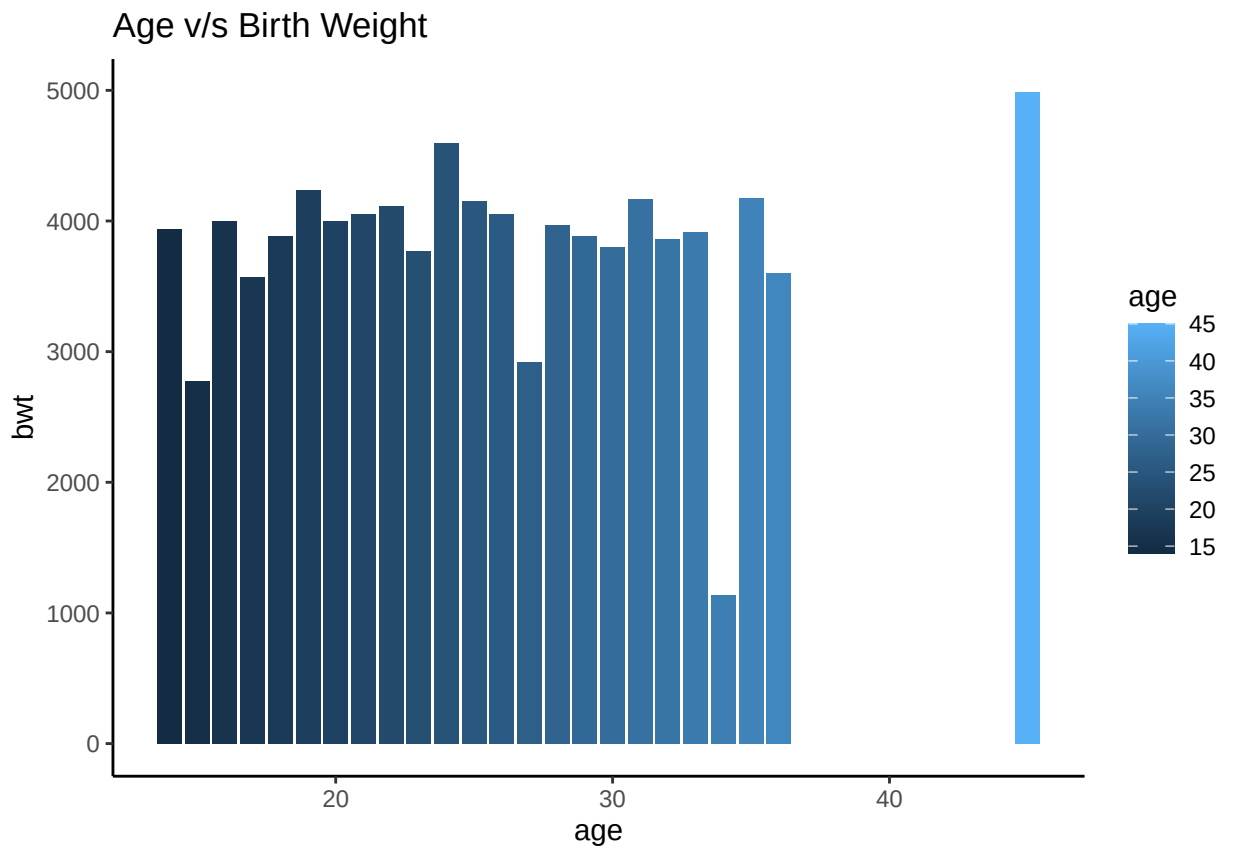
```
library(plyr)
library(knitr)
Average_by_smoking_habit <- ddply(birthwt, "smoke", summarise,
Mean_Birth_Weight = mean(bwt), Low_Weight_Proportion = sum(low)/sum(count(low)),
Median_Age = median(age))
kable(Average_by_smoking_habit)
```

smoke	Mean_Birth_Weight	Low_Weight_Proportion	Median_Age
Mother Does Not Smoke	3055.696	0.25	23
Mother Smokes	2771.919	0.40	22

As seen in the above table, the median age is 23 for mothers who don't smoke and 22 for those who smoke. The birth weight of the child does not change too much with the mother's age, so there seems to be no correlation between the mother's age and the birth weight per se.

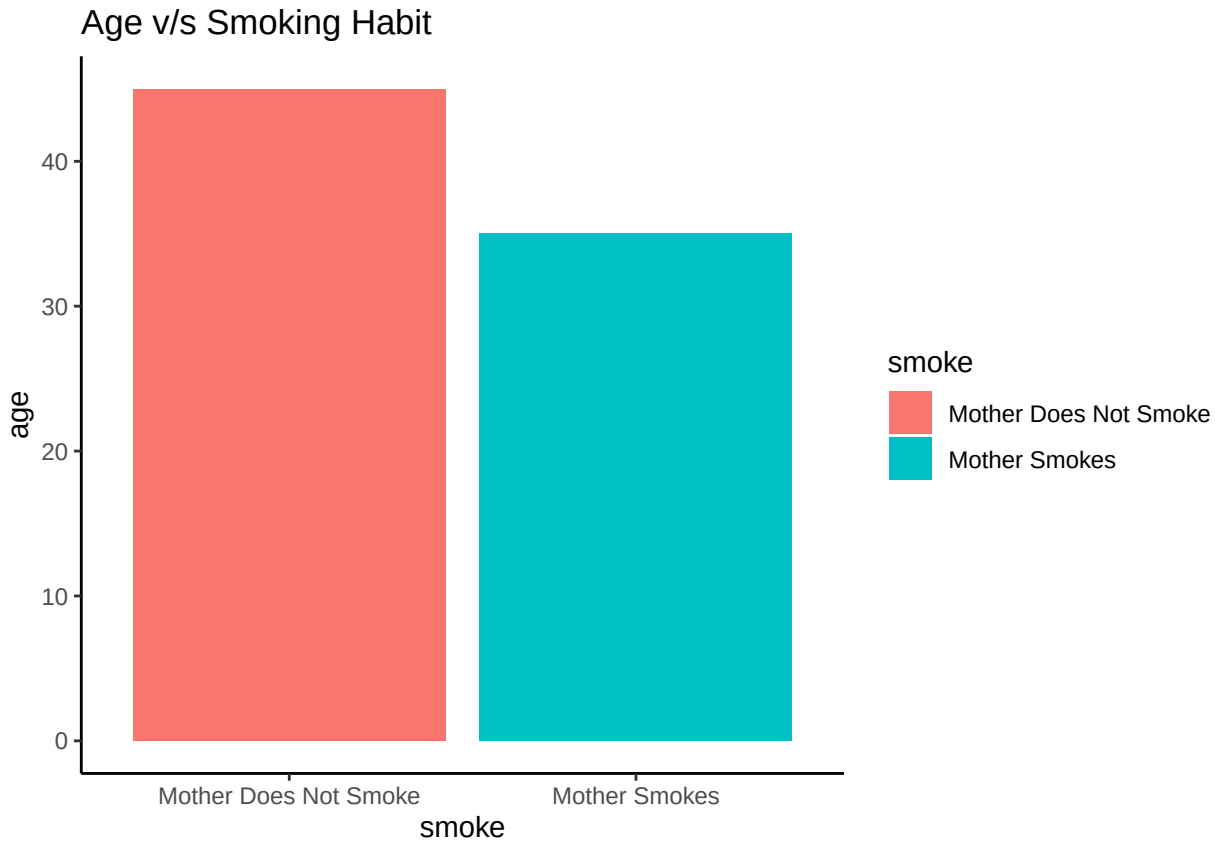
We can also check this by creating a bar graph of Age v/s Birth Weight, where we can see that the birth weight of the child doesn't change drastically as the age differs. There's one outlier in the data for age 45, but that does not change the overall hypothesis.

```
library(ggplot2)
ggplot(birthwt, aes(x=age, y=bwt, fill=age)) +
geom_bar(stat = "identity", position = position_dodge()) +
ggtitle("Age v/s Birth Weight") + theme_classic()
```



