ACKNOWLEDGEMENT

I undertook this project work, as the part of my XII-Informatics Practices course. I had tried to apply my best of knowledge and experience, gained during the study and class work experience. However, developing software system is generally a quite complex and time-consuming process. It requires a systematic study, insight vision and professional approach during the design and development. Moreover, the developer always feels the need, the help, and good wishes of the people near you, who have considerable experience and ideas.

I would like to extend my sincere thanks and gratitude to my principal **Ms. J. Bhuvaneswari.** I am very much thankful to my teacher **Ms. Ann Neethu** for giving valuable time and moral support to develop this software.

I would like to take the opportunity to extend my sincere thanks and gratitude to my parents for being a source of inspiration and providing time and freedom to develop a software project.

I feel indebted to my friends for the valuable suggestions during project work.

Dhruva .p XII-D

INDEX

SYSTEM SOFTWARE AND HARDWARE SPECIFICATIONS

SOFTWARE-

The following software are used-

- Windows 10.0
- Python 3.10

HARDWARE-

The following hardware are used-

- Windows 10 Operating System
- 16GB ram
- Intel Core i7

INTRODUCTION

Road accidents are a global problem, resulting in millions of deaths and injuries every year. According to the World Health Organization, approximately 1.35 million people lose their lives to road traffic accidents annually, and between 20 and 50 million more suffer non-fatal injuries.

These accidents are not only a leading cause of death but also a significant source of physical and psychological trauma for victims and their families.

Road accidents are scary for our lives. In recent times it has increased more. Every morning when you open a newspaper every second or third page will have news related to road accidents. The reason for increasing road accidents is due to the fact that people are buying more automobiles and have also become careless while driving vehicles.

Many a time we have seen that people are just avoiding following traffic rules. Especially in metropolitan cities, people are more careless while driving vehicles which ultimately lead to road accidents.

The foremost causes of road accidents in such metropolitan areas can be narrow roads and roads with potholes.

Thus a road accident damages the lives of life and material. People should be very careful while driving or walking on the road.

It is also seen that walking on the road is also equally dangerous because of heavy traffic it can also be harmful to the people walking on the road. Hence, such people should walk on the side of the road or walk on footpaths.

Risk factors:

Speeding- An increase in average speed is directly related both to the likelihood of a crash occurring and to the severity of the consequences of the crash.

Driving under the influence of alcohol and other psychoactive substances- Driving under the influence of alcohol and any psychoactive substance or drug increases the risk of a crash that results in death or serious injuries.

Non-use of motorcycle helmets, seat-belts, and child restraints -Correct helmet use can lead to a 42% reduction in the risk of fatal injuries and a 69% reduction in the risk of head injuries.

Distracted driving -Drivers using mobile phones are approximately 4 times more likely to be involved in a crash than drivers not using a mobile phone. Using a phone while driving slows reaction times (notably braking reaction time, but also a reaction to traffic signals), and makes it difficult to keep in the correct lane, and to keep the correct following distances.

Prevention from Road Accidents:

There are some most important points that every person should keep in mind while driving or walking or crossing the road. These points are as follows:

- Drive within the prescribed speed limit.
- Don't drink or smoke while driving.
- Follow all the traffic rules as they are for our safety.
- Never use mobile phones while driving a vehicle.
- Always drive in the proper lane.
- While riding a bike always wear a helmet.

Road Safety Initiative Taken of the Government:

Several initiatives have been taken by the Ministry which continues to implement a multi pronged road safety strategy based on Education, Engineering (both of roads and vehicles), Enforcement and Emergency Care consisting inter-alia of setting up Driver training schools, creating awareness, strengthening automobile safety standards, improving road infrastructure, carrying out road safety audit etc. High priority has been accorded to rectification of black spots.

Motor Vehicles Amendment Act 2019:

The Motor Vehicles Amendment Bill, 2019 was passed by both the Houses of Parliament in August 2019 and has now become an Act. The much needed amendments will improve road safety, facilitate citizens in their dealings with transport departments, strengthen rural transport, public transport and last mile connectivity through automation, computerization and online services and provide an efficient, safe and corruption free transport system in the country.

THEORETICAL BACKGROUND

What is **Python**?

Python is a widely used general-purpose, high-level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently. There are two major Python versions – Python 2 and Python 3. Both are quite different. Python 3 is used for this project.

Reason for increasing popularity:

1. Emphasis on code readability, shorter codes, ease of writing. Programmers can express logical concepts in fewer lines of code in comparison to languages such as C++ or Java.

2. Python supports multiple programming paradigms, like object-oriented, imperative and functional programming or procedural.

3. There exists inbuilt functions for almost all the frequently used concepts.

4. Philosophy is “Simplicity is the best”.

LANGUAGE FEATURES

- Interpreted
- There are no separate compilation and execution steps like C and C ++
- Directly run the program from the source code.
- Internally, Python converts the source code into an intermediate form called bytecodes which are then translated into native language of a specific computer.
- No need to worry about linking and loading with libraries, etc.
- Platform Independent
- Python programs can be developed and executed on multiple operating system platforms.
- Python can be used on Linux, Windows, Macintosh, Solaris and many more.
- Free and Open Source; Redistributable
- High-level Language
    - In Python, no need to take care about low-level details such as managing the memory used by the program
- Simple

- Closer to English Language
- More emphasis on the solution to the problem rather than the syntax
- Embeddable
    - Python can be used within C/C++ program to give scripting capabilities for the program's users.

- Robust
- Exceptional handling features
- Memory management techniques in built

What is a **CSV file**?

The CSV (Comma Separated Values) format refers to tabular data that has been saved as plaintext where data is separated by commas. In CSV Format:

Each row of the table is stored in one row where the number of rows in a CSV file are equal to the number of rows in the table.

The field-values of row are stored together with commas after every field value; but after the last field's value in a line/row, 0 comma is given, just the end of the line.

Advantages:

- A simple, compact and ubiquitous format for data storage.
- A common format for data interchange.
- It can be opened in popular spreadsheet packages like MSExcel, Calc etc.

- Nearly all spreadsheets and databases support import/export to csv format.

Reading from a CSV file to Data Frame:

- Read_csv() function is used to read data from a CSV file into a Data Frame.

Storing Data Frame's data into CSV file

- Python provides to csv function to save the data from Data Frame to CSV file.

PROJECT SYNOPSIS

The following are the types of analyses which have been done in this project

1. **DATA VISUALISATION**-
    - **Line chart**- line charts are plotted for the juveniles apprehended under the male, female, juvenile, and total categories.

    - **Bar Graph-** Bar charts are plotted for the juveniles apprehended under the male, female, juvenile, and total categories.

    - **Histogram-** Histograms are plotted for the juveniles apprehended under the male, female, juvenile, and total categories.

