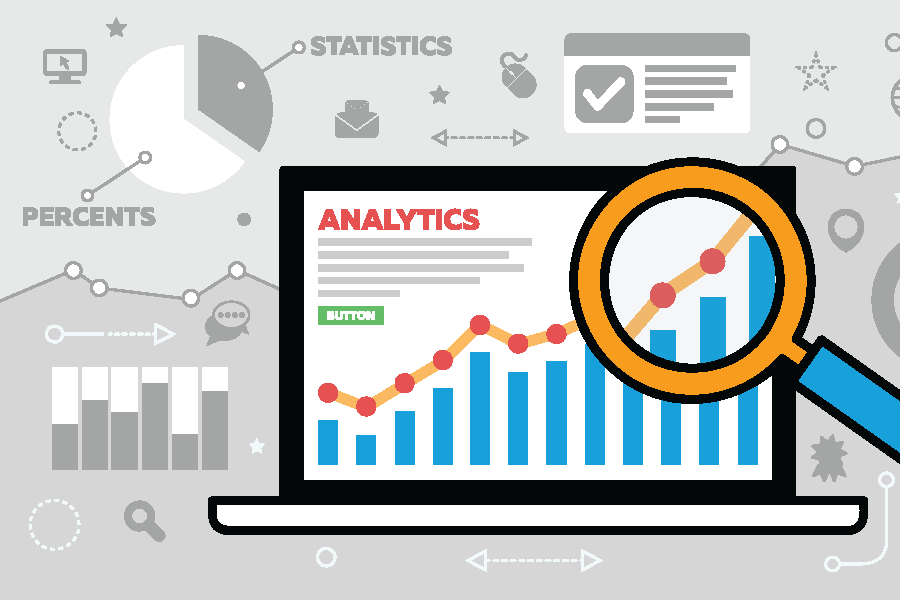
Application Design for an app launch on Google PlayStore



Submitted by: Samyak Sheth

Reeba Mistry

Dhyan Somaiya

Shourya Dagaria

Aryan Baretto

Date of Submission: 18th November 2019

College Name & Logo: Patkar college

Certificate of completion

This is to certify that, Mr./Ms. has successfully implemented an application designed to study the data and generate insights for an app launch on Google PlayStore.

The application has been accepted as a completed project as it meets all the requirements specified.

Acknowledgments:

We would like to express our sincere gratitude to our professor Mr. Junaid Khateeb for providing his invaluable guidance, comments and suggestions throughout the course of the project. We would especially like to thank Bobby ma’am for constantly motivating us to work harder and to be dedicated towards this case study.

Secondly, we would like to thank sir for giving us such a wonderful opportunity to work with real world data. This project allowed us to learn about excessive research methods.

|  |  |
| --- | --- |
| Topics | Pg. No |
| Section 1:  System Requirement Specifications | Pg 5 |
| Section 2:  Technology Used | Pg 6 |
| Section 3:  Data Provided by the client | Pg 7 – 8 |
| Section 4:  Screenshots of the code and output | Pg 9 - |
| Section 5:  Testing |  |
| Section 6:  Final code |  |

Section 1:

Section 2:

TECHNOLOGY USED: -

**1)Tkinter** is the standard GUI library for **Python**. **Python** when combined with **Tkinter** provides a fast and easy way to create GUI applications. **Tkinter** provides a powerful object-oriented interface to the **Tk** GUI toolkit. Import the **Tkinter** module

2)**PIL.image -Python** Imaging Library (abbreviated as **PIL**) (in newer versions known as Pillow) is a free library for the **Python** programming language that adds support for opening, manipulating, and saving many different image file formats

3) **ImageTk** Module. The **ImageTk** module contains support to create and modify Tkinter BitmapImage and PhotoImage objects from PIL images. ... Options, if any, are passed on to Tkinter. The most commonly used option is foreground, which is used to specify the color for the non-transparent parts

4) **matplotlib**. **pyplot** is a collection of command style **functions** that make **matplotlib** work like MATLAB. Each **pyplot function** makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the **plot** with labels, etc.