Similarly, to see the correlation between mother's age and smoking habit, we can create a similar barplot like we did for the problem above:

```
ggplot(birthwt, aes(x=smoke, y=age, fill=smoke)) +
geom_bar(stat = "identity", position = position_dodge()) +
ggtitle("Age v/s Smoking Habit") + theme_classic()
```



Per the above graph, we can see that there are four entries of mothers with ages 35, 36, 36, and 45 that do not smoke, but majority of the mothers till from the ages 14 to 35 either smoke or don't smoke. Hence, this data does not conclusively prove a solid correlation between the mother's age and smoking habits either.

Problem 4

```
library("datasets")
library("ggplot2")
library("tidyverse")
library("pacman")
library("rio")
```

Answer 4a:

The function `typeof(diamonds$price)` gives you the type of the variable price. The output gives us an integer value. Therefore, the type of the variable price is integer.

To find out if the distribution is symmetric, right skewed or left skewed we can look at the mean and median. If the mean and median are very close, then the distribution is symmetric. If mean > median, then the distribution is right-skewed and if the mean < median then the distribution is right-skewed.

```
typeof(diamonds$price)
```

```
## [1] "integer"
```

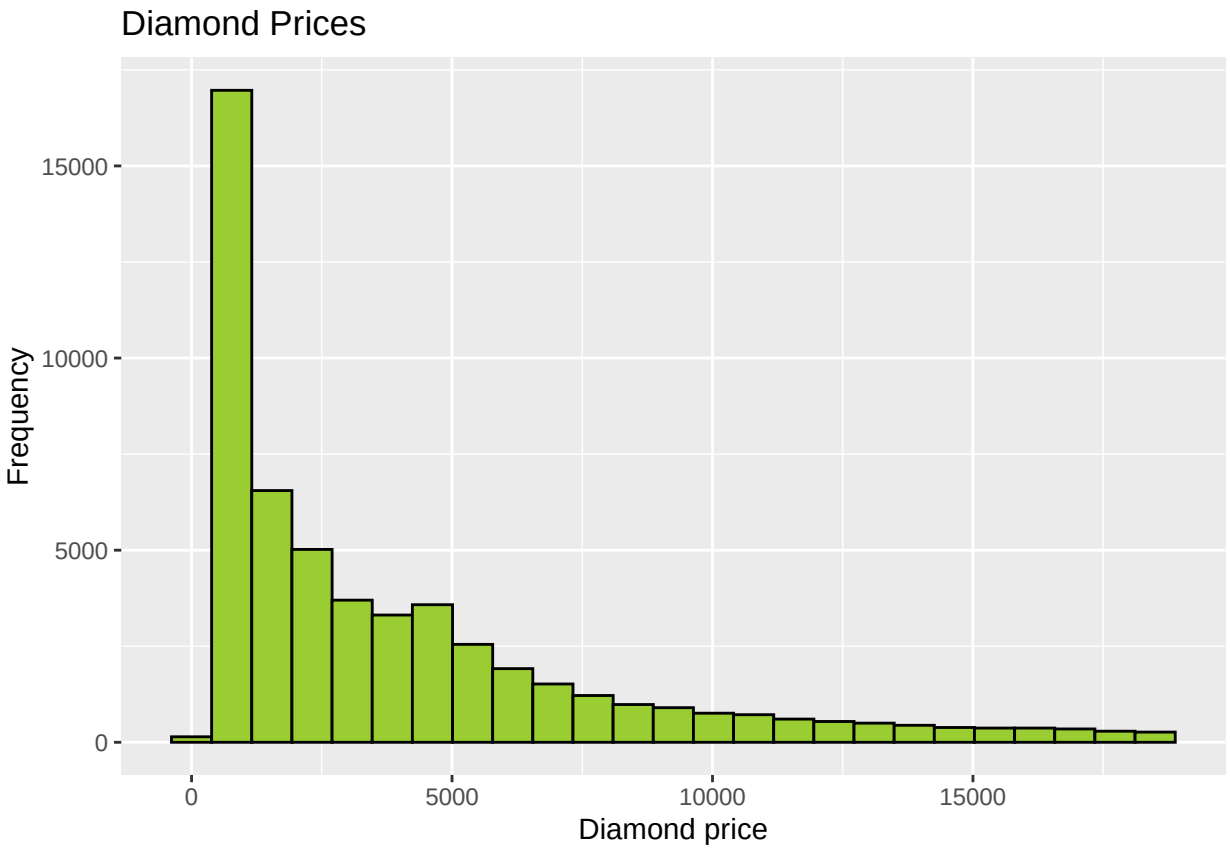
```
mean(diamonds$price)
```

```
## [1] 3932.8
```

```
median(diamonds$price)
```

```
## [1] 2401
```

```
df <- data.frame(rbind(diamonds))
ggplot(df,aes(x = price)) +
  geom_histogram(col = "black", fill = "yellowgreen", bins = 25) +
  labs(title = "Diamond Prices",
       x = "Diamond price",
       y = "Frequency")
```



So, while observing those values for price in the data set diamonds, the mean is 3932.8 [we can find that out by using `mean ()` & `median()` commands.] and the median is 2401. Since the mean > median, therefore the distribution is *right skewed*. **Yes, the shape of the distribution matches our expectations.**