2. **DATA ANALYSIS**-

    The following are the columns used for data analysis-

    - Population in terms of number of accidents per lakh
    - Bottom most population in terms of number of persons injured
    - Persons killed for specific years

3. **DATA MANIPULATION-**
    - Adding a row
    - Adding a column
    - Deleting a row
    - Deleting a column
    - Renaming a column

4. **READ CSV-**

The csv file was read and converted into a dataframe

	A	B	C	D	E	F	G	H	I	J	K	L
1	Years	Total Number of Roa	Persons_Killed	Persons_Injured	Population	Total Number of Reg	Road Length	Accidents_Per_Lakh	Accidents_per_Ten	Number of Persons K	Number of Persons K	Number of Persons In
2	1970	114100	14500	70100	539000	1401	1188728	21.16883117	814.4182727	2.690166976	103.4975018	13.00556586
3	1980	153200	24000	109100	673000	4521	1491873	22.76374443	338.8630834	3.566121842	53.08560053	16.21099554
4	1990	282600	54100	244100	835000	19152	1983867	33.84431138	147.556391	6.479041916	28.24770259	29.23353293
5	1994	325864	64463	311500	904000	27660	2890950	36.04690265	117.8105568	7.130862832	23.3054953	34.4579646
6	1995	351999	70781	323200	924359	30295	2975035	38.08033459	116.1904605	7.657306306	23.36392144	34.96477018
7	1996	371204	74665	369502	941579	33786	3202515	39.42356403	109.8691766	7.929764789	22.09939028	39.24280384
8	1997	373671	76977	378361	959792	37332	3298788	38.93249787	100.0940212	8.020175205	20.6195757	39.42114541
9	1998	385018	79919	390674	978081	41368	3228356	39.3646334	93.0714562	8.171000152	19.31903887	39.94290861
10	1999	386456	81966	375051	996130	44875	3296650	38.79573951	86.11832869	8.228444079	18.2654039	37.65080863
11	2000	391449	78911	399265	1014825	48857	3316078	38.57305447	80.12137462	7.775823418	16.1514215	39.34323652
12	2001	405637	80888	405216	1028610	54991	3373520	39.43545173	73.76425233	7.863816218	14.70931607	39.39452271
13	2002	407497	84674	408711	1045547	58924	3426603	38.97452721	69.15637092	8.098535982	14.37003598	39.09063868
14	2003	406726	85998	435122	1062388	67007	3528654	38.28412972	60.69903144	8.094782697	12.8341815	40.95697617
15	2004	429910	92618	464521	1079117	72718	3621507	39.8390536	59.12016282	8.58275794	12.73659892	43.04639812
16	2005	439255	94968	465282	1095722	81502	3809156	40.0881793	53.89499644	8.667161926	11.65222939	42.46350808
17	2006	460920	105749	496481	1112186	89618	3880651	41.44270832	51.43163204	9.508211756	11.79997322	44.64010516
18	2007	479216	114444	513340	1128521	96707	4016401	42.46407466	49.55339324	10.14106073	11.83409681	45.48785534
19	2008	484704	119860	523193	1144734	105353	4109592	42.34206375	46.0076125	10.47055473	11.37698974	45.70432956
20	2009	486384	125660	515458	1160813	114951	4471510	41.90028885	42.31228958	10.82517167	10.93161434	44.40491276
21	2010	499628	134513	527512	1176742	127745.972	4582439	42.4585848	39.11105706	11.43096788	10.52972535	44.82817814
22	2011	497686	142485	511394	1210193.422	141865.607	4676838	41.12450051	35.08151204	11.77373777	10.04366055	42.257212
23	2012	490383	138258	509667	1208116	159490.5783	4865394	40.59072142	30.7468319	11.44409974	8.66872523	42.18692576
24	2013	486476	137572	494893	1223581	181508	5231922	39.75838134	26.80190405	11.24339132	7.57939044	40.44628022
25	2014	489400	139671	493474	1238887	190704	5402486	39.50319924	25.66280728	11.27390957	7.323968034	39.83204279
26	2015	501423	146133	500279	1254019	210023	5472144	39.98527933	23.87467087	11.65317272	6.957952224	39.89405264
27	2016	480652	150785	494624	1268961	230031	5603293	37.87760223	20.89509675	11.88255589	6.554986067	38.97866049
28	2017	464910	147913	470975	1283601	253311	5897671	36.21919896	18.35332852	11.52328488	5.839185823	36.69169781
29	2018	467044	151417	469418	1298043	272987.665	6125797	35.98062622	17.10861185	11.66502188	5.546660872	36.16351693
30	2019	449002	151113	451361	1312241	297189.64	NA	34.21642823	15.10826555	11.51564385	5.084733102	34.39619704
31	CAGR 2009/2019	-0.796521752	1.861596155	-1.31910615	1.233705539	9.964395517	3.559470593	NA	NA	NA	NA	NA

We will be covering the following in our project

- Data visualisation in the form of graphs
- Data manipulation of rows and columns
- Changing the value of a particular cell.
- Analysis of the data

The following libraries are used-

- Pandas- pandas is used to analyse and modify data
- Matplotlib- it is used to plot charts of the given data

The following are the functions used-

- def()- The def () keyword in python is used to define the function that users can use to build their own function like datavisual(), analysis(), read(), etc…
- datavisual()- This is the function used to plot various graphs like bar graph, linechart,etc

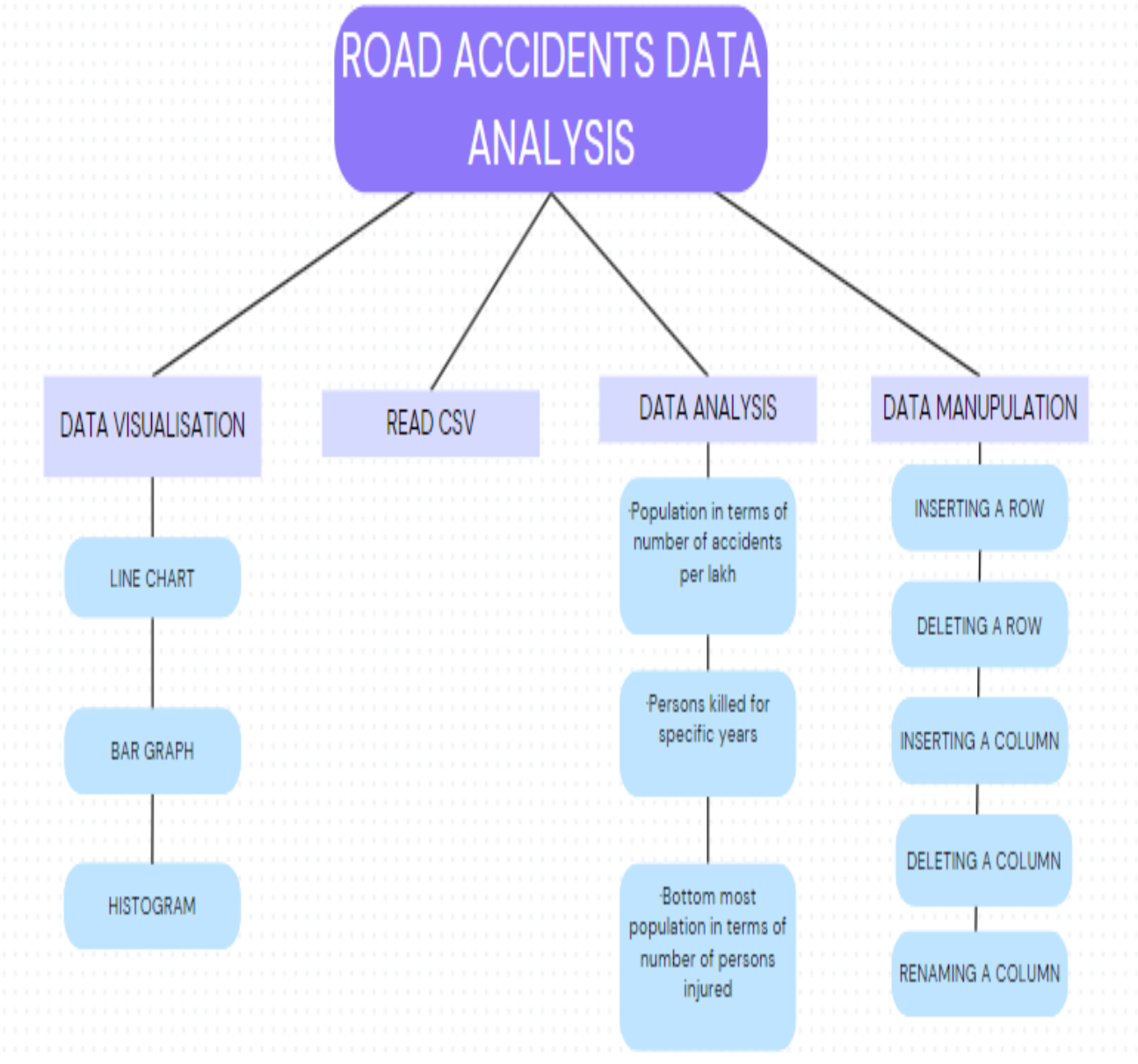
- dm()- The dm() function is used in data manipulation to change or add or drop values or columns

