EXECUTIVE SUMMARY REPORT 2

ALY 6000 INTRODUCTION TO ANALYTICS

RAHUL AVINASH JADHAV

Northeastern University

College of Professional Studies, Northeastern University, Boston, MA 02115

Contact: jadhav.ra@northeastern.edu

Submitted to Professor: Prof. Dr Mary Donhoffner

Date of submission: 10/03/2021

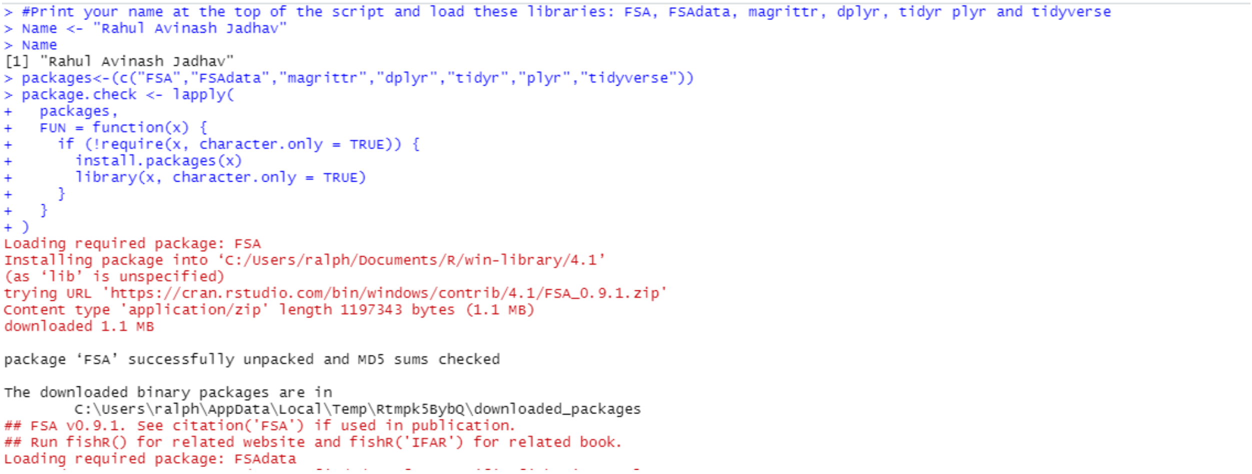
Introduction

In this Executive Summary report, we are going to learn about: the installation and loading of multiple libraries, import CSV files, functions (like count(), axis(),mtext(),cumsum(), line()), create a subset from the main set, different visualization techniques, and customizing the visualization for better understanding of the data and making the visualization more interpretable based on categories to understand more information.

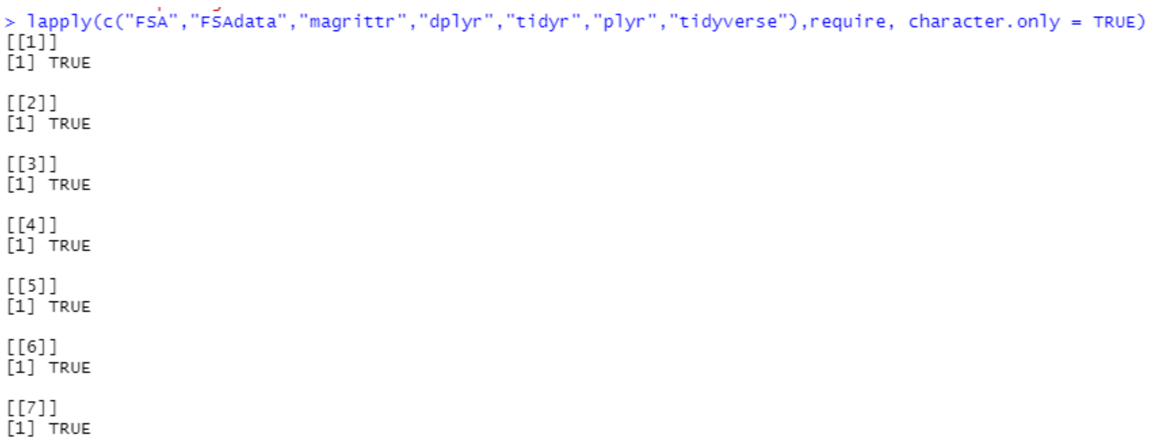
Key findings

1. Print your name at the top of the script and load these libraries: FSA, FSAdata, magrittr, dplyr, tidyr plyr and tidyverse

   Console screenshot:



```
> #Print your name at the top of the script and load these libraries: FSA, FSAdata, magrittr, dplyr, tidyr plyr and tidyverse
> Name <- "Rahul Avinash Jadhav"
> Name
[1] "Rahul Avinash Jadhav"
> packages<-(c("FSA","FSAdata","magrittr","dplyr","tidyr","plyr","tidyverse"))
> package.check <- lapply(
+   packages,
+   FUN = function(x) {
+     if (!require(x, character.only = TRUE)) {
+       install.packages(x)
+       library(x, character.only = TRUE)
+     }
+   }
+ )
Loading required package: FSA
Installing package into ‘C:/Users/ralph/Documents/R/win-library/4.1’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.1/FSA_0.9.1.zip'
Content type 'application/zip' length 1197343 bytes (1.1 MB)
downloaded 1.1 MB

package ‘FSA’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\ralph\AppData\Local\Temp\Rtmpk5BybQ\downloaded_packages
## FSA v0.9.1. See citation('FSA') if used in publication.
## Run fishR() for related website and fishR('IFAR') for related book.
Loading required package: FSAdata
```

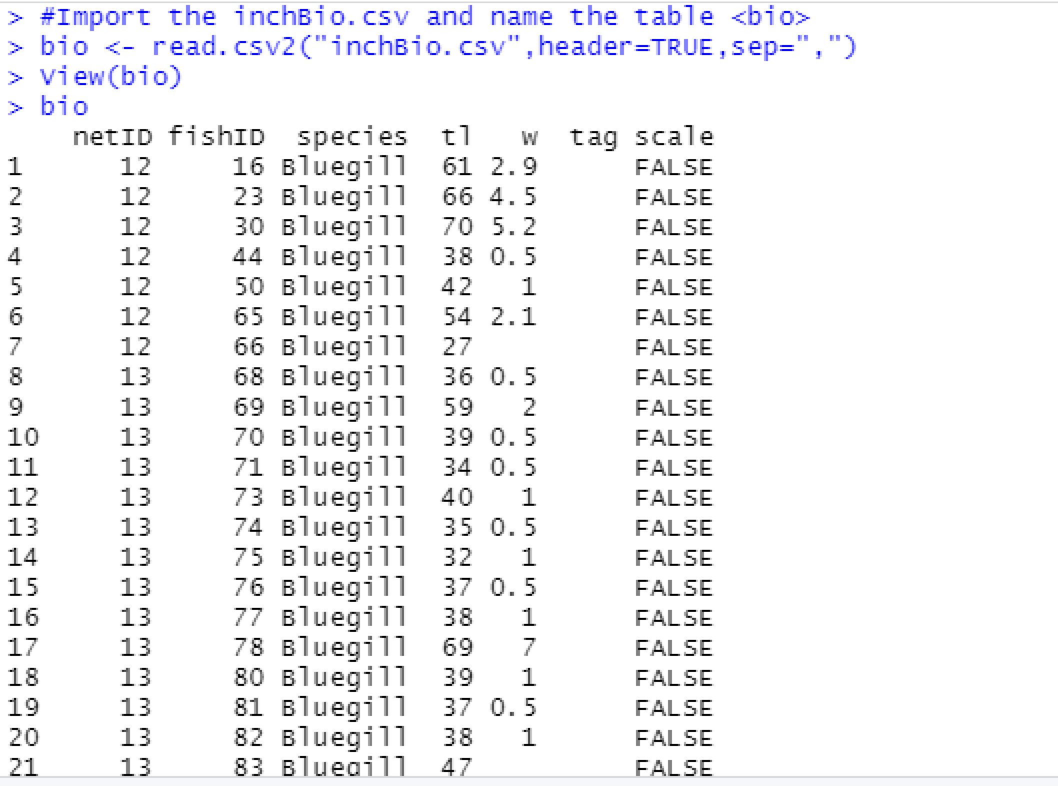


```
> lapply(c("FSA","FSAdata","magrittr","dplyr","tidyr","plyr","tidyverse"),require, character.only = TRUE)
[[1]]
[1] TRUE

[[2]]
[1] TRUE

[[3]]
[1] TRUE

[[4]]
[1] TRUE

[[5]]
[1] TRUE

[[6]]
[1] TRUE

[[7]]
[1] TRUE
```

Here we have printed my surname on the top of the script. We have also used a function to check if the package was installed or not. If not, then that package will get installed and will be imported. The package.check() is the function we created that will check and install the package. To re-check about package status, we have used the lapply(x, fun) function where x is vector and fun is the function to be applied to each element of x. We have used require() function in lapply() function. The require() function returns FALSE if the package is not installed and TRUE if the package is loaded.

2. Import the inchBio.csv and name the table <bio>

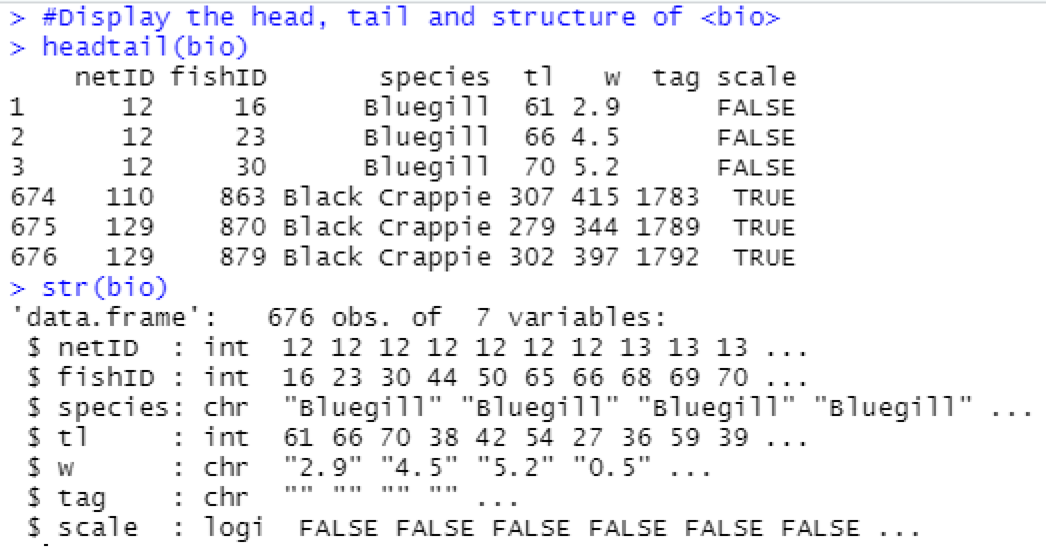
Console screenshot:



```
> #Import the inchBio.csv and name the table <bio>
> bio <- read.csv2("inchBio.csv",header=TRUE,sep=",")
> View(bio)
> bio
   netID fishID  species tl   w tag scale
1     12     16 Bluegill 61 2.9     FALSE
2     12     23 Bluegill 66 4.5     FALSE
3     12     30 Bluegill 70 5.2     FALSE
4     12     44 Bluegill 38 0.5     FALSE
5     12     50 Bluegill 42   1     FALSE
6     12     65 Bluegill 54 2.1     FALSE
7     12     66 Bluegill 27         FALSE
8     13     68 Bluegill 36 0.5     FALSE
9     13     69 Bluegill 59   2     FALSE
10    13     70 Bluegill 39 0.5     FALSE
11    13     71 Bluegill 34 0.5     FALSE
12    13     73 Bluegill 40   1     FALSE
13    13     74 Bluegill 35 0.5     FALSE
14    13     75 Bluegill 32   1     FALSE
15    13     76 Bluegill 37 0.5     FALSE
16    13     77 Bluegill 38   1     FALSE
17    13     78 Bluegill 69   7     FALSE
18    13     80 Bluegill 39   1     FALSE
19    13     81 Bluegill 37 0.5     FALSE
20    13     82 Bluegill 38   1     FALSE
21    13     83 Bluegill 47         FALSE
```

In this task we have imported the csv file provided by the instructor into table called bio and then printed the table.