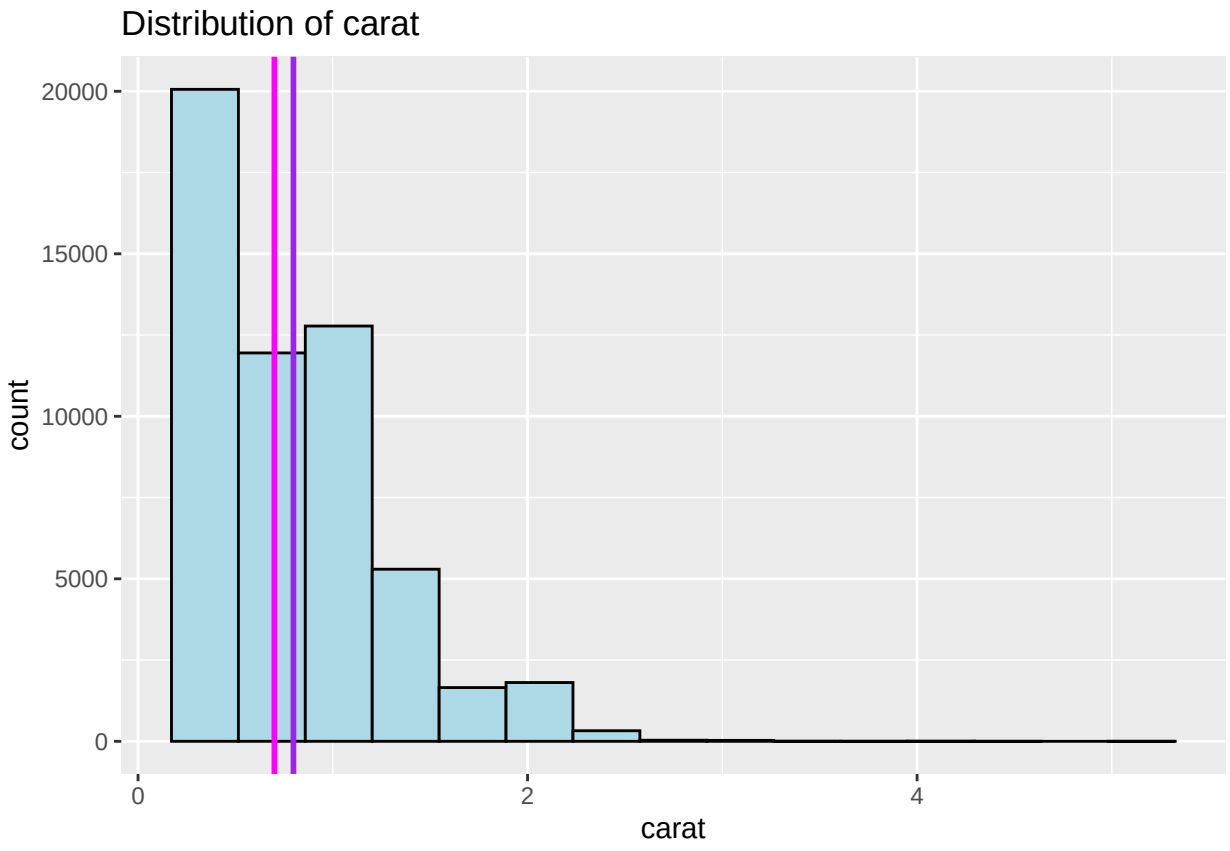
Answer 4b:

By looking at the data set `diamonds` we can find out which another variable is numerical. To verify that we also used the command `typeof(diamonds$carat)` which gives us double as the output making it sure that the variable is *numerical*.

```
typeof(diamonds$carat)
```

```
## [1] "double"
```

```
diamonds %>%
  group_by(carat) %>%
  ggplot() +
  geom_histogram(col = "black", fill = "lightblue",
    mapping = aes(x=carat),
    bins = 15
    ) +
  labs(title = "Distribution of carat")+
  geom_vline(aes(xintercept = mean(diamonds$carat)),col='purple',size=1) +
  geom_vline(aes(xintercept = median(diamonds$carat)),col='magenta',size=1)
```




```
summary(diamonds$carat)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2000  0.4000  0.7000  0.7979  1.0400  5.0100
```

To visualize the data better we plotted a histogram using the command `geom_histogram()`. In the following code we have incorporated piping. We have added color to our histogram making the border black and filled it with light-blue. Since we want to plot the distribution for carat therefore, we have mapped the x-axis with that particular variable. By adding the command `bins=15` we can control the number of vertical lines in a histogram. We have added a title to our chart. Since we were asked to also give information about the mean and median so we tried plotting the two on the graph by using `geom_vline`. This gives us a vertical line and we can change the color, width, etc. of the same inside the function.

We can also find the summary of the particular variable by using the command `summary(diamonds$carat)`. This give us information about the min, Q1, median, mean, Q3 and max value of the data.

Answer 4c:

```
typeof(diamonds$color)
```

```
## [1] "integer"
```

```
class(diamonds$color)
```

```
## [1] "ordered" "factor"
```

```
df <- data.frame(rbind(diamonds))
mode <- function(x) {
  a <- unique(x)
  b <- tabulate(match(x, a))
  a[b == max(b)]
}

data <- mode(diamonds$color)
as.character(data)
```