- read()- The read() function is used to read a csv file.

- analysis()- The analysis() function is used in data analysis to find out the maximum values, mean, median, etc…

  Apart from this, many in built functions like line(), plot(), bar(), etc… are also used in the project.

DESIGN WORK


ROAD ACCIDENTS DATA ANALYSIS
DATA VISUALISATION
LINE CHART
BAR GRAPH
HISTOGRAM
READ CSV
DATA ANALYSIS
·Population in terms of number of accidents per lakh
·Persons killed for specific years
·Bottom most population in terms of number of persons injured
DATA MANUPULATION
INSERTING A ROW
DELETING A ROW
INSERTING A COLUMN
DELETING A COLUMN
RENAMING A COLUMN

SOURCE CODE

```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~#
#class~~>12
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~#
#Topic~>ROAD ACCIDENTS DATA ANALYSIS
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~#



#IMPORTING LIBRARIES
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~#
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sys

#DATAFRAME USED
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#


df=pd.read_csv('C:\\Users\\IT-LAB\\Downloads\\DTA 12 D
ROAD ACCIDENTS.csv')
pd.set_option('display.max_columns',None)
```

```
#FUNCTION FOR THE MAIN MENU
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#
def menu():
    ans=True
    while ans:
        print("""
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
1-Data Visualisation
2-Data Analysis
3-Read CSV/EXCEL File
4-Data Manipulation
5-Exit
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::""")
        inp=int(input('enter your choice:'))
        if inp==1:
            datavisual()
        elif inp==2:
            raanalysis()
        elif inp==3:
            read_csv_excel()
        elif inp==4:
            manuplt()
        elif inp==5:
            ex=input('Are you sure you want to exit?(y/n)')
            if ex=='y'or ex=='Y':
                print("Exiting now........Done!\nHave A Nice Day!")
```

```
                sys.exit()
        else:
            print("\nInvalid Input Try Again")



#FUNCTION FOR PLOTTING GRAPHS/CHARTS
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
def datavisual():
    ans=True
    while ans:
        print("""
==========================================================
  DATA VISUALISATION
==========================================================
1. lINE CHART~> POPULATION OF INDIA VS TOTAL NUMBER OF PERSONS INJURED
2. lINE CHART~> POPULATION OF INDIA VS TOTAL NUMBER OF PERSONS KILLED
3. BAR CHART~>  POPULATION OF INDIA VS NO. OF ACCIDENTS PER LAKH POPULATION
4. BAR CHART~>  POPULATION OF INDIA VS NO. OF ACCIDENTS PER TEN THOUSAND VEHICLES
5. HISTOGRAM~>  TOTAL NUMBER OF ROAD ACCIDENTS
6. EXIT TO MAIN MENU
==========================================================
==========================================================""")
        ans = input("Please enter your choice:")
```

```
        if ans == '1':
            line_chart1()
        elif ans == '2':
            line_chart2()
        elif ans == '3':
            bar_chart1()
        elif ans == '4':
            bar_chart2()
        elif ans == '5':
            dhistogm()
        elif ans == '6':
            menu()
        else:
            print("\nInvalid choice.Try again")
            continue


#PLOT LINE CHART~> POPULATION OF INDIA VS
TOTAL NUMBER OF PERSONS INJURED
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#


def line_chart1():
    df=pd.read_csv("C:\\Users\\IT-LAB\\Downloads\\DTA 12
D ROAD ACCIDENTS.csv")


    df.sort_values(by='Persons_Injured',ascending=False,
                   inplace=True)



    df=df.loc[:,['Population','Persons_Injured']]
```

```
    df1=df.head(10)
    population_of_india=df1['Population']
    total_no_of_persons_injured=df1['Persons_Injured']
    plt.plot(population_of_india,total_no_of_persons_injured,
            linestyle=':',color='green', marker='.')
    x=np.arange(len(population_of_india))
    #plt.xticks(x,population_of_india,rotation=30)
    plt.xlabel('Population~ ~ ~~~~~>', fontsize=18,color='r')
    plt.ylabel('Persons Injured', color='r', fontsize=18)
    plt.title('Population vs Persons Injured',color =
'dimgrey',fontsize = 18)
    plt.show()


#PLOT LINE CHART~> POPULATION OF INDIA VS
TOTAL NUMBER OF PERSONS KILLED
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#


def line_chart2():
    df=pd.read_csv("C:\\Users\\IT-LAB\\Downloads\\DTA 12
D ROAD ACCIDENTS.csv")
    df.sort_values(by= 'Persons_Killed',ascending = False,
inplace = True)
    df = df.loc[:,['Population','Persons_Killed']]
    df1 = df.head(10)
    population_of_india=df1['Population']
    total_no_of_persons_killed=df1['Persons_Killed']
    plt.plot(population_of_india,total_no_of_persons_killed,
            linestyle='dashed',color='orange',marker='+')
    x=np.arange(len(population_of_india))
    #plt.xticks(x,population_of_india)
```

```
    plt.xlabel('Population~ ~ ~~~~~>', fontsize=18,color='blue')
    plt.ylabel('Persons Killed', color='blue', fontsize=18)
    plt.title('Population vs Persons Killed',color =
'dimgrey',fontsize = 18)
    plt.show()


#PLOT BAR CHART~> POPULATION OF INDIA VS NO.
OF ACCIDENTS PER LAKH POPULATION
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#


def bar_chart1():
    df=pd.read_csv("C:\\Users\\IT-LAB\\Downloads\\DTA 12
D ROAD ACCIDENTS.csv")
    df.sort_values('Accidents_Per_Lakh',ascending = False)
    df1 = df.head(n=10)
    x = np.arange(len(df1))
    population_of_india=df1['Population']
    accidents_per_lakh_population=df1['Accidents_Per_Lakh']
    plt.bar(x+0.25,accidents_per_lakh_population,width = 0.6,
label = 'No. Of Accidents Per Lakh Population',color = 'gold')
    plt.xticks(x,population_of_india,rotation = 30)
    plt.title('Population vs No. Of Accidents Per Lakh
Pop',color = 'dimgrey',fontsize = 18)
    plt.xlabel('Population~ ~ ~~~~~>',fontsize = 18, color = 'r')
    plt.ylabel('No. Of Accidents Per Lakh Population',fontsize =
18, color = 'r')
    plt.legend()
    plt.show()


#PLOT BAR CHART~> POPULATION OF INDIA VS NO.
OF ACCIDENTS PER TEN THOUSAND VEHICLES
```

```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#


def bar_chart2():
    df=pd.read_csv("C:\\Users\\IT-LAB\\Downloads\\DTA 12
D ROAD ACCIDENTS.csv")

df.sort_values('Accidents_per_TenThousand_Vehicles',ascend
ing = False)
    df1 = df.head(n=10)
    x = np.arange(len(df1))
    population_of_india=df1['Population']

accidents_per_tenthousand_vehicles=df1['Accidents_per_Ten
Thousand_Vehicles']
    plt.bar(x+0.25,accidents_per_tenthousand_vehicles,width =
0.6, label = 'Accidents Per Ten Thousand Vehicles',color =
'cyan')
    plt.xticks(x,population_of_india,rotation = 30)
    plt.title('Population vs No. Of Accidents Per Ten Thousand
Vehicles',color = 'dimgrey',fontsize = 18)