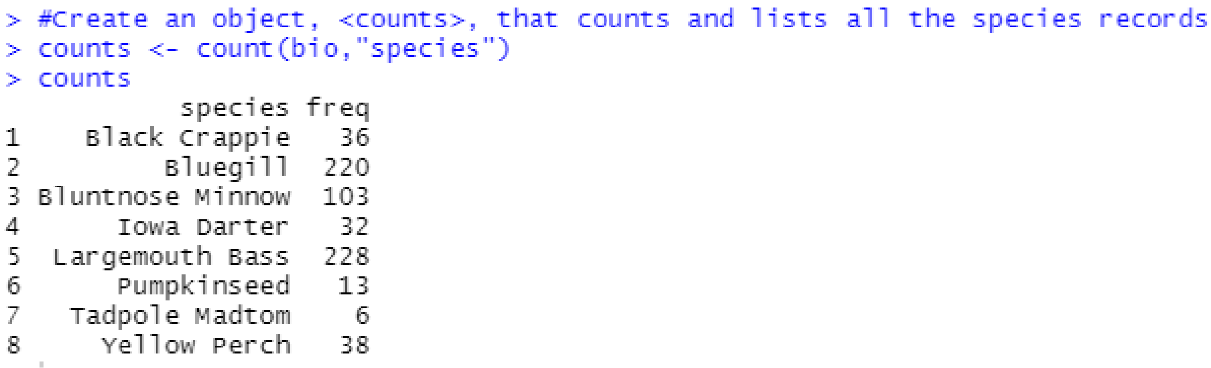
3. Display the head, tail and structure of <bio>



```
> #Display the head, tail and structure of <bio>
> headtail(bio)
    netID fishID       species  tl   w  tag scale
1      12     16      Bluegill  61 2.9      FALSE
2      12     23      Bluegill  66 4.5      FALSE
3      12     30      Bluegill  70 5.2      FALSE
674   110    863 Black Crappie 307 415 1783  TRUE
675   129    870 Black Crappie 279 344 1789  TRUE
676   129    879 Black Crappie 302 397 1792  TRUE
> str(bio)
'data.frame':   676 obs. of  7 variables:
 $ netID  : int  12 12 12 12 12 12 12 13 13 13 ...
 $ fishID : int  16 23 30 44 50 65 66 68 69 70 ...
 $ species: chr  "Bluegill" "Bluegill" "Bluegill" "Bluegill" ...
 $ tl     : int  61 66 70 38 42 54 27 36 59 39 ...
 $ w      : chr  "2.9" "4.5" "5.2" "0.5" ...
 $ tag    : chr  "" "" "" "" ...
 $ scale  : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

In this task, we have displayed the head, tail, and structure of the table bio. We have used the headtail() function to display the head and tail that returns the 'n' first and last rows of the table(by default n is 3), and for the structure, we have used the str() function that returns the internal structure of the object.

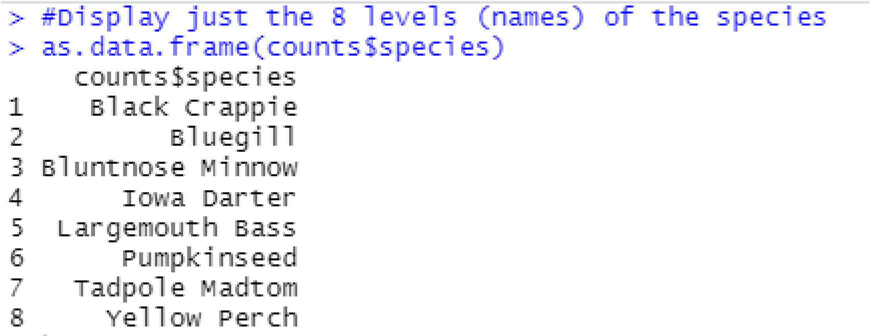
4. Create an object, <counts>, that counts and lists all the species records



```
> #Create an object, <counts>, that counts and lists all the species records
> counts <- count(bio,"species")
> counts
           species freq
1    Black Crappie   36
2         Bluegill  220
3 Bluntnose Minnow  103
4      Iowa Darter   32
5  Largemouth Bass  228
6      Pumpkinseed   13
7   Tadpole Madtom    6
8     Yellow Perch   38
```

In this task, we will create object counts that will count and list all the species records. We have used the count() function that counts the unique values of one or more variables.

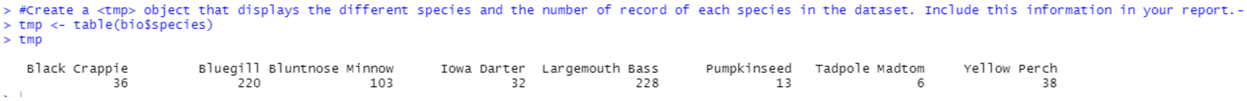
5. Display just the 8 levels (names) of the species



```
> #Display just the 8 levels (names) of the species
> as.data.frame(counts$species)
    counts$species
1     Black Crappie
2          Bluegill
3 Bluntnose Minnow
4       Iowa Darter
5   Largemouth Bass
6       Pumpkinseed
7     Tadpole Madtom
8      Yellow Perch
```

In this task, we will display the names of the species. We have used the as.data.frame() function to convert that table to dataframe and to display the data understandably.

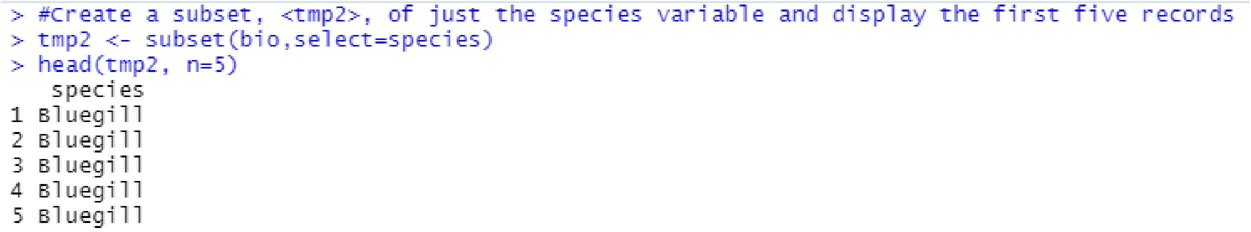
6. Create a <tmp> object that displays the different species and the number of record of each species in the dataset. Include this information in your report.-



```
> #Create a <tmp> object that displays the different species and the number of record of each species in the dataset. Include this information in your report.-
> tmp <- table(bio$species)
> tmp

   Black Crappie         Bluegill Bluntnose Minnow      Iowa Darter  Largemouth Bass      Pumpkinseed   Tadpole Madtom     Yellow Perch 
              36              220              103               32              228               13                6               38 
```

In this task, we have created a tmp object which stores the name and frequency of each species

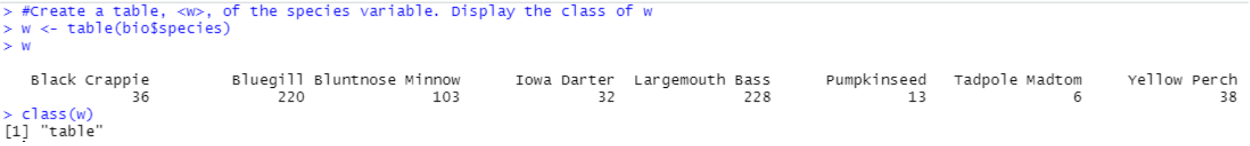
7. Create a subset, <tmp2>, of just the species variable and display the first five records



```
> #Create a subset, <tmp2>, of just the species variable and display the first five records
> tmp2 <- subset(bio,select=species)
> head(tmp2, n=5)
   species
1 Bluegill
2 Bluegill
3 Bluegill
4 Bluegill
5 Bluegill
```

In this task, we have created a tmp2 subset that is a subset of the bio dataset and displayed the first five records of the tmp2 object.

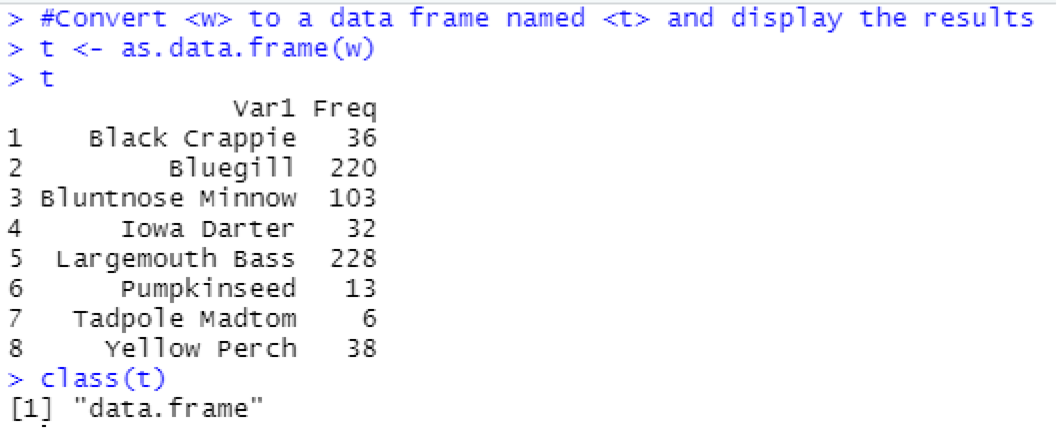
8. Create a table, <w>, of the species variable. Display the class of w



```
> #Create a table, <w>, of the species variable. Display the class of w
> w <- table(bio$species)
> w

   Black Crappie         Bluegill Bluntnose Minnow      Iowa Darter  Largemouth Bass      Pumpkinseed   Tadpole Madtom     Yellow Perch 
              36              220              103               32              228               13                6               38 
> class(w)
[1] "table"
```

In this task, we have created a table w of the species variable using the table() function and have provided the species variable from the bio dataset. We have also displayed the class of the newly created table using the class() function that returns the class of the object.

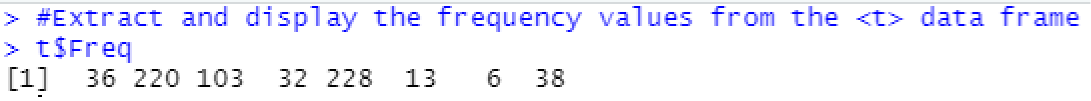
9. Convert <w> to a data frame named <t> and display the results



```
> #Convert <w> to a data frame named <t> and display the results
> t <- as.data.frame(w)
> t
              Var1 Freq
1    Black Crappie   36
2         Bluegill  220
3 Bluntnose Minnow  103
4      Iowa Darter   32
5  Largemouth Bass  228
6      Pumpkinseed   13
7   Tadpole Madtom    6
8     Yellow Perch   38
> class(t)
[1] "data.frame"
```

In this task, we have converted a table to a data frame and displayed the results. For this task, we have used as.data.frame() function that converts the table into a data frame. We have also printed the class of the data frame t.

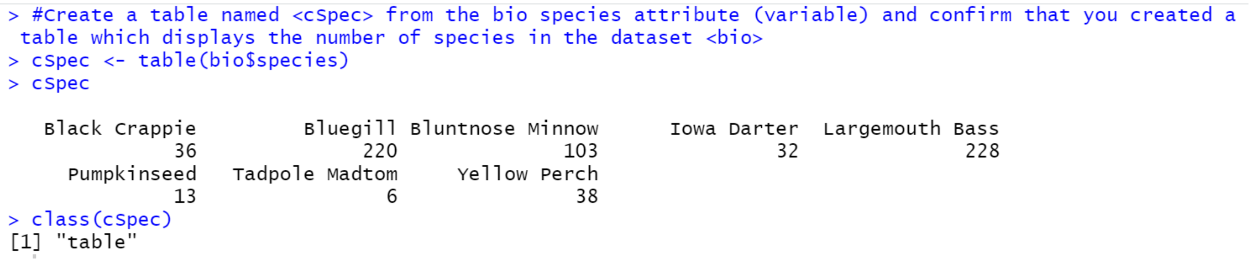
10. Extract and display the frequency values from the <t> data frame