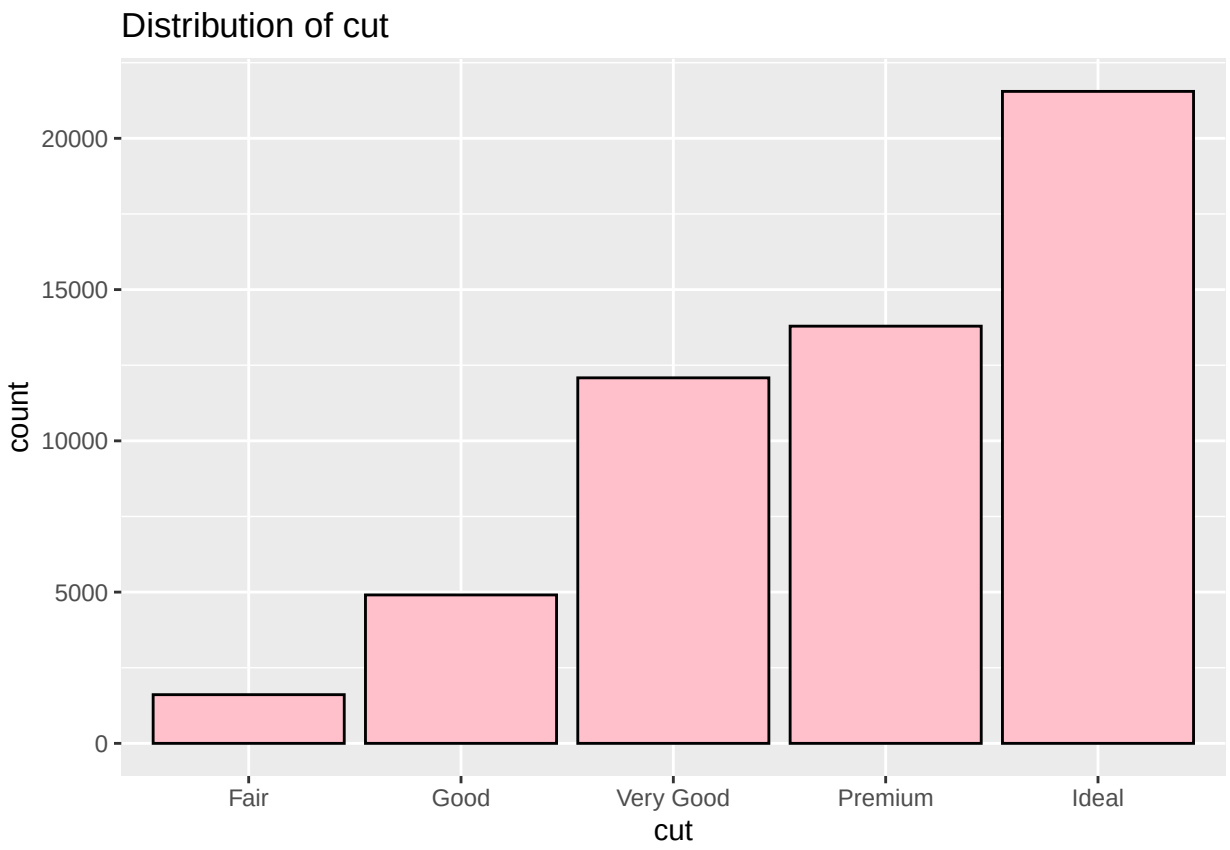
```
## [1] "G"
```

To find out the type of variable price is we use `class(diamonds$color)` command. Since using the command `typeof(diamonds$color)` gives an integer output and when we look at the data set we can see that color is not an integer. Therefore, we use `class()` to find out the type of the variable. Color is an *ordered factor*.

To find out which color is most prominently represented in the data set we can use the mathematical mode. Since there is no command for that in R, therefore, we made a function that can perform a similar task. We are converting the data set diamonds into a data frame. The lines that follow that are forming a function mode. In that we're finding the unique values, then we're finding out the frequency which will tell us how many times a particular value will appear in the data frame. After that we will tabulate it and find out the maximum value among them all. This will give us the mathematical mode which tells us the most prominent color in the dataset. The most prominent color in the data set is **G**.

Answer 4d:

```
ggplot(df,aes(x = cut)) +
  geom_bar(col = "black", fill = "pink") +
  labs(title = "Distribution of cut")
```




```
summary(diamonds$cut)
```

```
##      Fair      Good Very Good   Premium     Ideal
##      1610      4906     12082     13791     21551
```

To plot the distribution we have used the command `geom_bar()`. The distribution of the data is left-skewed. Cut has a positive correlation with count. With increase in cut from fair to ideal the count also increases. We have also added the summary to the code to give us a better idea of the trend of the distribution.

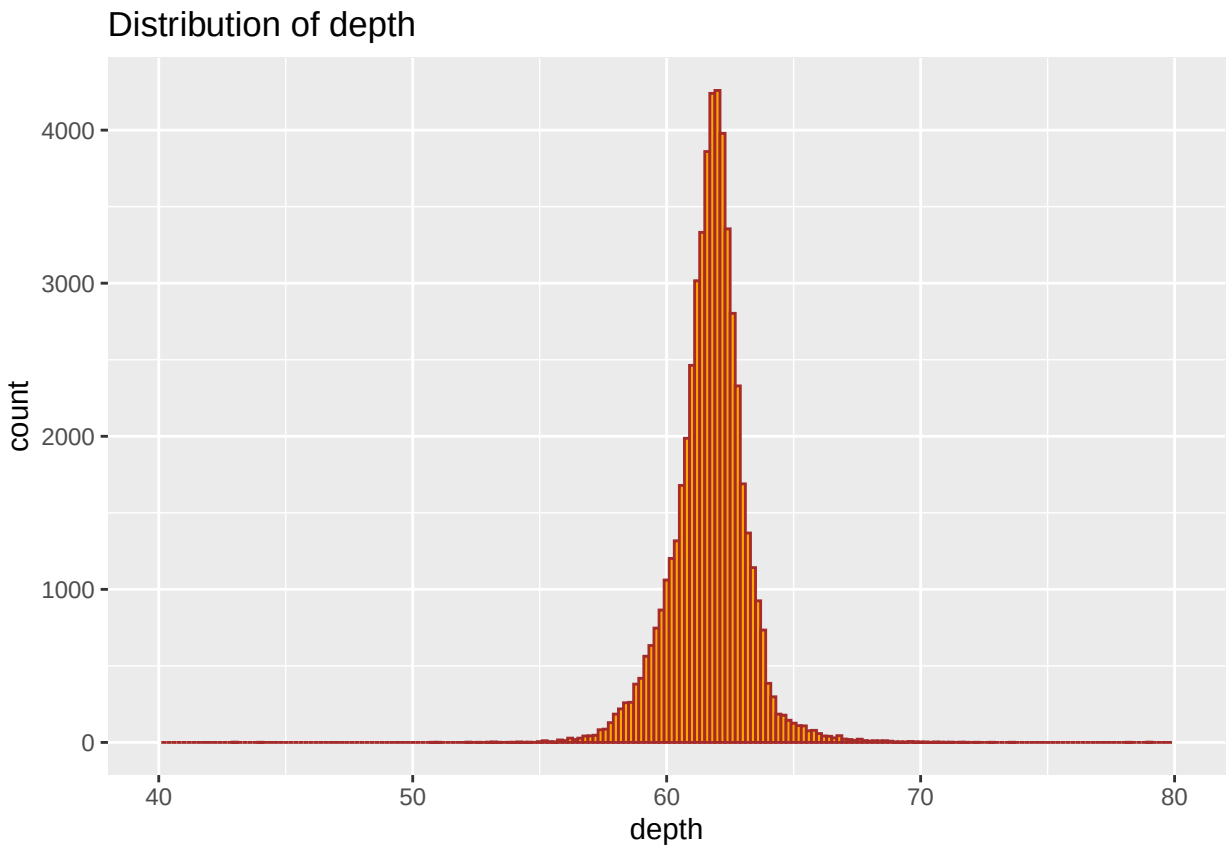
Answer 4e:

To make a histogram we use the command `geom_histogram()`. To add more definition to the graph we find out it's summary. The summary let's us know the max and min value of depth which helps us choose our x co-ordinate limit. The distribution of depth graph gives us a histogram for the depth of diamonds. The `binwidth` command helps us control the width of each vertical bar. The summary also let's us know about distribution. The `xlim` command helps us limit the x axis of the graph.