    plt.xlabel('Population~ ~ ~~~~~>',fontsize = 18, color =
'blue')
    plt.ylabel('No. Of Accidents Per Ten Thousand
Vehicles',fontsize = 18, color = 'blue')
    plt.legend()
    plt.show()


#PLOT HISTOGRAM~> TOTAL NUMBER OF ROAD
ACCIDENTS
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#


def dhistogm():
```

```
    df=pd.read_csv("C:\\Users\\IT-LAB\\Downloads\\DTA 12
D ROAD ACCIDENTS.csv")
    i = df['Persons_Injured']
    k = df['Persons_Killed']
    dat = ['Injured','Killed']
    plt.hist([i,k],rwidth=0.9,color=['pink','purple'],label=dat)
    plt.title('Total Number Of Road Accidents',color =
'dimgrey', fontsize = 18)
    plt.xlabel('Injured/Killed',fontsize=18,color='r')
    plt.ylabel('Total Road Accidents',fontsize=18,color='r')
    plt.legend()
    plt.show()



#FUNCTION FOR ANALYSIS OF ROAD ACCIDENTS
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#


def raanalysis():
    while True:
        print('<------------------->')
        print('Data Frame Analysis')
        print('<------------------->')
        mn = """        1) To print records of population in terms
of number of accidents per lakh
        2) To print records of bottom most population in terms of
number of persons injured
        3) To print records of column specified by the user
        4) To print records of persons killed for specific years
        5) To go back to the main menu"""
        print(mn)
```

```
        x = int(input("Enter your choice : "))
        print("---------x------------------x------------------x----------
---------x")
        df = pd.read_csv("C:\\Users\\IT-LAB\\Downloads\\DTA
12 D ROAD ACCIDENTS.csv")
        if x ==1:
            df = df.sort_values('Accidents_Per_Lakh',ascending =
False, ignore_index=True)
            df = df.loc[:,['Population','Accidents_Per_Lakh']]
            nok = int(input('Enter the number of records to be
displayed : '))
            print('Top',nok,'records from dataframe')
            print(df.head(nok))
            print("---------x------------------x------------------x-------
------------x")
        elif x ==2 :
            df = df.sort_values('Persons_Injured',ascending =
False, ignore_index=True)
            df = df.loc[:,['Population','Persons_Injured']]
            noi = int(input('Enter the number of records to be
displayed : '))
            print('Bottom',nok,'records from dataframe')
            print(df.tail(noi))
            print("---------x------------------x------------------x-------
------------x")
        elif x ==3 :
            print('Name of the columns~~>',df.columns)
            clm = eval(input('Enter the column names in a list in
quotes :'))
            print(df[clm])
            print("---------x------------------x------------------x-------
------------x")
        elif x ==4 :
```

```
            print('Years')
            print(df['Years'].values)
            entry = eval(input('Enter year in the form of list in
quotes:'))
            print('Total number of persons killed in years :')
            for ent in entry :

print(df.loc[df['Years']==ent,['Years','Persons_Killed']])
            print("---------x------------------x------------------x-------
------------x")
        elif x ==5 :
            menu()
            break



#READING CSV FILE/EXCEL FILE AND DISPLAYING
DATAFRAME
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#
def read_csv_excel():
    ans = True
    while ans:
        print("""1) Read csv file and display DataFrame
2) Press 2 to go back to main menu""")
        ans = int(input('Enter your choice:'))
        if ans ==1:
            df = pd.read_csv("C:\\Users\\IT-
LAB\\Downloads\\DTA 12 D ROAD ACCIDENTS.csv")
            print(df)
            print('Done!')
        elif ans ==2:
```

```
            menu()



#MANIPULATION OF DATA
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~#
def manuplt():
    df = pd.read_csv("C:\\Users\\IT-LAB\\Downloads\\DTA 12
D ROAD ACCIDENTS.csv")
    ans = True
    while ans:
        print("""DATA MANIPULATION\n
    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    1) Inserting a row
    2) Deleting a row
    3) Inserting a column
    4) Deleting a column
    5) Renaming a column
    6) Exit to main menu""")
        ans = int(input('Enter your choice[12 columns/30
rows]:'))
        pd.set_option('display.max_columns',None)

        if ans ==1:
            print('Enter the input in following format:')
            col = df.columns
            print(col)
            lst = eval(input('Enter the row values in list:'))
            sr = pd.Series(lst,index=col)
            row_df1 = pd.DataFrame([sr])
```

```
            df = pd.concat([row_df1,df],ignore_index = True)
            print(df.loc[[0,1,2]])
            print('Row Added Successfully!!')
            print('~'*30)
        elif ans ==2:
            print('DataFrame before deleting the row')
            print(df)
            inp = int(input("Enter the row's index you want to be
deleted :"))
            df1 = df.drop(inp,axis=0)
            print("~"*30)
            print("DataFrame after row index no.",inp,"is deleted--
>")
            print("~"*30)
            print(df1.loc[[0,1,2]])
            print("~"*30)
        elif ans ==3:
            pd.set_option('display.width',500)
            pd.set_option('display.max_columns',None)
            clname = input('Enter the column name :')
            inp = int(input('Enter where you want to put the
column (column index):'))
            df.insert(inp,clname,'Nan')
            print(df.loc[[0,1,2]])
            print("~"*30)
        elif ans ==4:
            pd.set_option('display.width',500)
            pd.set_option('display.max_columns',None)
            print('DataFrame before deleting the column')
            print(df)
```

```
            inp = input('Column name you want to delete :')
            df = df.drop(inp,axis=1)
            print('DataFrame after deleting the column',inp,':')
            print(df.loc[[0,1,2]])
            print("~"*30)
        elif ans ==5:
            pd.set_option('display.width',500)
            pd.set_option('display.max_columns',None)
            print("~"*30)
            print('DataFrame before changing the column name')
            print('~'*30)
            print(df)
            oldclm = input('Enter the column name you want to
rename :')
            newclm = input('Enter the new column name :')
            df = df.rename(columns={oldclm:newclm})
            print("~"*30)
            print('DataFrame after changing the column name')
            print("~"*30)
            print(df.loc[[0,1,2]])
            print("~"*30)
        elif ans ==6:
            print('Returning to main menu.....Done!!')
            menu()


menu()
#<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~END OF
PROGRAM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>#
```