```
> #Extract and display the frequency values from the <t> data frame
> t$Freq
[1]  36 220 103  32 228  13   6  38
```

In this task, we have extracted and displayed the frequency values from the t dataframe.

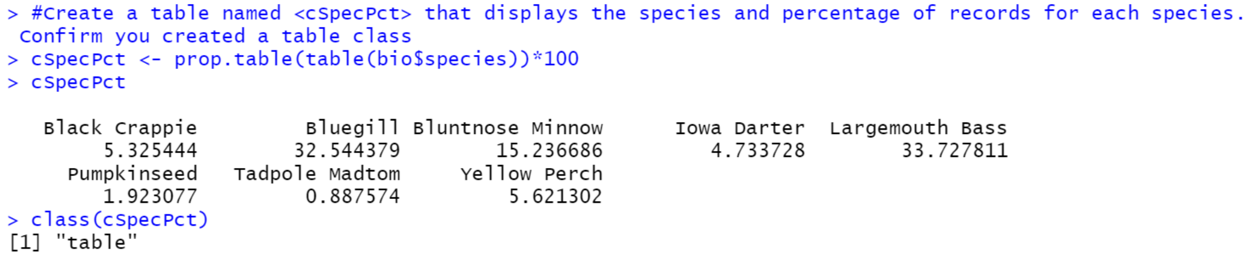
11. Create a table named <cSpec> from the bio species attribute (variable) and confirm that you created a table which displays the number of species in the dataset <bio>.



```
> #Create a table named <cSpec> from the bio species attribute (variable) and confirm that you created a
 table which displays the number of species in the dataset <bio>
> cSpec <- table(bio$species)
> cSpec

   Black Crappie         Bluegill Bluntnose Minnow      Iowa Darter  Largemouth Bass 
              36              220              103               32              228 
     Pumpkinseed   Tadpole Madtom     Yellow Perch 
              13                6               38 
> class(cSpec)
[1] "table"
```

In this task, we have created a table cSpec from the bio species attribute and have confirmed that the created object is a table using the class() function and have displayed the table. The class() function displays the class of the object. The table() function creates a categorical representation of data in the form of a table.

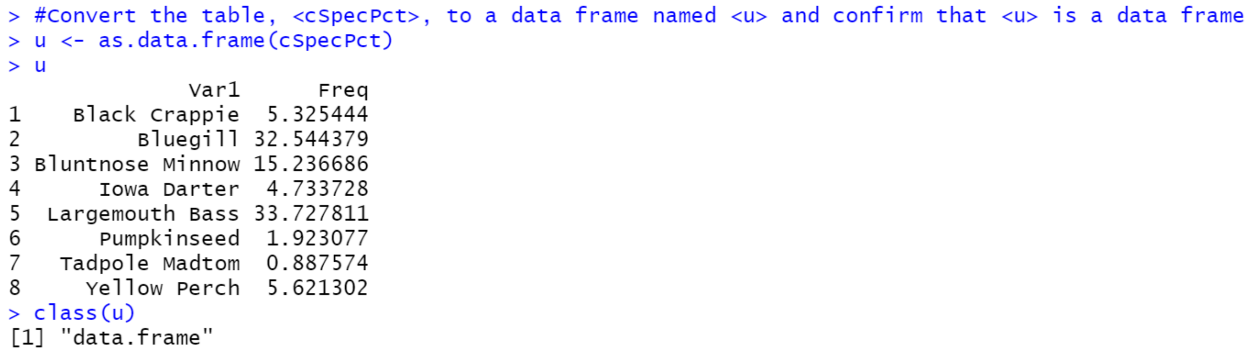
12. Create a table named <cSpecPct> that displays the species and percentage of records for each species. Confirm you created a table class.



```
> #Create a table named <cSpecPct> that displays the species and percentage of records for each species.
 Confirm you created a table class
> cSpecPct <- prop.table(table(bio$species))*100
> cSpecPct

   Black Crappie         Bluegill Bluntnose Minnow      Iowa Darter  Largemouth Bass 
        5.325444        32.544379        15.236686         4.733728        33.727811 
     Pumpkinseed   Tadpole Madtom     Yellow Perch 
        1.923077         0.887574         5.621302 
> class(cSpecPct)
[1] "table"
```

In this task, we have created a table named cSpecPct which contains the information of species and percentage of each species. We have also confirmed that the created object is a table using the class() function and displayed it along with the records.

13. Convert the table, <cSpecPct>, to a data frame named <u> and confirm that <u> is a data frame



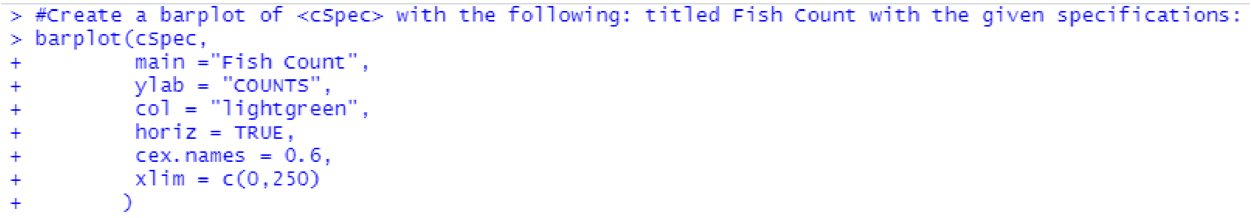
```
> #Convert the table, <cSpecPct>, to a data frame named <u> and confirm that <u> is a data frame
> u <- as.data.frame(cSpecPct)
> u
             Var1      Freq
1   Black Crappie  5.325444
2        Bluegill 32.544379
3 Bluntnose Minnow 15.236686
4     Iowa Darter  4.733728
5 Largemouth Bass 33.727811
6     Pumpkinseed  1.923077
7   Tadpole Madtom  0.887574
8    Yellow Perch  5.621302
> class(u)
[1] "data.frame"
```

In this task, we have created a dataframe called u from a table called cSpecPct. We have displayed the dataframe and confirmed that the object created is a dataframe using the class() function.

14. . Create a barplot of <cSpec> with the following: titled Fish Count with the following specifications:

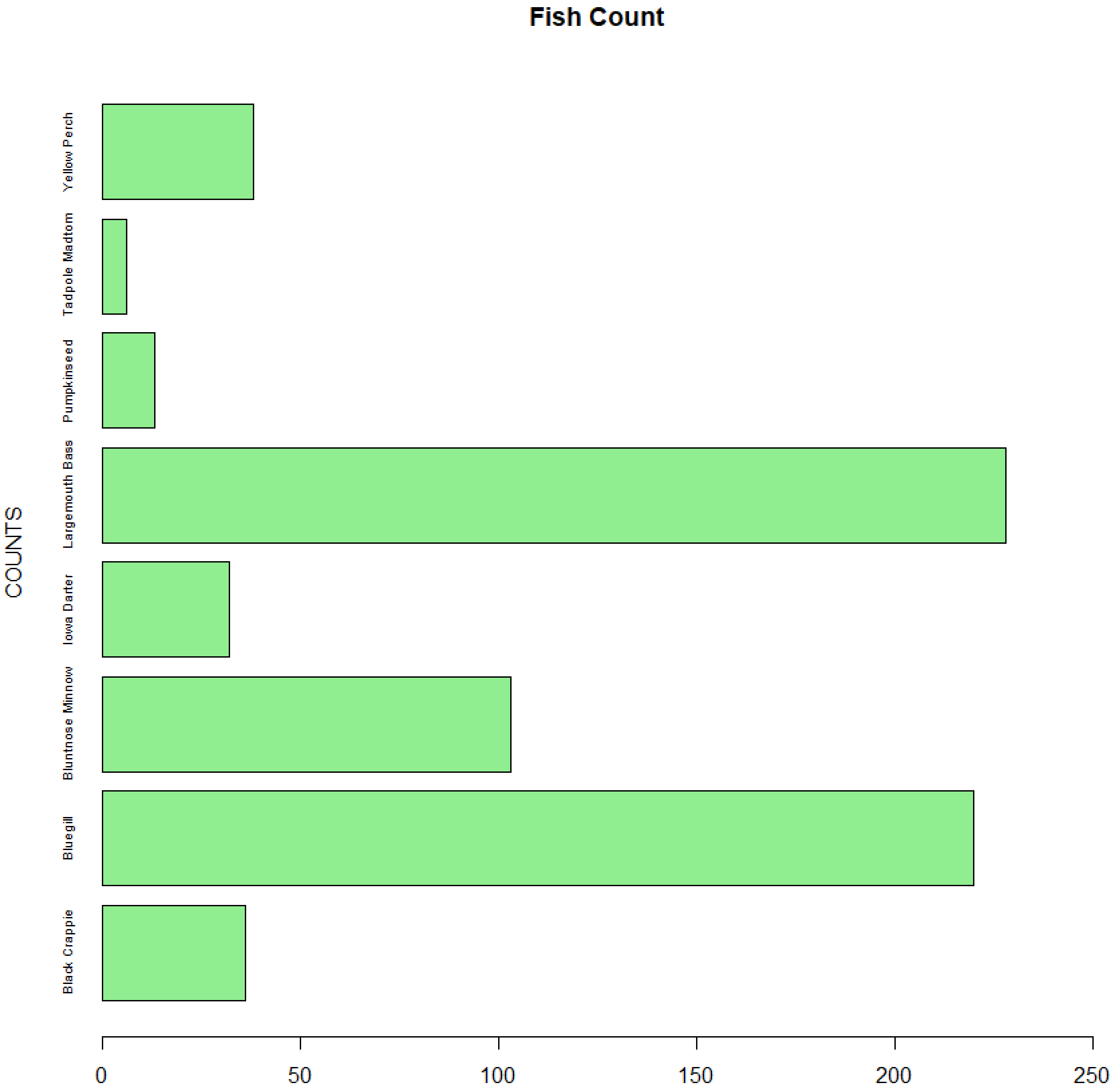
- Title: Fish Count
- Y axis is labeled "COUNTS"
- Color the bars Light Green
- Rotate Y axis to be horizontal
- Set the X axis font magnification to 60% of nominal

Console screenshot :