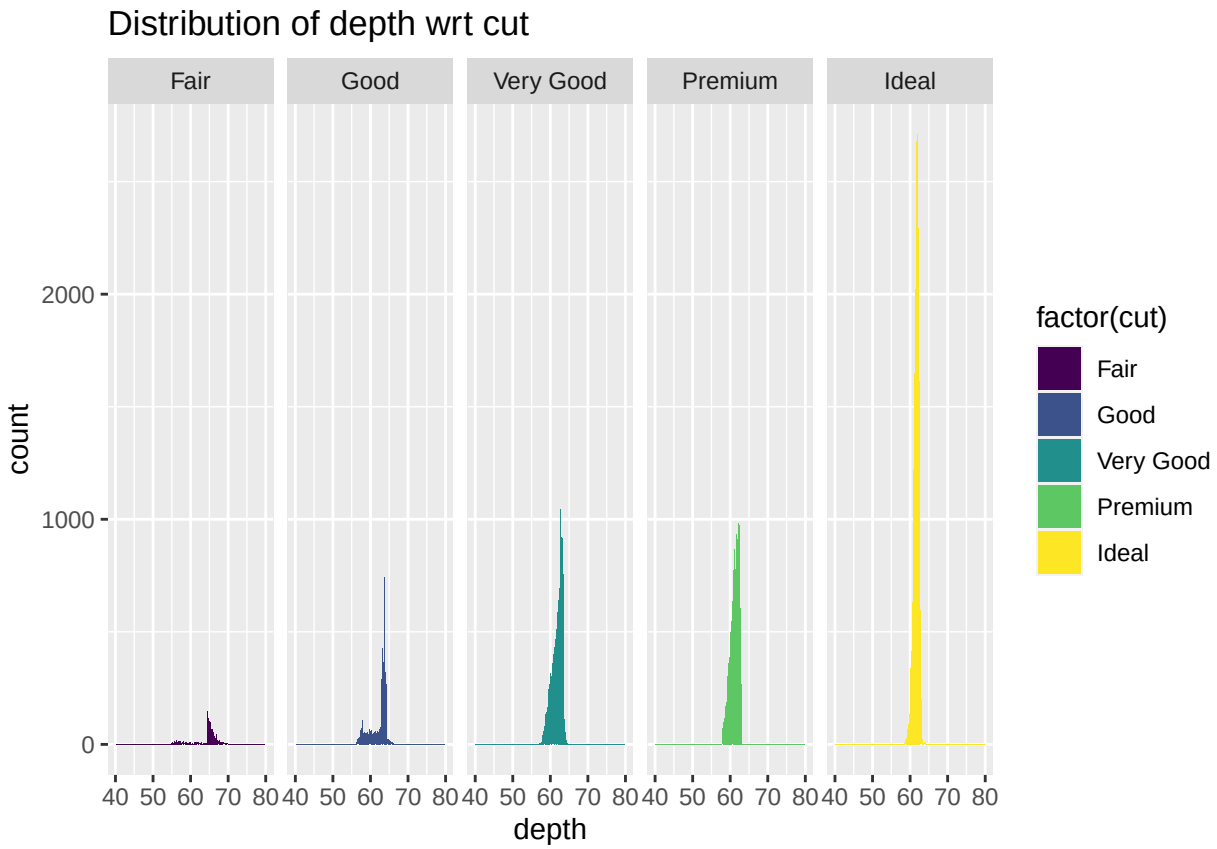
```
summary(diamonds$depth)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   43.00   61.00   61.80   61.75   62.50   79.00
```

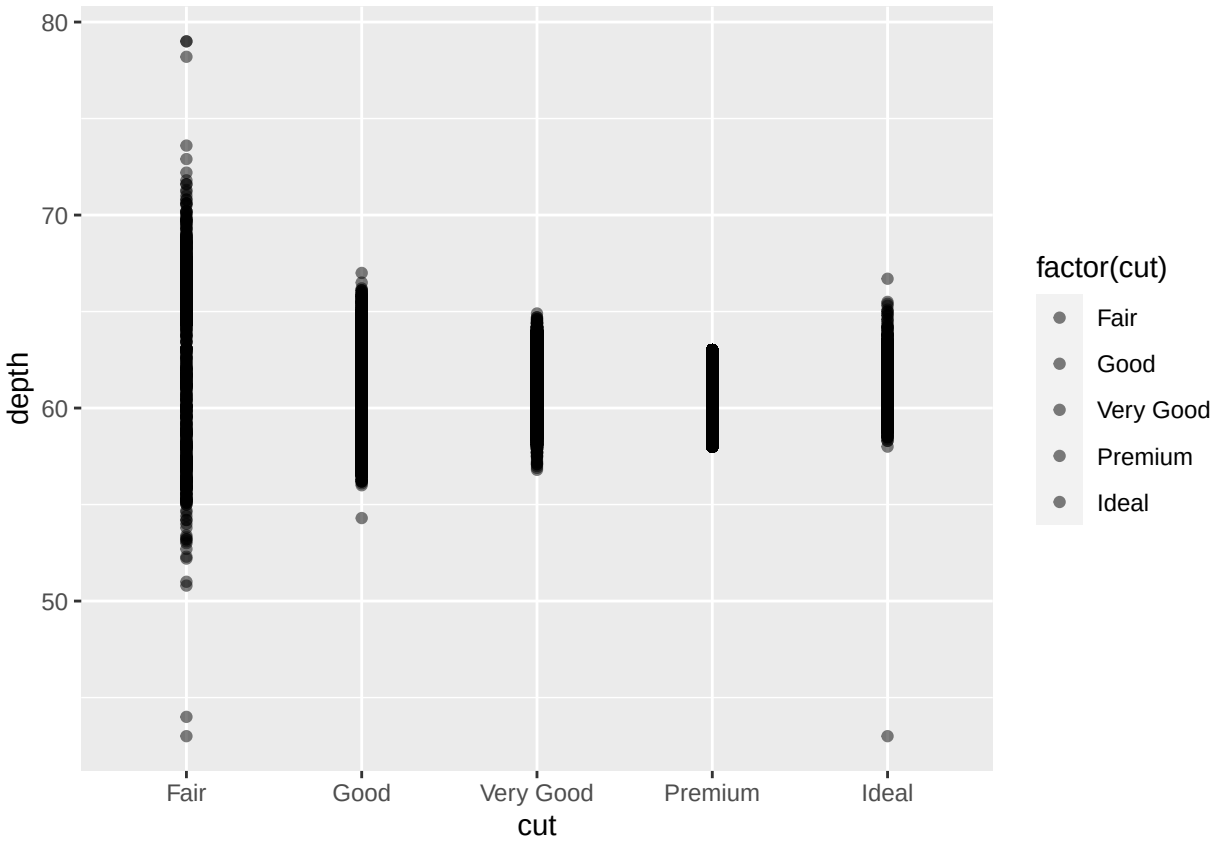
```
ggplot(df,aes(x = depth)) +
  geom_histogram(
    binwidth = 0.2,
    col = "brown", fill = "orange") +
    xlim(40, 80) +
    labs(title = "Distribution of depth")
```




```
ggplot(diamonds,aes(x=depth, group=cut, fill=factor(cut))) +
  geom_histogram(position="dodge", binwidth =0.2 ) +
  xlim(40,80) +
  labs(title = "Distribution of depth wrt cut") +
  facet_grid(~ cut)
```




```
ggplot(diamonds,aes(x=cut, y=depth, group=depth, fill=factor(cut))) +
  geom_point(alpha =1/2)
```