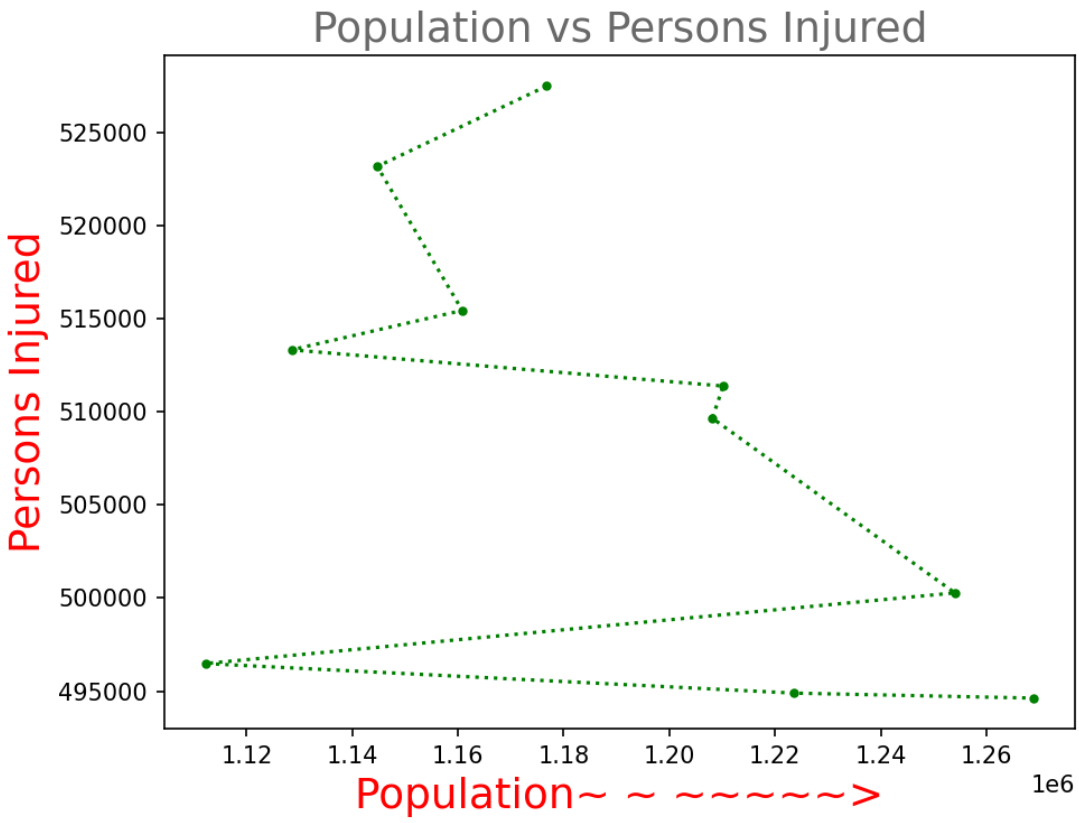
OUTPUT

```
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
1-Data Visualisation
2-Data Analysis
3-Read CSV/EXCEL File
4-Data Manipulation
5-Exit
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
enter your choice:
```

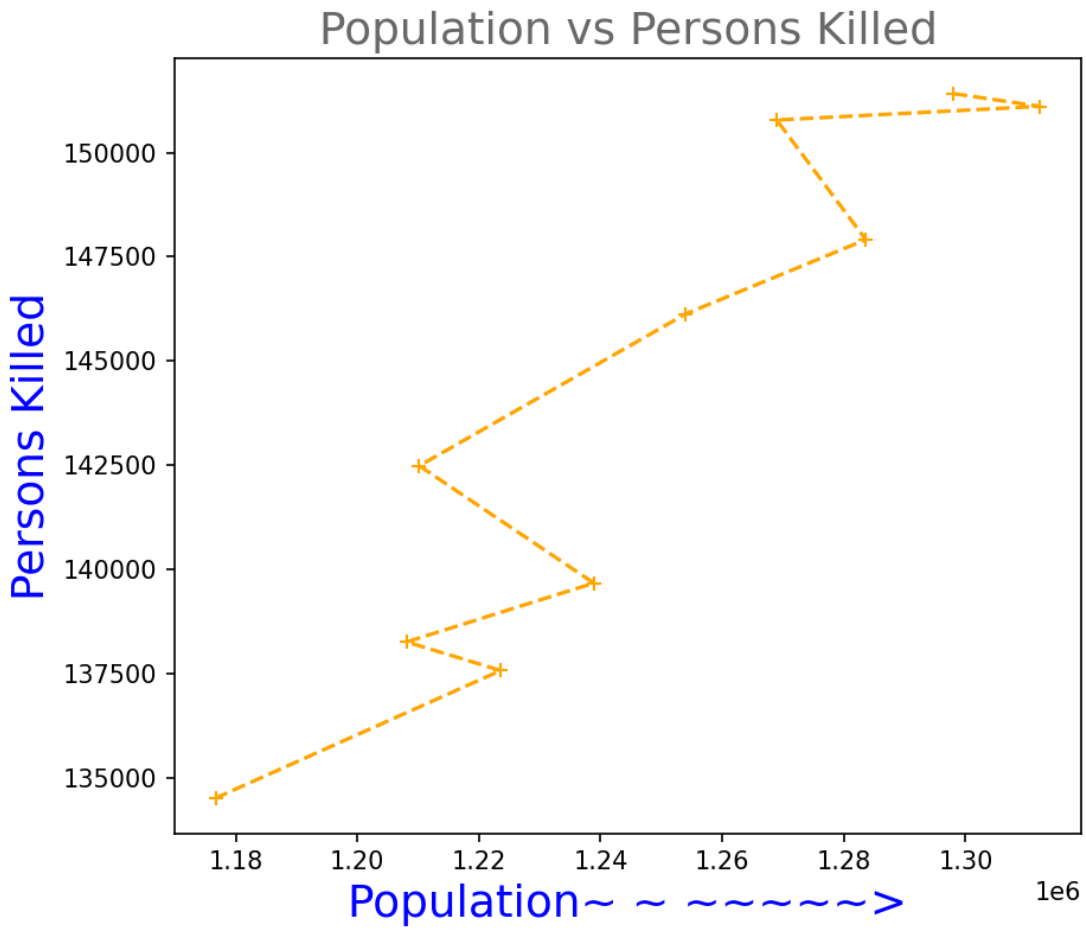
```
enter your choice:1

=========================================================
  DATA VISUALISATION
=========================================================
1. lINE CHART~> POPULATION OF INDIA VS TOTAL NUMBER OF PERSONS INJURED
2. lINE CHART~> POPULATION OF INDIA VS TOTAL NUMBER OF PERSONS KILLED
3. BAR CHART~>  POPULATION OF INDIA VS NO. OF ACCIDENTS PER LAKH POPULATION
4. BAR CHART~>  POPULATION OF INDIA VS NO. OF ACCIDENTS PER TEN THOUSAND VEHICLES
5. HISTOGRAM~>  TOTAL NUMBER OF ROAD ACCIDENTS
6. EXIT TO MAIN MENU
=========================================================
=========================================================
Please enter your choice:
```