```
> #Create a barplot of <cSpec> with the following: titled Fish Count with the given specifications:
> barplot(cSpec,
+         main ="Fish Count",
+         ylab = "COUNTS",
+         col = "lightgreen",
+         horiz = TRUE,
+         cex.names = 0.6,
+         xlim = c(0,250)
+        )
```

Barplot of cSpec :



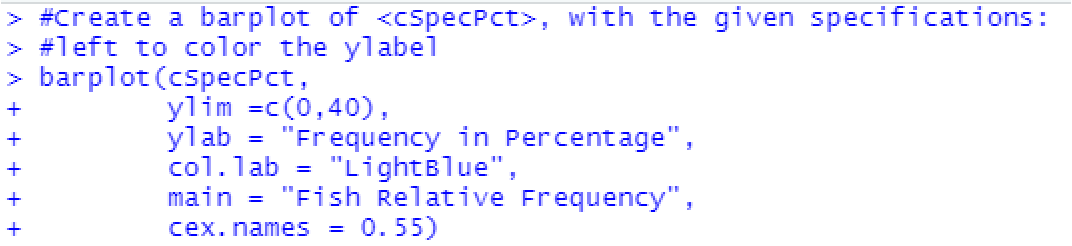
In this task, we have created a barplot of the cspec and used arguments to change: the xlim, the title, y label, the color of bars, etc as instructed by the instructor.

From the graph, we can say that the count of tadpole madtom fish is the least, and the largemouth bass fish is the highest.

15. Create a barplot of <cSpecPct>, with the following specifications:

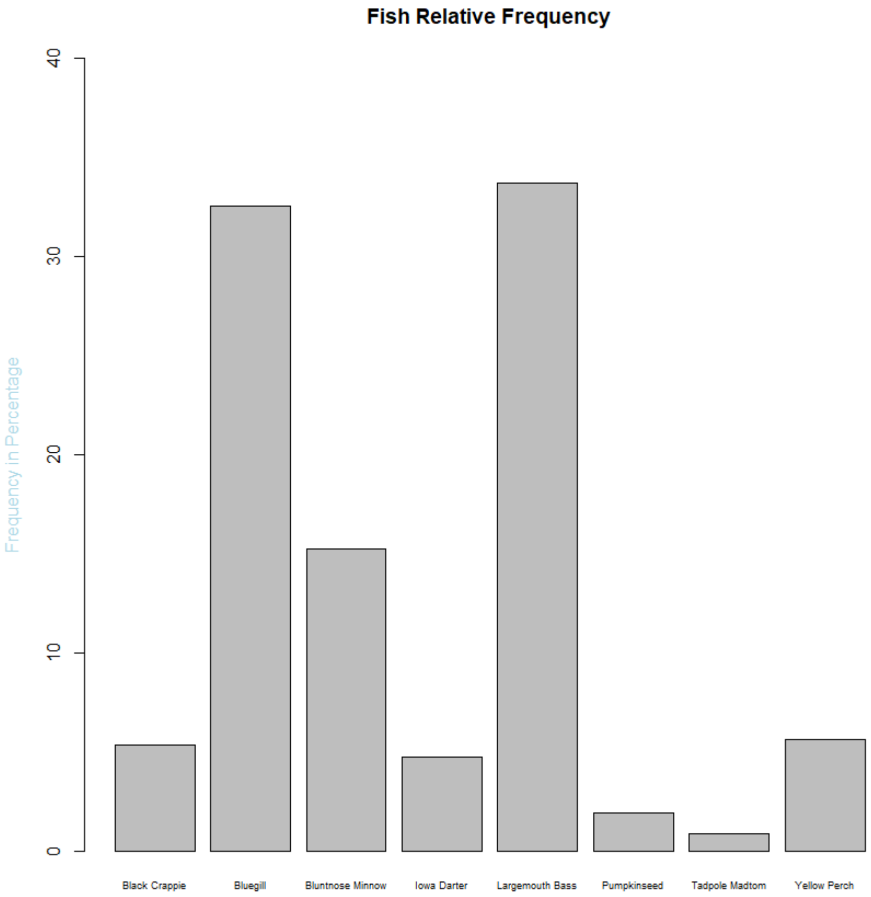
- Y axis limits of 0 to 4
- Y axis label color of Light Blue
- Title of “Fish Relative Frequency”

Console Screenshot :



```
> #Create a barplot of <cSpecPct>, with the given specifications:
> #left to color the ylabel
> barplot(cSpecPct,
+         ylim =c(0,40),
+         ylab = "Frequency in Percentage",
+         col.lab = "LightBlue",
+         main = "Fish Relative Frequency",
+         cex.names = 0.55)
```

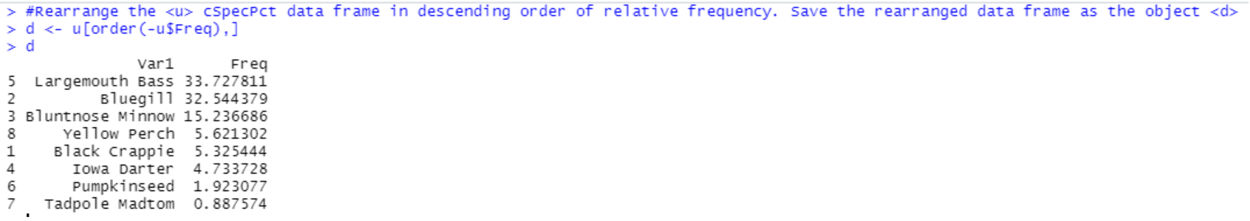
Plotting Screenshot :



In this task, we have created a barplot of cSpecPct and used arguments to change the: y lim, y label color, and title as instructed by the instructor.

From the plotting, we can say that the percentage of largemouth bass is the highest whereas the tadpole madtom is the least.

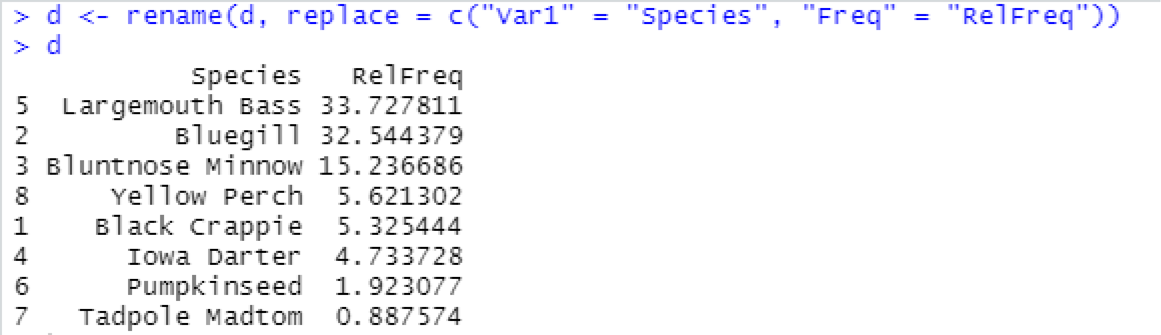
16. Rearrange the <u> cSpec Pct data frame in descending order of relative frequency. Save the rearranged data frame as the object <d>



```
> #Rearrange the <u> cSpecPct data frame in descending order of relative frequency. Save the rearranged data frame as the object <d>
> d <- u[order(-u$Freq),]
> d
              Var1      Freq
5  Largemouth Bass 33.727811
2         Bluegill 32.544379
3 Bluntnose Minnow 15.236686
8     Yellow Perch  5.621302
1    Black Crappie  5.325444
4      Iowa Darter  4.733728
6      Pumpkinseed  1.923077
7   Tadpole Madtom  0.887574
```

In this task, we have created an object 'd' and stored the dataframe 'u' in it in descending order, and printed it.

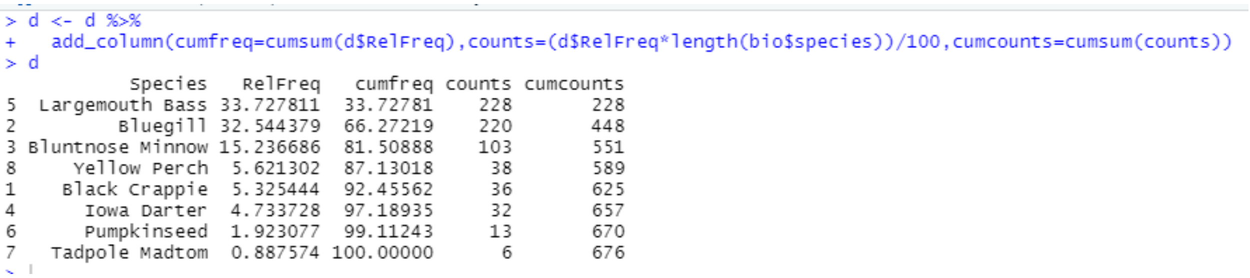
17. Rename the <d> columns Var 1 to Species, and Freq to RelFreq



```
> d <- rename(d, replace = c("Var1" = "Species", "Freq" = "RelFreq"))
> d
           Species   RelFreq
5  Largemouth Bass 33.727811
2         Bluegill 32.544379
3 Bluntnose Minnow 15.236686
8     Yellow Perch  5.621302
1    Black Crappie  5.325444
4      Iowa Darter  4.733728
6      Pumpkinseed  1.923077
7   Tadpole Madtom  0.887574
```

In this task, we have renamed the column names of d dataframe as per instruction from the instructor using rename() function. The rename() function changes the name of the variable in the dataframe.

18. Add new variables to <d> and call them cumfreq, counts, and cumcounts

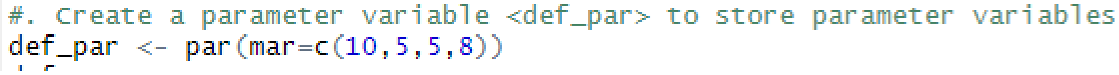


```
> d <- d %>%
+   add_column(cumfreq=cumsum(d$RelFreq),counts=(d$RelFreq*length(bio$species))/100,cumcounts=cumsum(counts))
> d
           Species   RelFreq   cumfreq counts cumcounts
5  Largemouth Bass 33.727811  33.72781    228       228
2         Bluegill 32.544379  66.27219    220       448
3 Bluntnose Minnow 15.236686  81.50888    103       551
8     Yellow Perch  5.621302  87.13018     38       589
1    Black Crappie  5.325444  92.45562     36       625
4      Iowa Darter  4.733728  97.18935     32       657
6      Pumpkinseed  1.923077  99.11243     13       670
7   Tadpole Madtom  0.887574 100.00000      6       676
```

In this task, we are going to add column into the existing dataframe using add_column() function. The column added to the dataframe are cumfreq, counts and cumcounts and displayed them.we have use cumsum() function to find the cumulative frequency and cumulative counts in the dataframe.

The cumfreq stores the cumulative frequencies of the dataframe. The counts stores the frequency of the species in the dataframe and the cumcounts store the cumulative counts in the dataframe.

19. Create a parameter variable <def_par> to store parameter variables

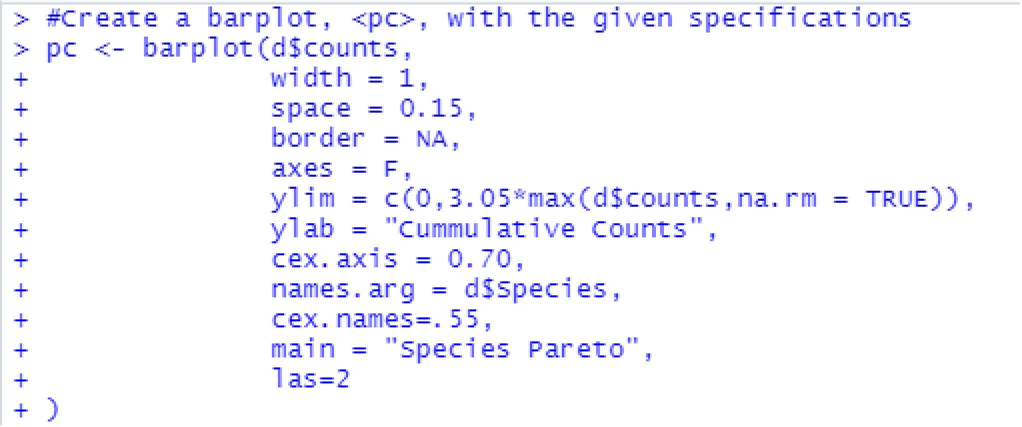