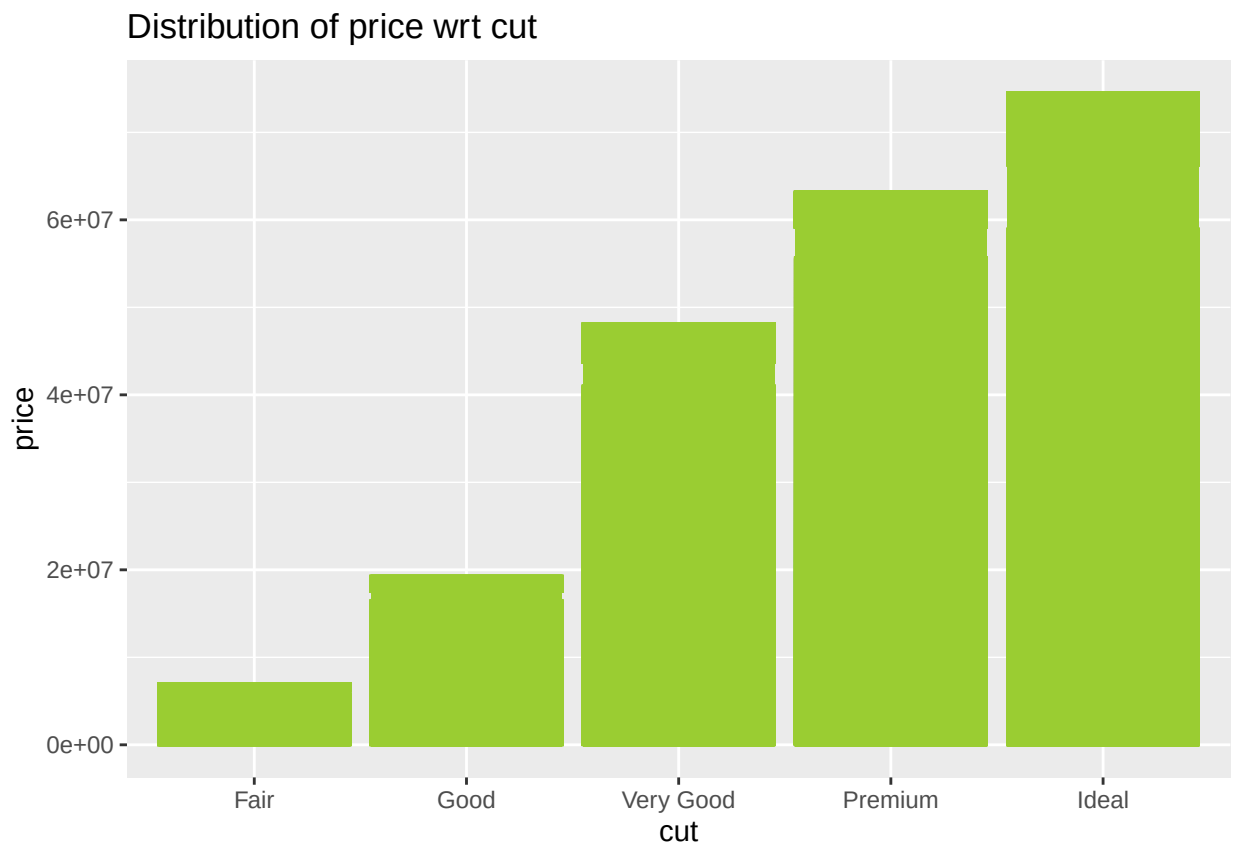


Since, the mean is approximately equal to median therefore, the distribution is **symmetric**.

To add cut to the visualization we will either have to use aesthetic or facet. We have used aesthetic for mapping and facet to make grids for each cut in the dataset. To determine the increase or decrease in cut as depth increases or decreases, we plotted a graph for depth and cut using aesthetic. It can be visualized from the plot that the average trend is *decreasing and then it increases for ideal cut*,which means that as the cut range increases the depth of the diamond decreases except for ideal cut diamonds.

Answer 4f:

```
diamonds %>%
  group_by(price) %>%
  ggplot() +
  labs(title = "Distribution of price wrt cut") +
  geom_col(mapping = aes(y=price, x=cut),color = "yellowgreen")
```