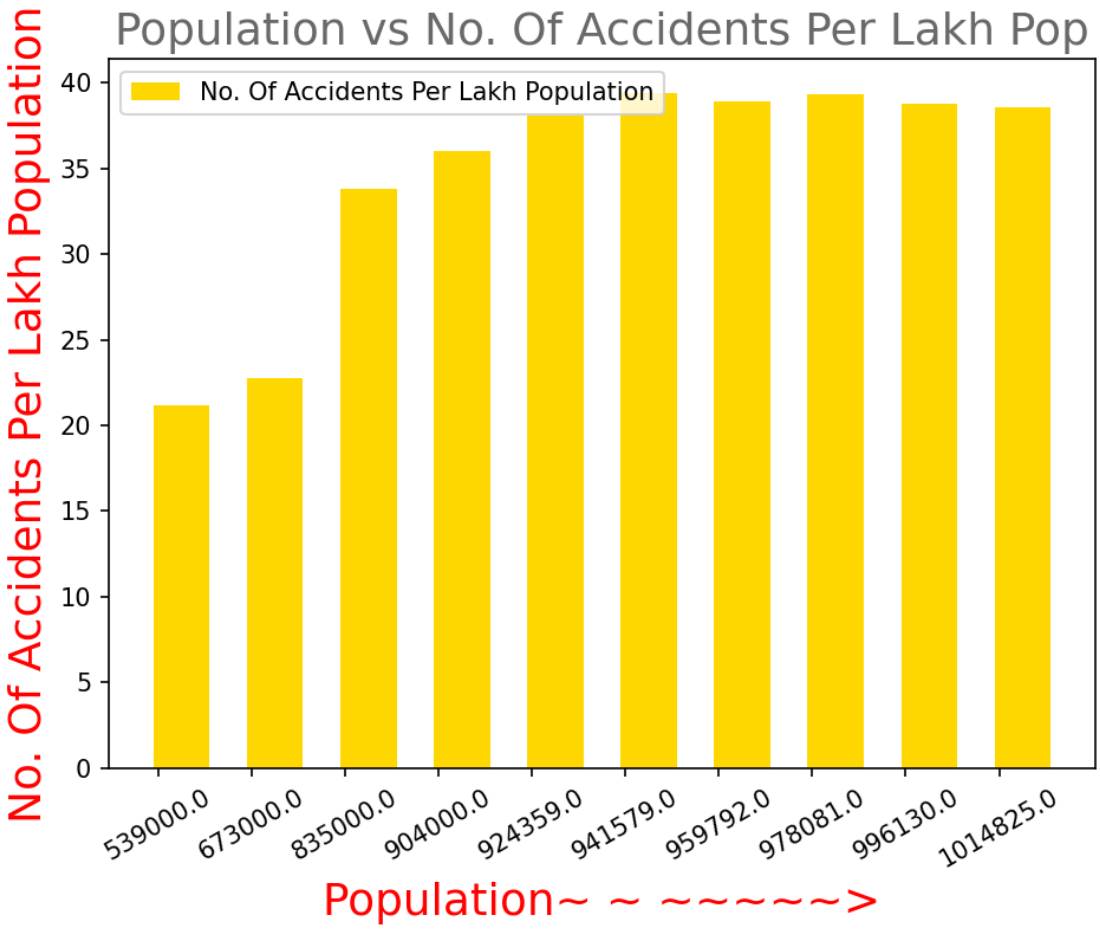
```
Please enter your choice:1
```



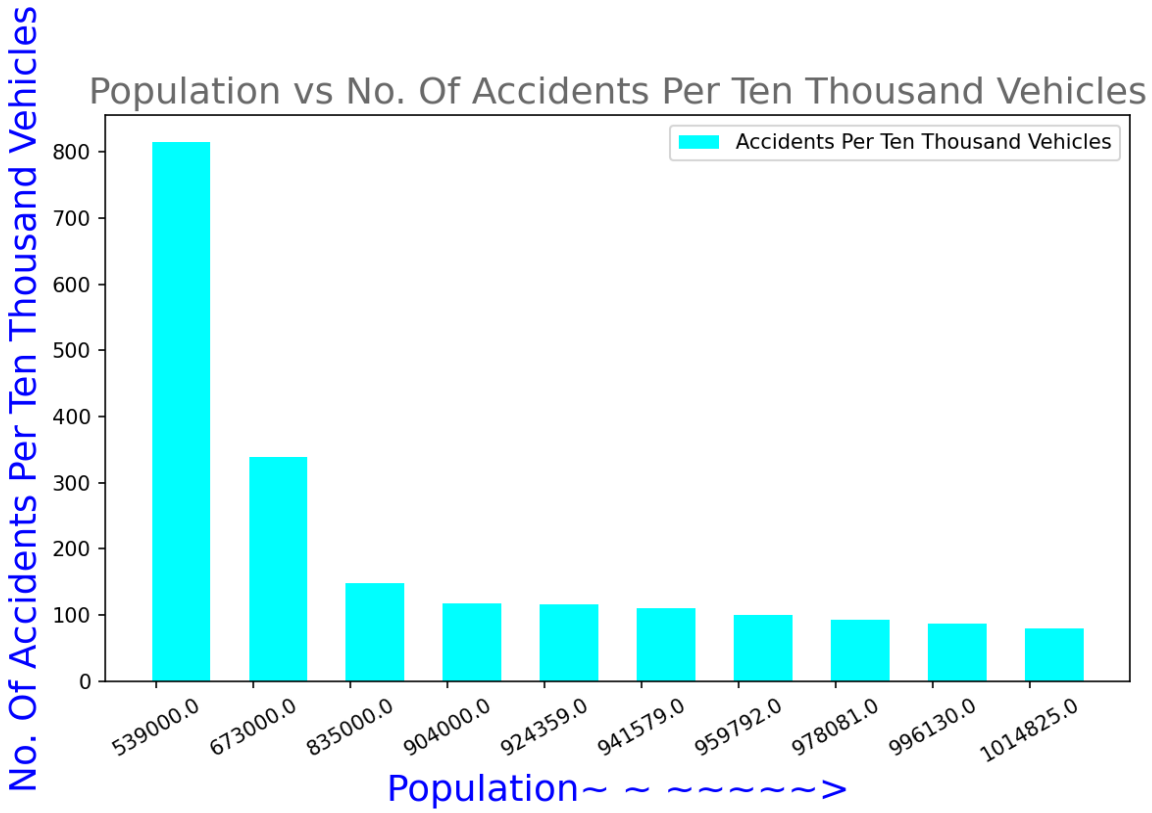
```
Please enter your choice:2
```



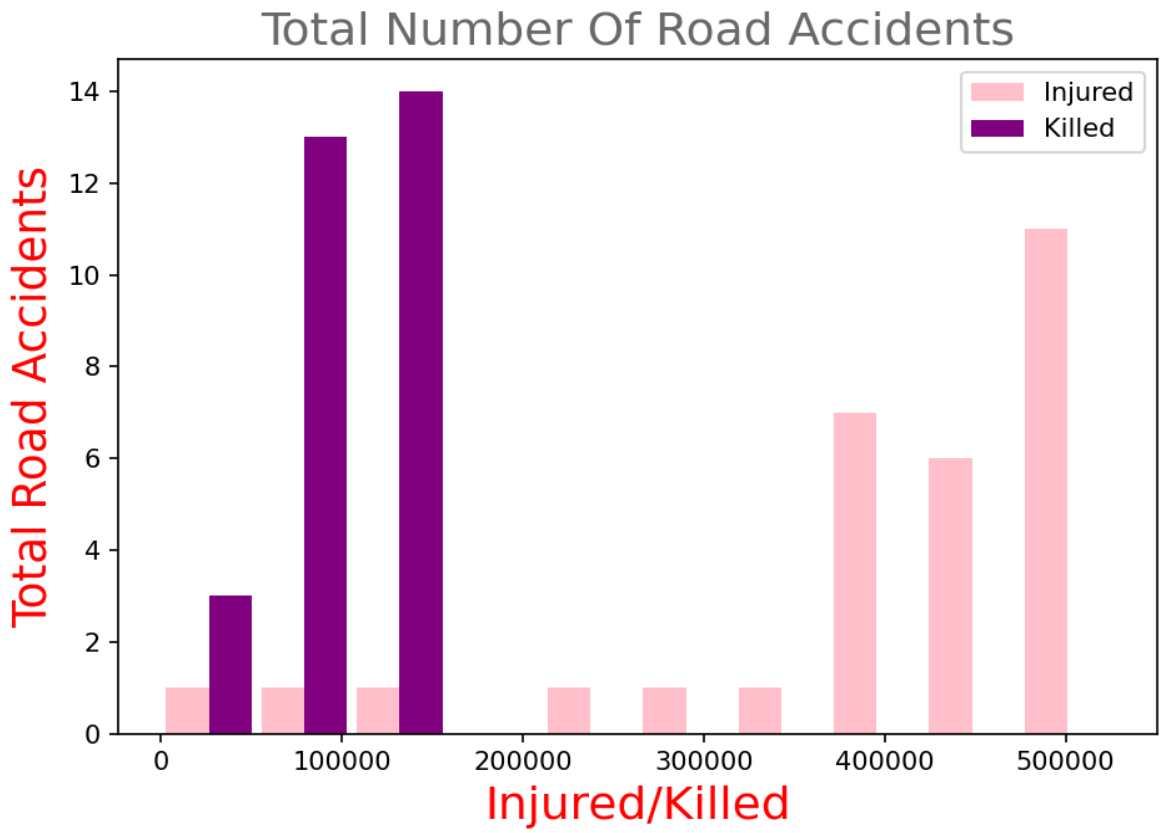
```
Please enter your choice:3
```



```
Please enter your choice:4
```




```
Please enter your choice:5
```



```
enter your choice:2
<------------------>
Data Frame Analysis
<------------------>
        1) To print records of population in terms of number of accidents per lakh
        2) To print records of bottom most population in terms of number of persons injured
        3) To print records of column specified by the user
        4) To print records of persons killed for specific years
        5) To go back to the main menu
Enter your choice : 1
```

```
---------x------------------x------------------x------------------x
Enter the number of records to be displayed : 2
Top 2 records from dataframe
   Population  Accidents_Per_Lakh
0   1128521.0           42.464075
1   1176742.0           42.458585
---------x------------------x------------------x------------------x
```

```
Enter your choice : 2
---------x------------------x------------------x------------------x
Enter the number of records to be displayed : 2
Bottom 2 records from dataframe
       Population  Persons_Injured
28  539000.000000     70100.000000
29       1.233706        -1.319106
---------x------------------x------------------x------------------x
```

```
Enter your choice : 3
---------x------------------x------------------x------------------x
Name of the columns~~> Index(['Years', 'Total Number of Road Accidents ', 'Persons_Killed',
       'Persons_Injured', 'Population',
       'Total Number of Registered Motor Vehicles ', 'Road Length ',
       'Accidents_Per_Lakh', 'Accidents_per_TenThousand_Vehicles',
       'Number of Persons Killed Per Lakh Population',
       'Number of Persons Killed Per Ten Thousand Vehicles ',
       'Number of Persons Injured per Lakh Population '],
      dtype='object')
Enter the column names in a list in quotes :['Population']
   Population
0    539000.0
1    673000.0
2    835000.0
3    904000.0
4    924359.0
5    941579.0
---------x------------------x------------------x------------------x
```

```
Enter your choice : 4
---------x------------------x------------------x------------------x
Years
['1970' '1980' '1990' '1994' '1995' '1996' '1997' '1998' '1999' '2000'
 '2001' '2002' '2003' '2004' '2005' '2006' '2007' '2008' '2009' '2010'
 '2011' '2012' '2013' '2014' '2015' '2016' '2017' '2018' '2019'
 'CAGR 2009/2019']
Enter year in the form of list in quotes:['2007']
Total number of persons killed in years :
   Years  Persons_Killed
16  2007        114444.0
---------x------------------x------------------x------------------x
```

```
enter your choice:3
1) Read csv file and display DataFrame
2) Press 2 to go back to main menu
Enter your choice:1
```

```
  Years  Total Number of Road Accidents   Persons_Killed  Persons_Injured  \
0  1970                      114100.0         14500.0          70100.0
1  1980                      153200.0         24000.0         109100.0
2  1990                      282600.0         54100.0         244100.0
3  1994                      325864.0         64463.0         311500.0
4  1995                      351999.0         70781.0         323200.0
5  1996                      371204.0         74665.0         369502.0

   Population  Total Number of Registered Motor Vehicles   Road Length   \
0    539000.0                                  1401.0     1188728.0
1    673000.0                                  4521.0     1491873.0
2    835000.0                                 19152.0     1983867.0
3    904000.0                                 27660.0     2890950.0
4    924359.0                                 30295.0     2975035.0
5    941579.0                                 33786.0     3202515.0

   Accidents_Per_Lakh  Accidents_per_TenThousand_Vehicles  \
0           21.168831                          814.418273
1           22.763744                          338.863083
2           33.844311                          147.556391
3           36.046903                          117.810557
4           38.080335                          116.190461
5           39.423564                          109.869177

   Number of Persons Killed Per Lakh Population  \
0                                      2.690167
```

```
1                                   3.566122
2                                   6.479042
3                                   7.130863
4                                   7.657306
5                                   7.929765

   Number of Persons Killed Per Ten Thousand Vehicles   \
0                                   103.497502
1                                    53.085601
2                                    28.247703
3                                    23.305495
4                                    23.363921
5                                    22.099390

   Number of Persons Injured per Lakh Population
0                                   13.005566
1                                   16.210996
2                                   29.233533
3                                   34.457965
4                                   34.964770
5                                   39.242804
Done!
```

```
enter your choice:4
DATA MANIPULATION