```
#. Create a parameter variable <def_par> to store parameter variables
def_par <- par(mar=c(10,5,5,8))
```

20. . Create a barplot, <pc>, with the following specifications:

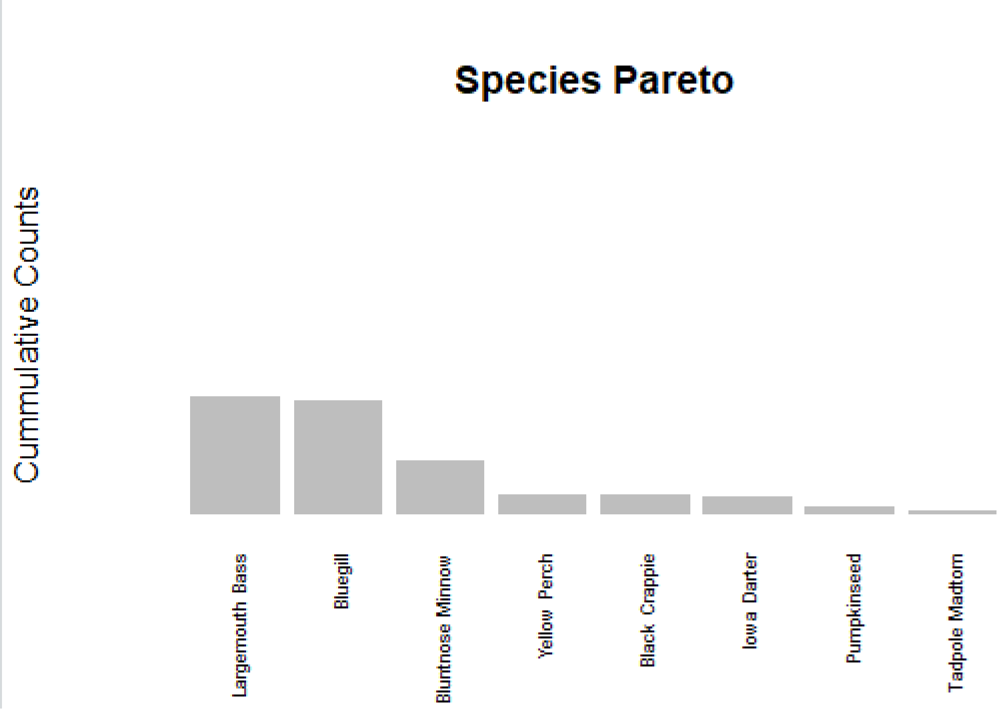
- d$counts of width 1, spacing of .15
- no boarder
- Axes: F
- Yaxis limit 0,3.05*max
- d$counts na.rm is true
- y label is Cummulative Counts
- scale x axis to 70%
- names.arg: d$Species
- Title of the barplot is “Species Pareto”
- las: 2)

Console screenshot:



```
> #Create a barplot, <pc>, with the given specifications
> pc <- barplot(d$counts,
+               width = 1,
+               space = 0.15,
+               border = NA,
+               axes = F,
+               ylim = c(0,3.05*max(d$counts,na.rm = TRUE)),
+               ylab = "Cummulative Counts",
+               cex.axis = 0.70,
+               names.arg = d$Species,
+               cex.names=.55,
+               main = "Species Pareto",
+               las=2
+ )
```

Barplot :

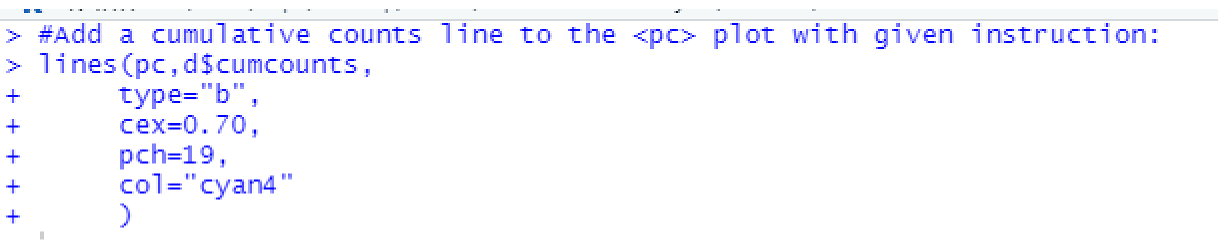


In this task, we will create a barplot pc, and using arguments we will change: the width, the space, remove the border, set axes to f, ylim, ylab, etc as per instructors instruction.

21.. Add a cumulative counts line to the <pc> plot with the following:

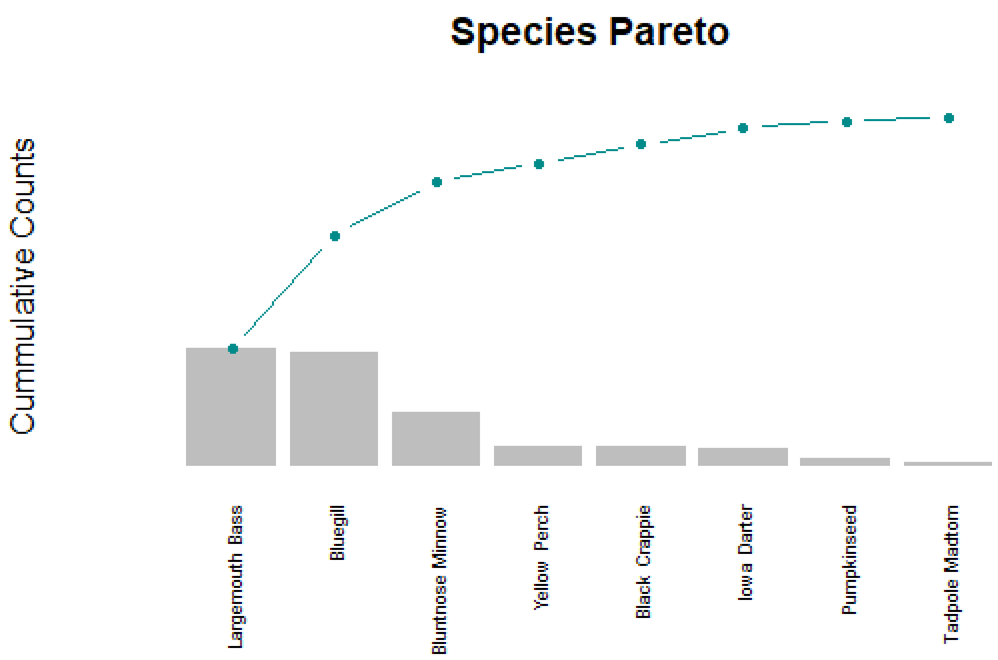
- Spec line type is b
- Scale plotting text at 70%
- Data values are solid circles with color cyan4

Console screenshot:



```
> #Add a cumulative counts line to the <pc> plot with given instruction:
> lines(pc,d$cumcounts,
+       type="b",
+       cex=0.70,
+       pch=19,
+       col="cyan4"
+       )
```

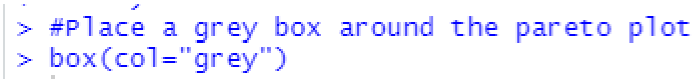
Barplot:



In this task, we have added a cumulative count line to the previous barplot using line() function. The line() function is used to add lines to existing plot. We have given arguments to line() function as per our requirements.

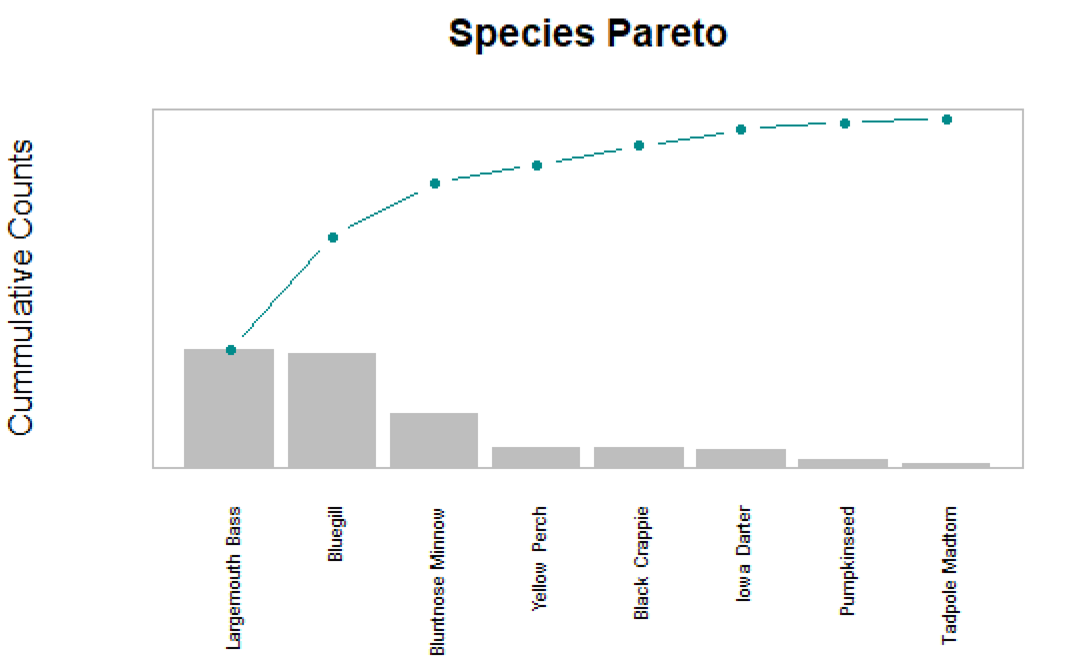
22. Place a grey box around the pareto plot

Console Screenshot :



```
> #Place a grey box around the pareto plot
> box(col="grey")
```

Barplot:

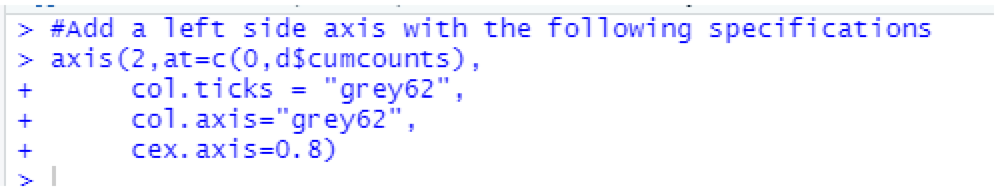


In this task we have just added grey box to our barplot.

23.. Add a left side axis with the following specifications

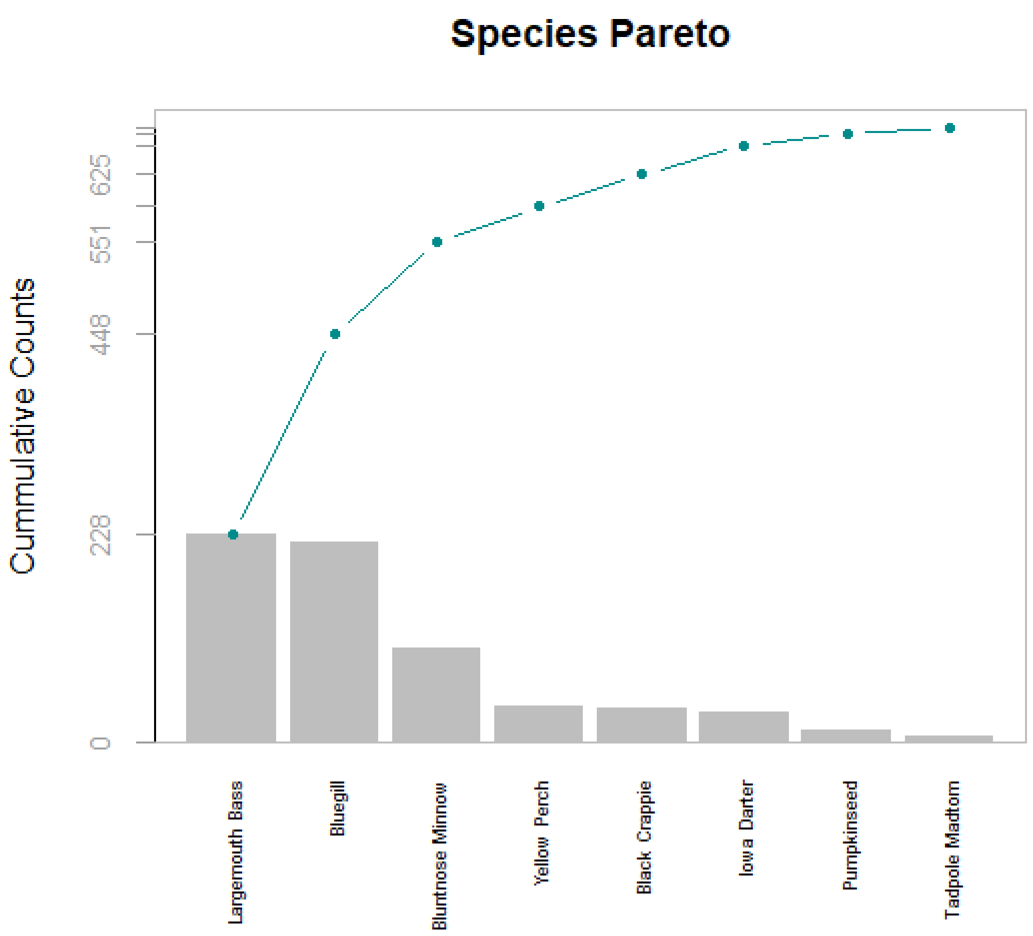
- Horizontal values at tick marks at cumcounts on side 2
- Tickmark color of grey62
- Color of axis is grey62
- Axis scaled to 80% of normal

Console Screenshot :