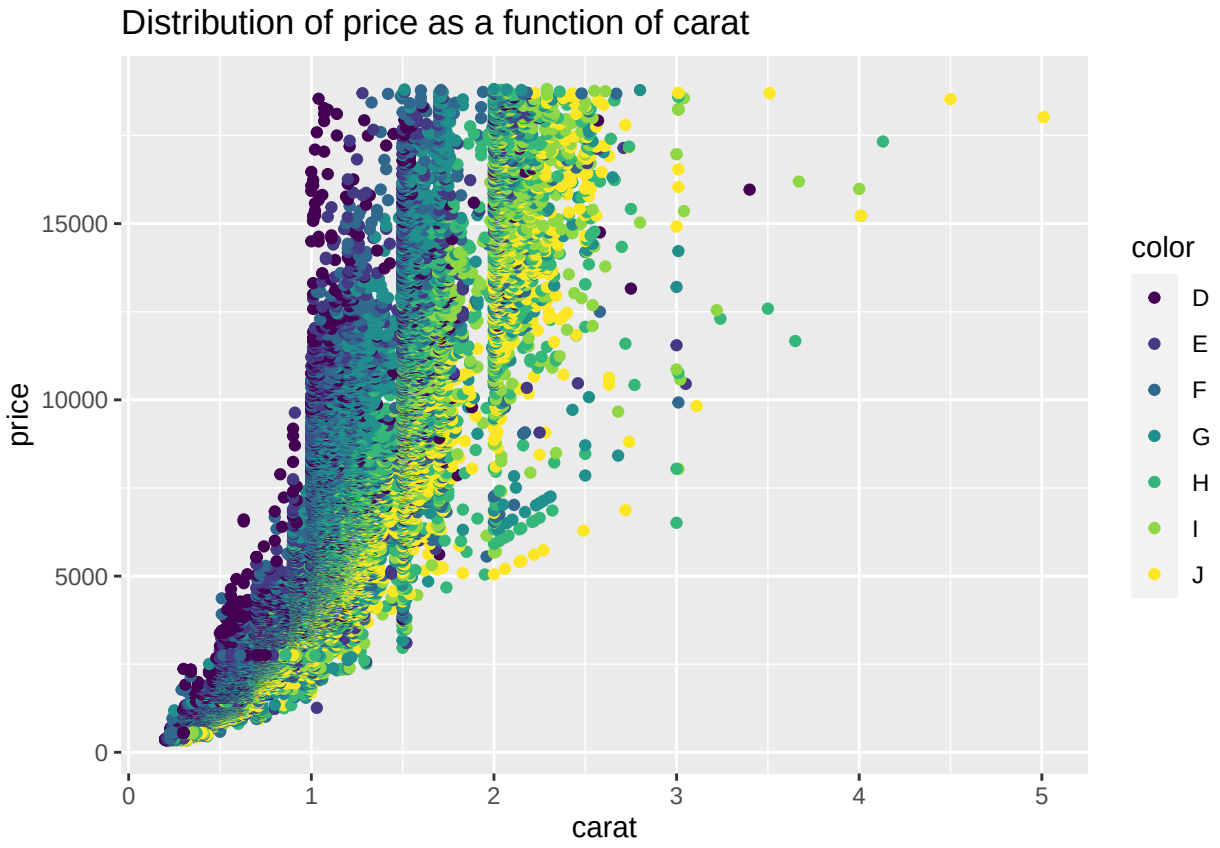


To plot the distribution of price for different cuts we have used piping along with `geom_col`. This command is used when you what to plot a histogram but your x and y-axis both have variables. The graph is left-skewed but there is nothing unusual in it other than the y-axis values. The values displayed on the y-axis are much greater than the actual values. The max and minimum values of price are 18823 and 326 respectively. Nothing can be stated without a detailed analysis of the dataset.

Answer 4g:

To draw the scatterplot showing price as a function of carat we will use the command `geom_point()`. The different colors in the scatterplot tell us about the different carat diamonds and their prices.

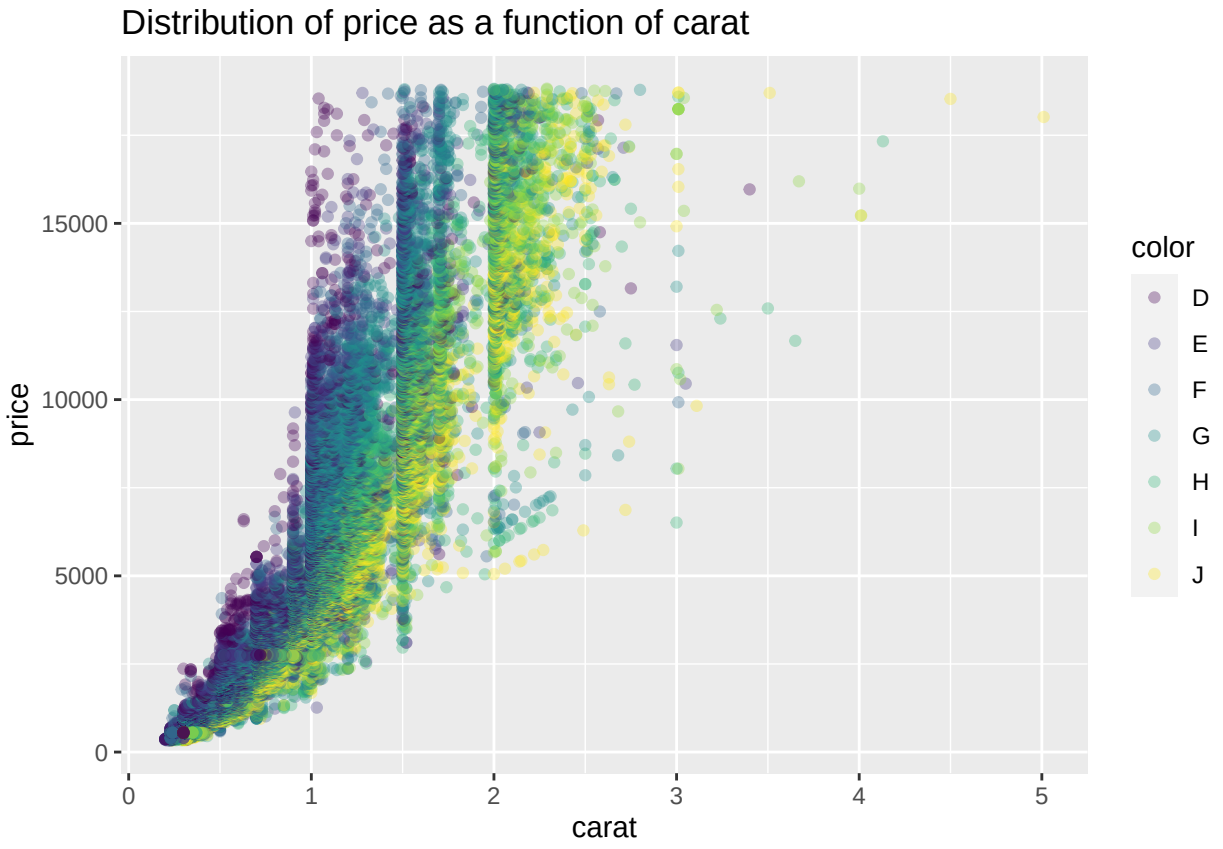
```
ggplot(diamonds) +
  labs(title = "Distribution of price as a function of carat") +
  geom_point(mapping = aes(x=carat , y=price, colour = color))
```



Answer 4h:

This part of the question is similar to the g part except one minor change that is the alpha argument used. The alpha command is used for the *opacity* of the points. As you keep decreasing the number the opacity decreases.