    ~~~~~~~~~~~~~~~~~~~
    1) Inserting a row
    2) Deleting a row
    3) Inserting a column
    4) Deleting a column
    5) Renaming a column
    6) Exit to main menu
Enter your choice[12 columns/30 rows]:
```

```
Enter your choice[12 columns/30 rows]:1
Enter the input in following format:
Index(['Years', 'Total Number of Road Accidents ', 'Persons_Killed',
       'Persons_Injured', 'Population',
       'Total Number of Registered Motor Vehicles ', 'Road Length ',
       'Accidents_Per_Lakh', 'Accidents_per_TenThousand_Vehicles',
       'Number of Persons Killed Per Lakh Population',
       'Number of Persons Killed Per Ten Thousand Vehicles ',
       'Number of Persons Injured per Lakh Population '],
      dtype='object')
Enter the row values in list:[1,2,3,4,5,6,7,8,9,10,11,12]
  Years  Total Number of Road Accidents   Persons_Killed  Persons_Injured  \
0     1                          2.0           3.0             4.0
1  1970                     114100.0        14500.0         70100.0
2  1980                     153200.0        24000.0        109100.0

   Population  Total Number of Registered Motor Vehicles   Road Length   \
0         5.0                                  6.0          7.0
1    539000.0                                  1401.0     1188728.0
2    673000.0                                  4521.0     1491873.0

   Accidents_Per_Lakh  Accidents_per_TenThousand_Vehicles  \
0            8.000000                           9.000000
1           21.168831                         814.418273
2           22.763744                         338.863083

   Number of Persons Killed Per Lakh Population  \
0                                     10.000000
1                                      2.690167
2                                      3.566122

   Number of Persons Killed Per Ten Thousand Vehicles   \
0                                         11.000000
1                                        103.497502
2                                         53.085601

   Number of Persons Injured per Lakh Population
0                                      12.000000
1                                      13.005566
2                                      16.210996
Row Added Successfully!!
```

```
Enter your choice[12 columns/30 rows]:2
DataFrame before deleting the row
Squeezed text (191 lines).
Enter the row's index you want to be deleted :29
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
DataFrame after row index no. 29 is deleted-->
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  Years  Total Number of Road Accidents   Persons_Killed  Persons_Injured  \
0  1970                     114100.0        14500.0         70100.0
1  1980                     153200.0        24000.0        109100.0
2  1990                     282600.0        54100.0        244100.0

   Population  Total Number of Registered Motor Vehicles   Road Length   \
0    539000.0                                  1401.0     1188728.0
1    673000.0                                  4521.0     1491873.0
2    835000.0                                 19152.0     1983867.0

   Accidents_Per_Lakh  Accidents_per_TenThousand_Vehicles  \
0           21.168831                          814.418273
1           22.763744                          338.863083
2           33.844311                          147.556391

   Number of Persons Killed Per Lakh Population  \
0                                      2.690167
1                                      3.566122
2                                      6.479042

   Number of Persons Killed Per Ten Thousand Vehicles   \
0                                         103.497502
1                                          53.085601
2                                          28.247703

   Number of Persons Injured per Lakh Population
0                                      13.005566
1                                      16.210996
2                                      29.233533
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Enter your choice[12 columns/30 rows]:3
```

```
Enter the column name :Games
Enter where you want to put the column (column index):1
  Years Games  Total Number of Road Accidents   Persons_Killed  Persons_Injured  Population  Total Number of Registered Motor Vehi
cles   Road Length   Accidents_Per_Lakh  Accidents_per_TenThousand_Vehicles  Number of Persons Killed Per Lakh Population  Num
ber of Persons Killed Per Ten Thousand Vehicles   Number of Persons Injured per Lakh Population
0  1970   Nan                  114100.0        14500.0        70100.0    539000.0                                1401.0    1188728.0
21.168831                  814.418273                          2.690167                        103.497502
       13.005566
1  1980   Nan                  153200.0        24000.0        109100.0    673000.0                                4521.0    1491873.0
22.763744                  338.863083                          3.566122                        53.085601
       16.210996
2  1990   Nan                  282600.0        54100.0        244100.0    835000.0                                19152.0    1983867.0
 33.844311                  147.556391                          6.479042                        28.247703
       29.233533
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Enter your choice[12 columns/30 rows]:4