```
> #Add a left side axis with the following specifications
> axis(2,at=c(0,d$cumcounts),
+      col.ticks = "grey62",
+      col.axis="grey62",
+      cex.axis=0.8)
> 
```

Barplot :

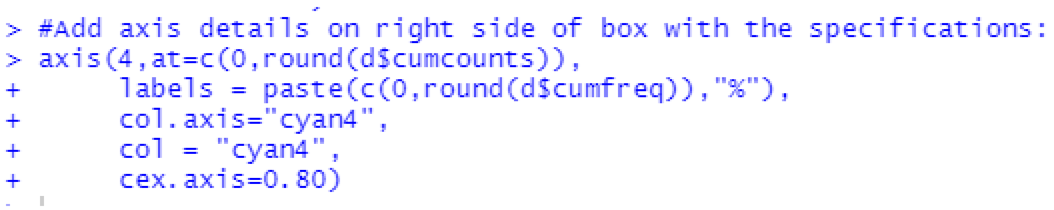


In this task, we have used axis() function and added left side axis to the bar plot and have changed the color of ticks, color of axis, and scaled axis size using arguments as per instructions. The left axis shows the cumulative counts of the fish.

24. Add axis details on right side of box with the specifications:

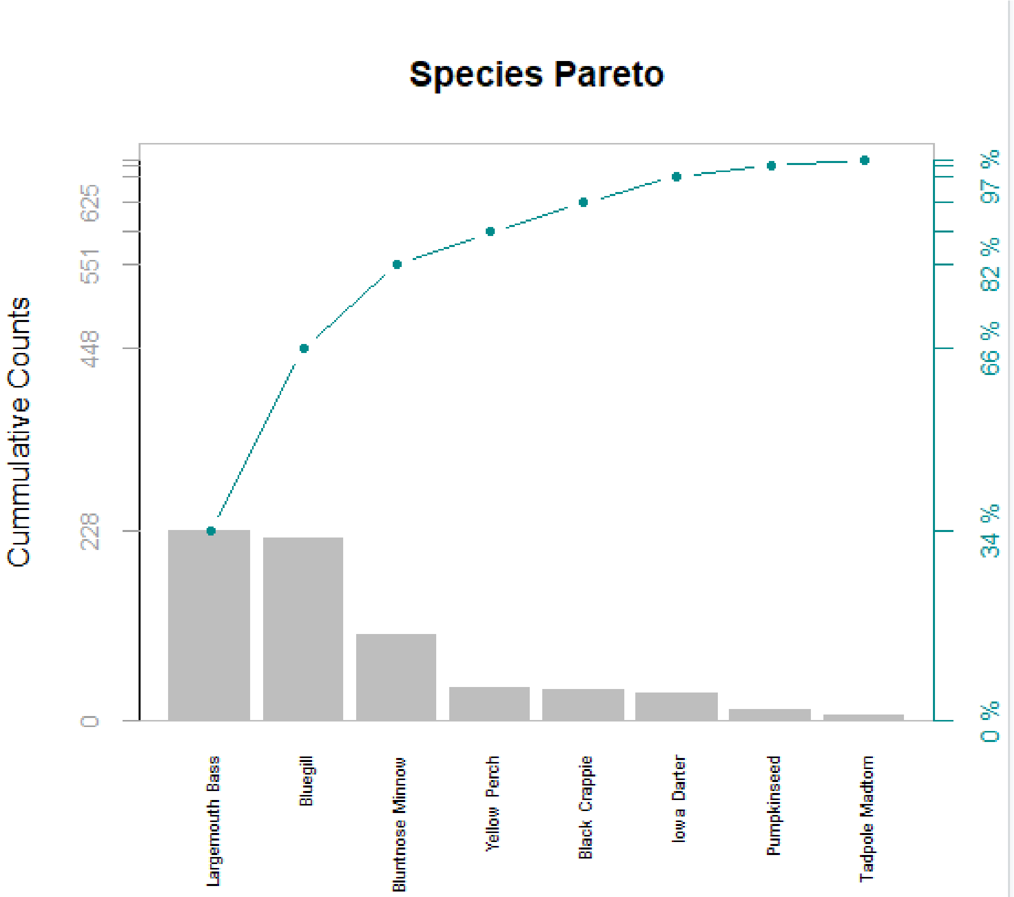
- Spec: Side 4
- Tickmarks at cumcounts with labels from 0 to cumfreq with %,
- Axis color of cyan5 and label color of cyan4
- Axis font scaled to 80% of nominal

Console Screenshot :



```
> #Add axis details on right side of box with the specifications:
> axis(4,at=c(0,round(d$cumcounts)),
+       labels = paste(c(0,round(d$cumfreq)),"%"),
+       col.axis="cyan4",
+       col = "cyan4",
+       cex.axis=0.80)
+
```

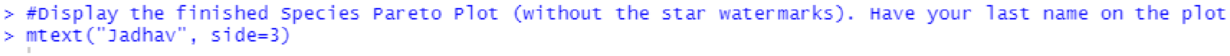
Barplot:



In this task we have added right axis using axis() function. The right axis shows the cumulative frequency of the fishes.

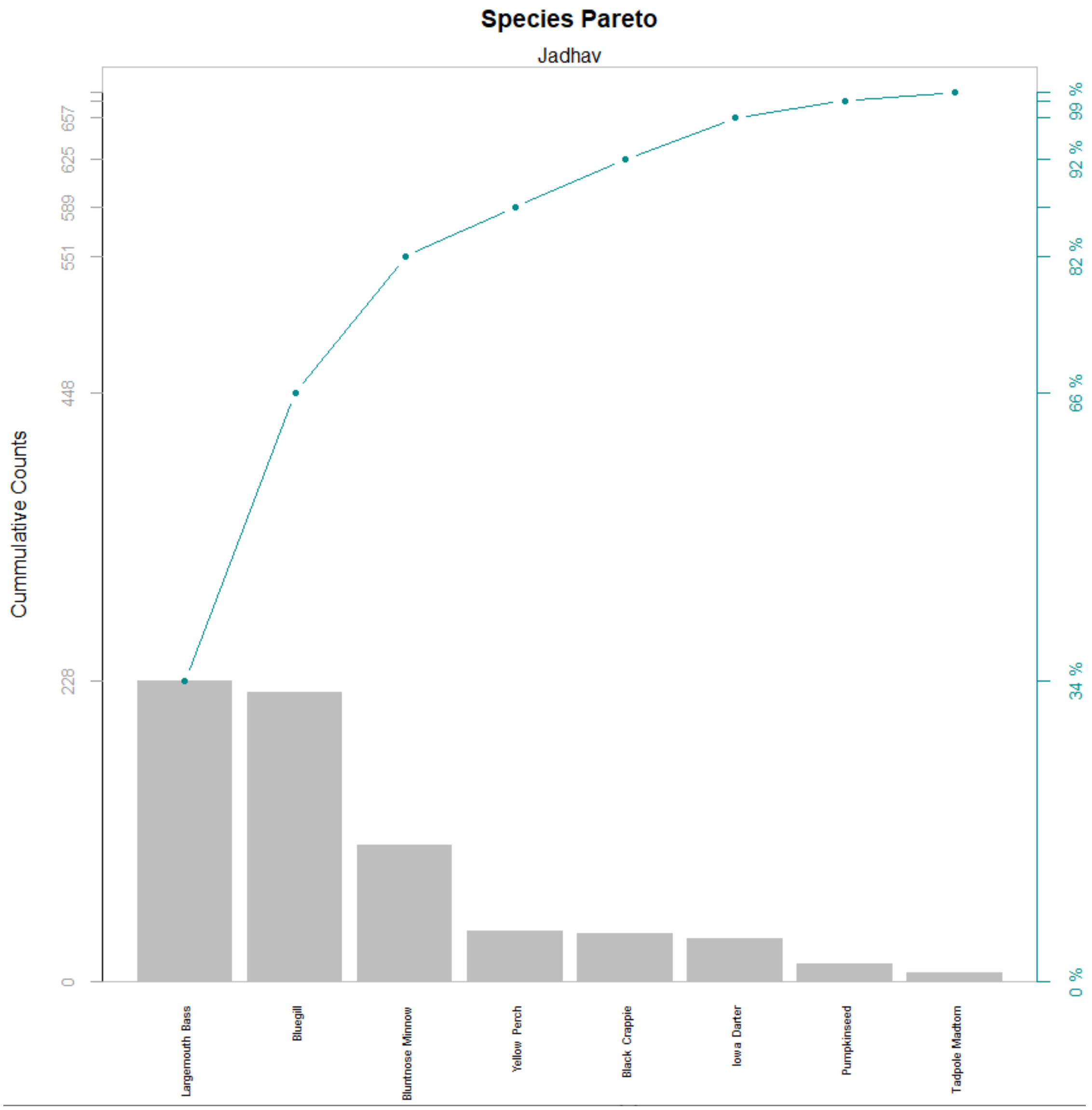
25. Display the finished Species Pareto Plot (without the star watermarks). Have your last name on the plot

Console screenshot :