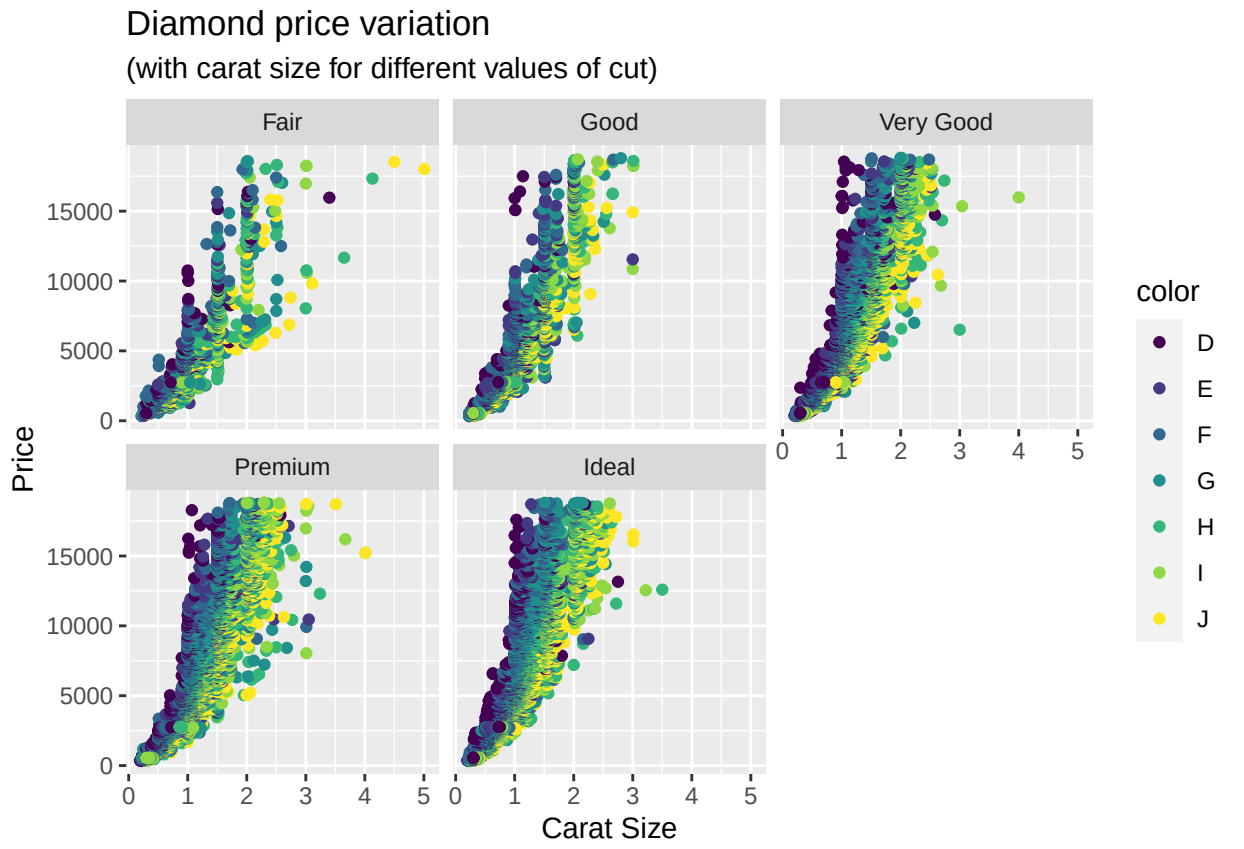
```
ggplot(diamonds) +
  labs(title = "Distribution of price as a function of carat") +
  geom_point(mapping = aes(x=carat , y=price , colour = color),
             alpha=1/3)
```



Answer 4i:

To show the diamond price variation with carat size for different values of cut we have used `facet_wrap` in our code. We will be using the 2 variables price and carat as the x and y axis variable by using aesthetic. We then facet_wrap it with cut.

```
ggplot(data = diamonds, aes(carat, price, colour=color)) +
  geom_point() +
  labs(title = "Diamond price variation",
       subtitle = "(with carat size for different values of cut)",
       y = "Price", x = "Carat Size") +
  facet_wrap(~ cut)
```



Problem 5

Answer 5a:

Below are 3 functions mean_func() to find mean, median_func() to find median and input_func() to get user input. mean_func() and meadina_func() have 2 parameters the vector for which we want to find the mean and median and the length of that vector.

*The mean_func() takes the sum of all values in vector and then divides it by the length of that vector. it returns the mean of the vector.

*The median_func() first sorts the vector in ascending order using the inbuilt sort() function. It then check whether the length of vector is even or odd and according to that, the median is calculated using the If Else loop structure.

*The input_func() takes the user input of numbers.

```
> mean_func<- function(x,len){
+   sum_x = sum(x)
+   mean = sum_x/len
+   return(as.double(mean))
+ }
>
> median_func<-function(x,len){
+   x<-sort(x,decreasing=FALSE)
+   if(len%%2==0){
+     median = (x[len/2]+x[(len/2)+1])/2
+   } else{
+     median = x[(len+1)/2]
+   }
+   return(as.double(median))
+ }
>
> input_func<-function(){
+   input<-as.integer(readline())
+   return(input)
+ }
>
>
> vector_y<-c(1,1,1,2,3)
> len_y = length(vector_y)
> mean_func(vector_y,len_y)
```

```
## [1] 1.6
```

```
> median_func(vector_y, len_y)
```

```
## [1] 1
```

Below is the code for the dynamic user input and output. But somehow, R markdown does not allow to show user inputs in the final R markdown file. But the code runs perfectly in normal R script.

```
vector_x<-input_func()
```

```
vector_x
```

```
## [1] NA
```

```
len_x = length(vector_x)
mean = mean_func(vector_x,len_x)
print(mean)
```

```
## [1] NA
```

```
median = median_func(vector_x, len_x)
print(median)
```

```
## [1] NA
```

Answer 5b:

Below is the function select10() that takes in an integer n value above 2 and outputs a final list comprising of n lists which has n random numbers from 1 to 10.

The select10() first checks whether the number passed is greater than 2 or not. It then runs a while loop while n>0 to generate the lists of n random numbers and append it to our final list V. This final list V is then returned at the end.

```
select10<-function(n){
  v<-list()
  m<-n
  if(n>2){
    while(n>0){
      random_list<-list()
      random_list<-list(sample(1:10,m))
      v<-c(v,random_list)
      n<-n-1
    }
  } else{
    print("Please enter a number greater than 2")
  }
  return(v)
}

print("Please enter a number greater than 2")
```

```
## [1] "Please enter a number greater than 2"
```

```
a = as.integer(readline())
```

```
final_list<-select10(a)
```

```
## Error in if (n > 2) {: missing value where TRUE/FALSE needed
```

```
print(final_list)
```

```
## Error in print(final_list): object 'final_list' not found
```

```
final_list<-select10(5)
final_list
```

```
## [[1]]
## [1] 5 3 7 1 9
##
## [[2]]
## [1]  9  3  8  5 10
##
## [[3]]
## [1] 6 1 5 7 2
##
```

```
## [[4]]
## [1] 9 7 6 2 3
##
## [[5]]
## [1]  6  1  7 10  5
```

Answer 5c:

Below is the function for Z score Transformation. Z Score is the score of a variable where it shows how many standard deviations is the data point away from the mean. Positive Z Score tells that the data point is on the right side of mean and Negative Z Score tells that it is on left side of mean.

```
zscore<-function(x){
  m<-mean(x)
  sd<-sd(x)

  x_zscore<-(x-m)/sd
  return(x_zscore)
}

z_out<-zscore(c(55,63,42,78,69))
z_out
```

```
## [1] -0.4663960  0.1165990 -1.4137629  1.2097146  0.5538452
```