```
DataFrame before deleting the column
Squeezed text (124 lines).
Column name you want to delete :Persons_Killed
DataFrame after deleting the column Persons_Killed :
  Years  Total Number of Road Accidents   Persons_Injured  Population  Total Number of Registered Motor Vehicles   Road Length   Ac
cidents_Per_Lakh  Accidents_per_TenThousand_Vehicles  Number of Persons Killed Per Lakh Population  Number of Persons Killed Pe
r Ten Thousand Vehicles   Number of Persons Injured per Lakh Population
0  1970                  114100.0        70100.0    539000.0                                1401.0    1188728.0        21.168831
        814.418273                          2.690167                        103.497502                                 13.005566
1  1980                  153200.0        109100.0    673000.0                                4521.0    1491873.0        22.763744
        338.863083                          3.566122                        53.085601                                 16.210996
2  1990                  282600.0        244100.0    835000.0                                19152.0    1983867.0        33.844311
        147.556391                          6.479042                        28.247703                                 29.233533
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Enter your choice[12 columns/30 rows]:5

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
DataFrame before changing the column name
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Squeezed text (124 lines).
Enter the column name you want to rename :Population
Enter the new column name :Pop
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
DataFrame after changing the column name
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  Years  Total Number of Road Accidents   Persons_Injured       Pop  Total Number of Registered Motor Vehicles   Road Length   Accide
nts_Per_Lakh  Accidents_per_TenThousand_Vehicles  Number of Persons Killed Per Lakh Population  Number of Persons Killed Per Te
n Thousand Vehicles   Number of Persons Injured per Lakh Population
0  1970                  114100.0        70100.0  539000.0                                1401.0    1188728.0        21.168831
        814.418273                          2.690167                        103.497502                                 13.005566
1  1980                  153200.0        109100.0  673000.0                                4521.0    1491873.0        22.763744
        338.863083                          3.566122                        53.085601                                 16.210996
2  1990                  282600.0        244100.0  835000.0                                19152.0    1983867.0        33.844311
        147.556391                          6.479042                        28.247703                                 29.233533
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
1-Data Visualisation
2-Data Analysis
3-Read CSV/EXCEL File
4-Data Manipulation
5-Exit
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
enter your choice:5
Are you sure you want to exit?(y/n)y
Exiting now.....Done!
Have A Nice Day!
```

BIBLIOGRAPHY

www.python.org

www.w3schools.com

https://pandas.pydata.org