```
> #Display the finished Species Pareto Plot (without the star watermarks). Have your last name on the plot
> mtext("Jadhav", side=3)
```

Barplot :



In this task we have used mtext() function to print my surname on the plot. The mtext() function allows us to write in one of the 4 margin of the figure. From task 20 to 25 we have learned and made the bar plot more understandable and presentable.

Analysis : From the barplot we can say that :

- the frequency of largemouth bass is the highest and the tadpole madtom is the least.
- The data is arranged in descending order which is way the plot is reverse J skewed.
- The left axis shows the cumulative count of fishes
- The right axis shows the frequency percentage of the fishes
- The line shows the cumulative count in graphical formal.

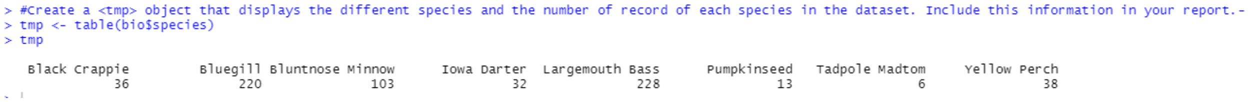
26. Summary

In this assignment, we learned: how to install and load multiple libraries,
import CSV files, create a subset from the main set, functions (like count(),
axis(),mtext(),cumsum(), line()), and how to create barplot and customization which help in making barplot more understandable.
We also learned about the dataset provided by the instructor that was about fish. We also learned how to created a subset from the main dataset in different classes like table and dataframe, which helped us to identify the quantity for each species of the fish. By doing certain operations on the dataset, we can say that the data of largemouth bass fish is the largest, whereas the least was of tadpole madtom. We also learned how to add columns to the existing dataset. We learned how to find cumulative sums. Lastly, we have combined the barplot with the cumulative counts, which helps us understand the dataset. We can see that there are only two species which are having more frequency compared to others.

This assignment helps to understand the different ways to look at the dataset, create graphical representation that helps to get the hidden information from dataset which in analysis.

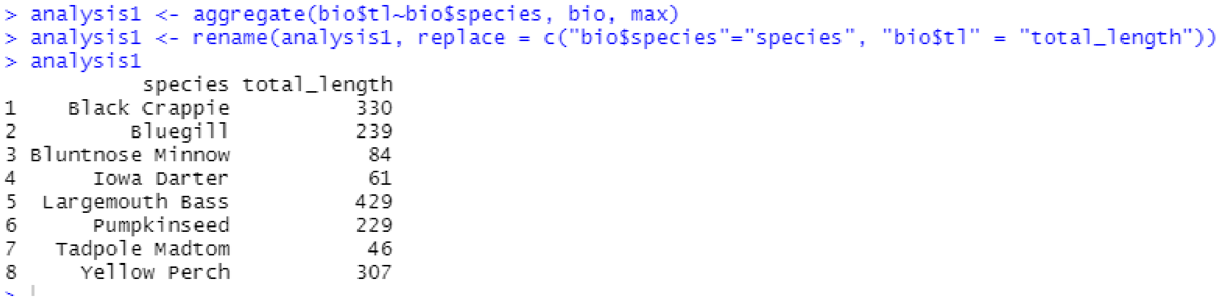
Data Analysis

A.



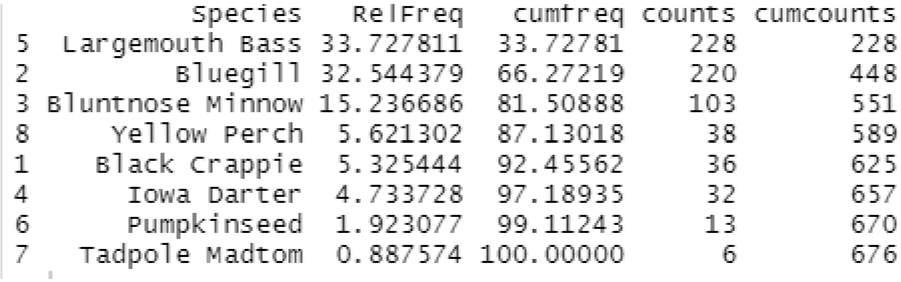
```
> #Create a <tmp> object that displays the different species and the number of record of each species in the dataset. Include this information in your report.-
> tmp <- table(bio$species)
> tmp

  Black Crappie        Bluegill Bluntnose Minnow      Iowa Darter  Largemouth Bass      Pumpkinseed   Tadpole Madtom     Yellow Perch 
             36              220              103               32              228               13                6               38 
```

- ➢ The Structural analysis of the tmp dataset reveals that there are only 2 fishes that dominates this study. The largemouth bass and the bluegill. The data also reveals that the tadpole madtom fish is the with lowest frequency.



```
> analysis1 <- aggregate(bio$tl~bio$species, bio, max)
> analysis1 <- rename(analysis1, replace = c("bio$species"="species", "bio$tl" = "total_length"))
> analysis1
           species total_length
1    Black Crappie          330
2         Bluegill          239
3 Bluntnose Minnow           84
4      Iowa Darter           61
5  Largemouth Bass          429
6      Pumpkinseed          229
7   Tadpole Madtom           46
8     Yellow Perch          307
```

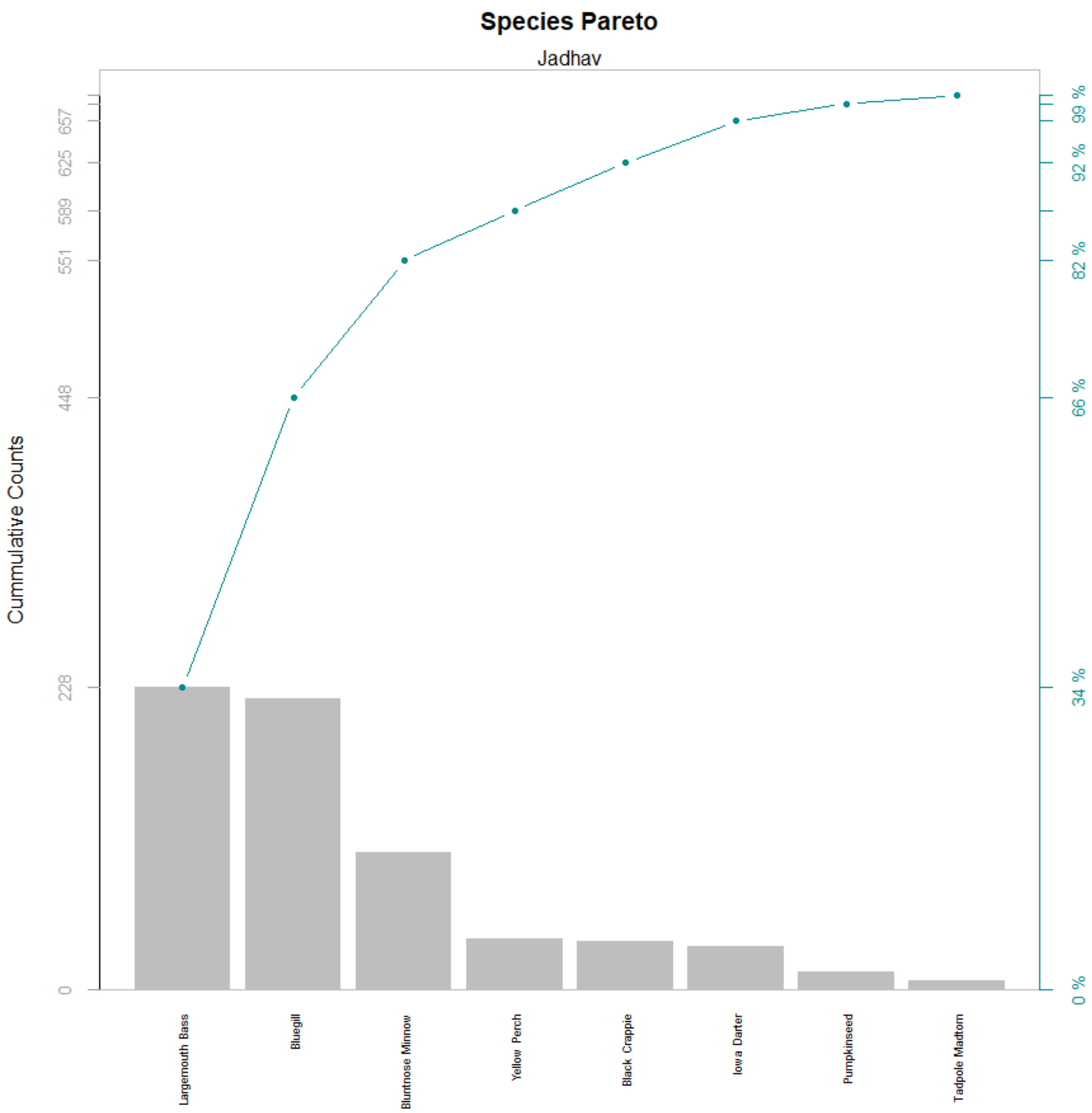
- ➢ To study in deep, I created a subset from main data set, which contains species name and the maximum total length of that fish. If compared about the length of fish with each other It was surprising to find that the largemouth bass fish has max length of 429 where as the tadpole madtom is the fish with the smallest length of 46.



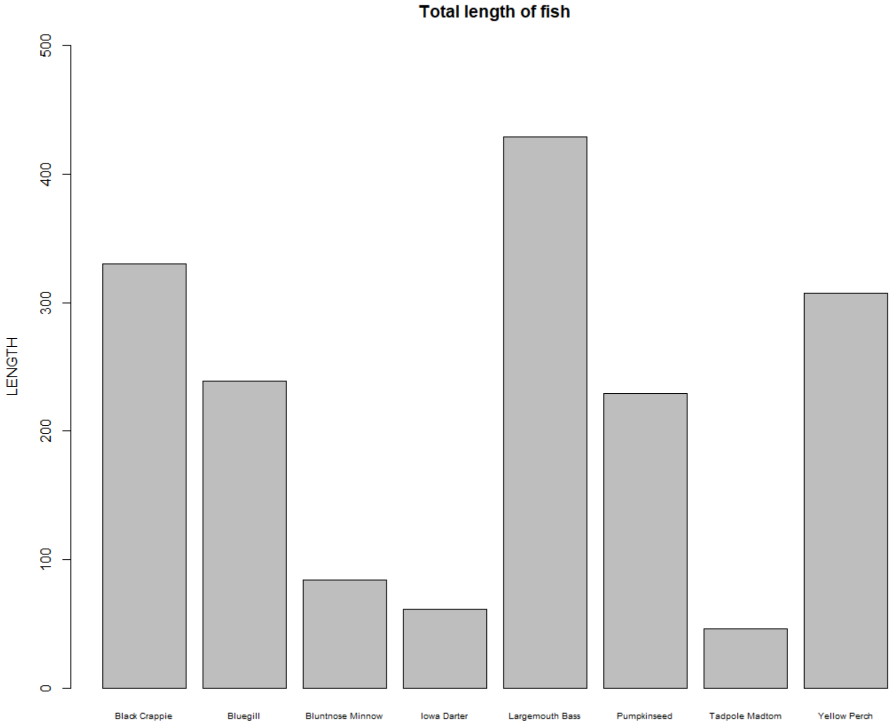
```
           Species   RelFreq   cumfreq counts cumcounts
5  Largemouth Bass 33.727811  33.72781    228       228
2         Bluegill 32.544379  66.27219    220       448
3 Bluntnose Minnow 15.236686  81.50888    103       551
8     Yellow Perch  5.621302  87.13018     38       589
1    Black Crappie  5.325444  92.45562     36       625
4      Iowa Darter  4.733728  97.18935     32       657
6      Pumpkinseed  1.923077  99.11243     13       670
7   Tadpole Madtom  0.887574 100.00000      6       676
```

- ➢ From the above screenshot you can also shows the dataframe with cumulative frequency of the fish and cumulative counts .

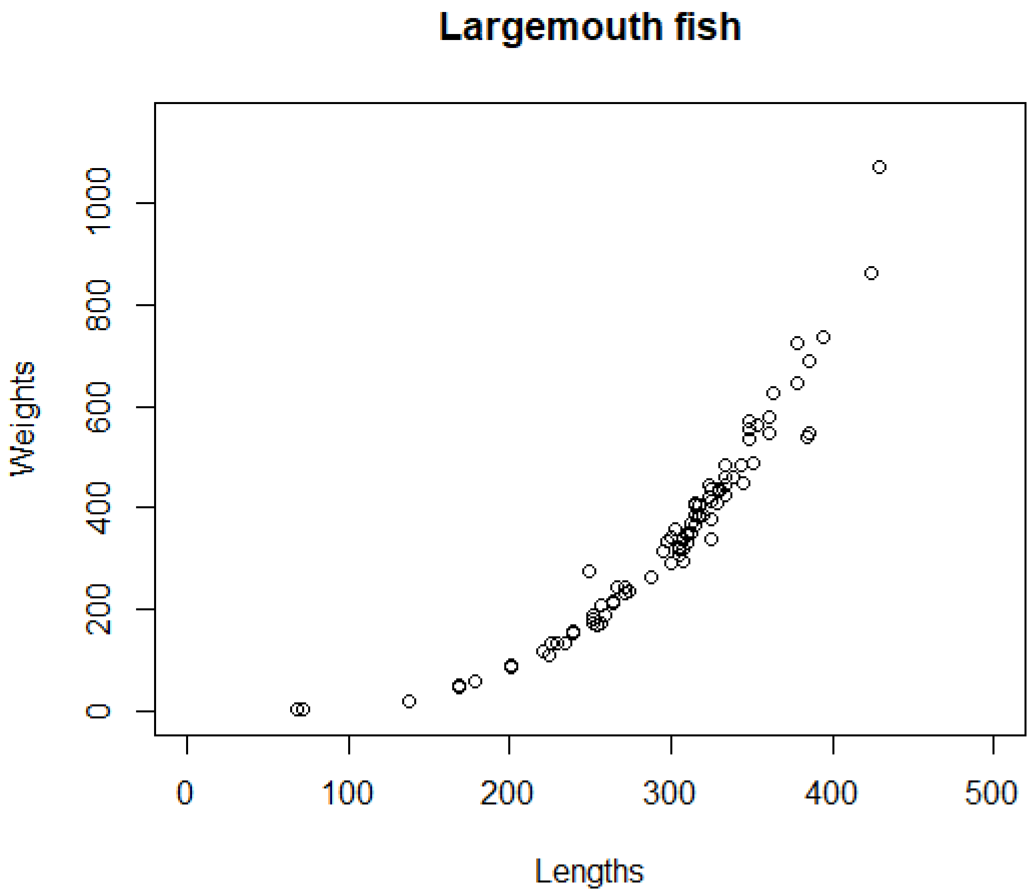
B.



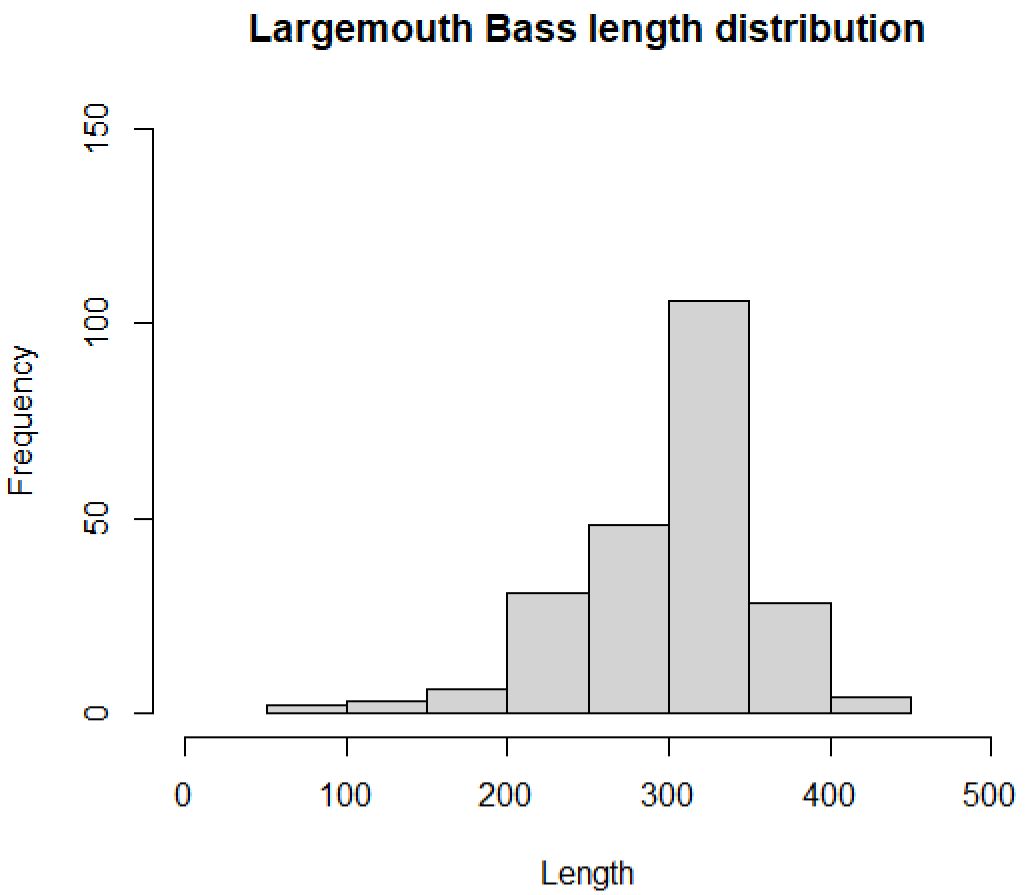
- The Pareto chart is combination of barplot with cumulative distribution graph. The left axis shows the cumulative count where as the right axis shows the cumulative frequency.
- The pareto chart leads us to investigation that Are tadpole madtom endangered species or their lifespan is less or they have been eaten a lot by other fishes?
- We can say that largemouth bass and bluegill are the 2 dominating species whereas the pumpkinseed and tadpole madtom are weak species among other species.



- From the above barplot we can see that highest lengths of each fish among their species. It can be seen that highest height of largemouth bass is above 400 and highest height of tadpole madtom is below 100.
- We can also say that the life span of largemouth bass is more than other fishes whereas the life span of tadpole is the least.



- For this plotting we created a subset from main data which contained data of fish largemouth bass and performed scatter plot which showed the weights and lengths of the fish.
- We can say that as the length of the fish increases the weight of the fish increases
- The frequency of fish is more between length 200 to 400 and height 100 to 600.



- Above is the histogram plot of largemouth bass length distribution
- From the plotting you can say that the frequency of fish from 300 to 350 length is the highest.
- The graph is symmetric.

C. From the plotting that I carried out in this assignment following are the key points I liked everyone to know:

- The polulation density of largemouth bass is the highest as per the dataset provided.
- The highest length of largemouth bass fish is 429 which is the highest length of the fishes whereas the tadpole madtom's length is 46 which is the smallest length.
- pumpkinseed and tadpole madtom are the weakest fish whereas the largemouth bass and bluegill are the dominating spieces.

Bibliography

- *Check if packages are installed (and install if not) in R*. posts by vikram. (2019, April 28). Retrieved October 11, 2021, from https://vbaliga.github.io/verify-that-r-packages-are-installed-and-loaded/.

- robk@statmethods.net, R. K.-. (n.d.). *Sorting data*. Quick-R: Sorting. Retrieved October 11, 2021, from https://www.statmethods.net/management/sorting.html.

- YouTube. (2019, September 20). *DPLYR Select & rename columns*. YouTube. Retrieved October 11, 2021, from https://www.youtube.com/watch?v=jCFpdvIDrIs.

- Erik Marsja, & *, N. (2021, September 16). *How to add a column to a dataframe in R with tibble & dplyr*. Erik Marsja. Retrieved October 11, 2021, from https://www.marsja.se/how-to-add-a-column-to-dataframe-in-r-with-tibble-dplyr/.

Appendix

```
#Print your name at the top of the script and load these libraries: FSA, FSAdata, magrittr, dplyr, tidyr plyr and tidyverse

Name <- "Rahul Avinash Jadhav"

Name

packages<-(c("FSA","FSAdata","magrittr","dplyr","tidyr","plyr","tidyverse"))

package.check <- lapply(

  packages,

  function(x) {

    if (!require(x, character.only = TRUE)) {

      install.packages(x, dependencies = TRUE)

      library(x, character.only = TRUE)

      }

    }

)


lapply(c("FSA","FSAdata","magrittr","dplyr","tidyr","plyr","tidyverse"),require, character.only = TRUE)


#Import the inchBio.csv and name the table <bio>

bio <- read.csv2("inchBio.csv",header=TRUE,sep=",")

bio


#Display the head, tail and structure of <bio>

headtail(bio)

str(bio)



#Create an object, <counts>, that counts and lists all the species records
```

```
counts <- count(bio,"species")

counts


#Display just the 8 levels (names) of the species

as.data.frame(counts$species)


#Create a <tmp> object that displays the different species and the number of record of each species in the dataset. Include this information in your report.-

tmp <- table(bio$species)

tmp


#Create a subset, <tmp2>, of just the species variable and display the first five records

tmp2 <- subset(bio,select=species)

head(tmp2, n=5)


#Create a table, <w>, of the species variable. Display the class of w

w <- table(bio$species)

w

class(w)


#Convert <w> to a data frame named <t> and display the results

t <- as.data.frame(w)

t

class(t)


#Extract and display the frequency values from the <t> data frame

t$Freq



#Create a table named <cSpec> from the bio species attribute (variable) and confirm that you created a table which displays the number of species in the dataset <bio>
```

```
cSpec <- table(bio$species)

cSpec

class(cSpec)
```

```
#Create a table named <cSpecPct> that displays the species and percentage of records for each species. Confirm you created a table class

cSpecPct <- prop.table(table(bio$species))*100

cSpecPct

class(cSpecPct)
```

```
#Convert the table, <cSpecPct>, to a data frame named <u> and confirm that <u> is a data frame

u <- as.data.frame(cSpecPct)

u

class(u)
```

```
#Create a barplot of <cSpec> with the following: titled Fish Count with the given specifications:

barplot(cSpec,

     main ="Fish Count",

     ylab = "COUNTS",

     col = "lightgreen",

     horiz = TRUE,

     cex.names = 0.6,

     xlim = c(0,250)

    )
```

```
#Create a barplot of <cSpecPct>, with the given specifications:
```

```
#left to color the ylabel
barplot(cSpecPct,
     ylim =c(0,40),
     ylab = "Frequency in Percentage",
     col.lab = "LightBlue",
     main = "Fish Relative Frequency",
     cex.names = 0.55)

#Rearrange the <u> cSpecPct data frame in descending order of relative frequency. Save the rearranged data frame as the object <d>
d <- u[order(-u$Freq),]
d

#Rename the <d> columns Var 1 to Species, and Freq to RelFreq

d <- rename(d, replace = c("Var1" = "Species", "Freq" = "RelFreq"))
d

#Add new variables to <d> and call them cumfreq, counts, and cumcounts
d <- d %>%

add_column(cumfreq=cumsum(d$RelFreq),counts=(d$RelFreq*length(bio$species))/100,cumcounts=cumsum(counts))
d

#. Create a parameter variable <def_par> to store parameter variables
def_par <- par(mar=c(10,5,5,8))
def_par

#Create a barplot, <pc>, with the given specifications
pc <- barplot(d$counts,
            width = 1,
            space = 0.15,
            border = NA,
            axes = F,
            ylim = c(0,3.05*max(d$counts,na.rm = TRUE)),
            ylab = "Cummulative Counts",
            cex.axis = 0.70,
            names.arg = d$Species,
            cex.names=.55,
            main = "Species Pareto",
            las=2
)


#Add a cumulative counts line to the <pc> plot with given instruction:
lines(pc,d$cumcounts,
    type="b",
    cex=0.70,
    pch=19,
    col="cyan4"
    )

#Place a grey box around the pareto plot
box(col="grey")

#Add a left side axis with the following specifications
axis(2,at=c(0,d$cumcounts),
    col.ticks = "grey62",
    col.axis="grey62",
    cex.axis=0.8)
```

```
#Add axis details on right side of box with the specifications:

axis(4,at=c(0,round(d$cumcounts)),

     labels = paste(c(0,round(d$cumfreq)),"%"),

     col.axis="cyan4",

     col = "cyan4",

     cex.axis=0.80)


#Display the finished Species Pareto Plot (without the star watermarks). Have your last name on the plot

mtext("Jadhav", side=3)



#data analysis


analysis1 <- aggregate(bio$tl~bio$species, bio, max)

analysis1 <- rename(analysis1, replace = c("bio$species"="species", "bio$tl" = "total_length"))

analysis1



barplot(analysis1$total_length,

  main ="Total length of fish",

  ylab = "LENGTH",

  cex.names = 0.6,

  names.arg = analysis1$species,

  ylim = c(0,500)

  )

box(col = "grey")


Largemouth <- filterD(bio,species=="Largemouth Bass")
```

```
summary(Largemouth)
plot(Largemouth$tl,Largemouth$w,
     xlab = "Lengths",
     ylab = "Weights",
     main = "Largemouth fish",
     xlim = c(0,500),
     ylim = c(0,1150)
     )

hist(Largemouth$tl,
     xlab = "Length",
     main = "Largemouth Bass length distribution",
     xlim = c(0,500),
     ylim = c(0,150))
```