5)**pandas** have several important functions in python such as:-

* Handling of data. The **Pandas** library provides a really fast and efficient way to manage and explore data. ...
* Alignment and indexing. ...
* Handling missing data. .
* Cleaning up data. ...
* Input and output tools. ...
* Multiple file formats supported.

6)**re** :- it stands for regular expression and A **regular expression** is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. Regular expressions are widely used in UNIX world. The **Python** module **re** provides full support for Perl-like regular expressions in **Python**.

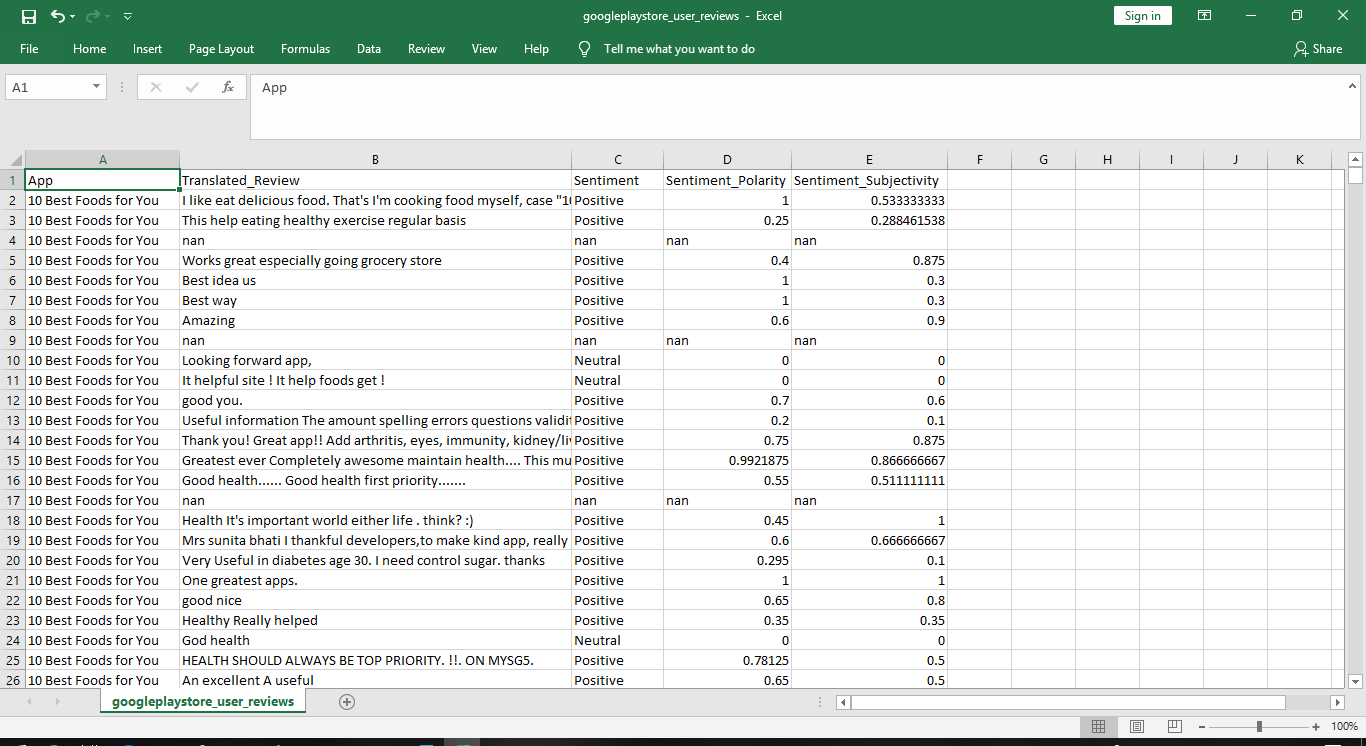
7)NumPy: - **NumPy** is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with **Python**. ... A powerful N-dimensional array object

Section 3:

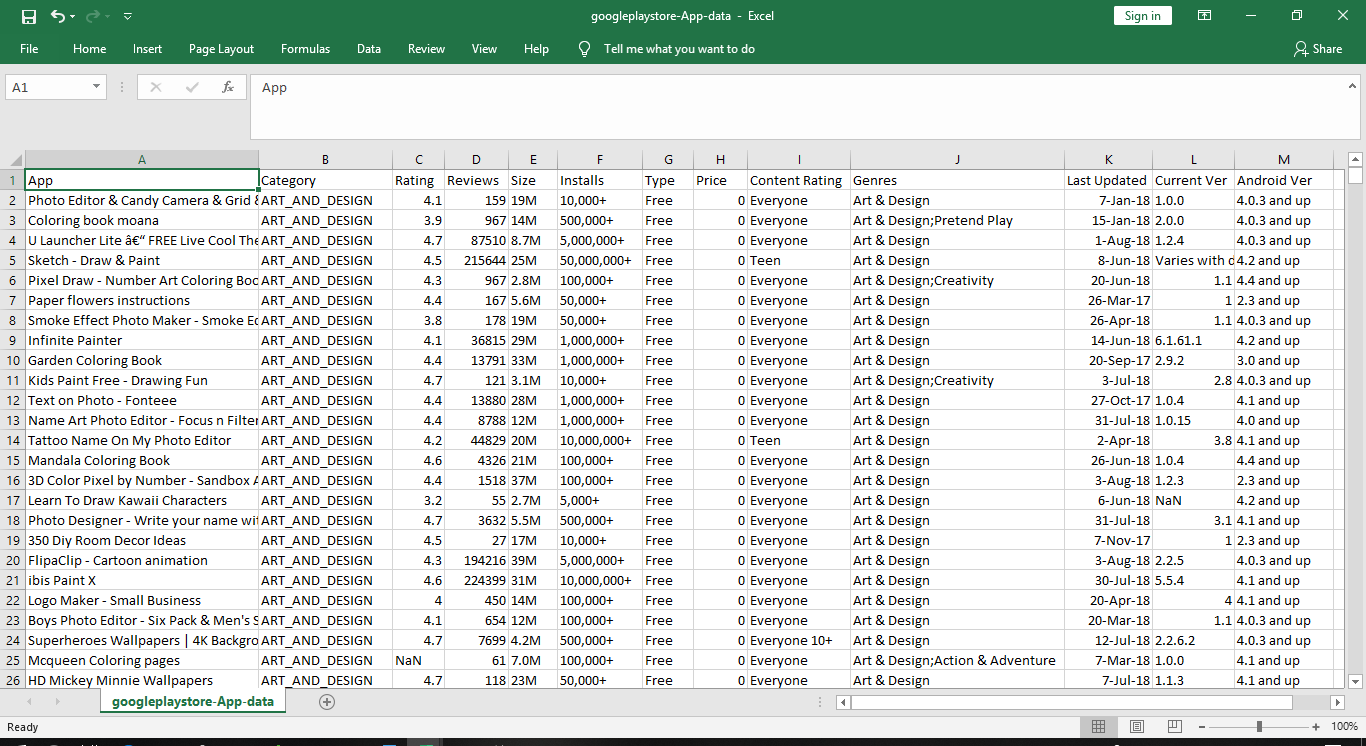
Data provided by the client:

Two csv files were provided by the client as a form of data.

One of them had information about the reviews on apps, the statements given by the audience which was segregated in positive, negative and neutral. There were two more columns named sentiment polarity and sentiment subjectivity.



The second file has complete information on apps like to which category they belong, number of installs, rating, app version, etc.



This dataset has a total of 10842 rows out of which a row (index=10472) had Category missing and each value from a cell was shifted to one cell left. For analysis this row has been dropped out of the data-frame as the Category of the app was unknown.

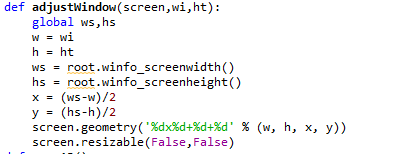
Section 4:

Screenshots:

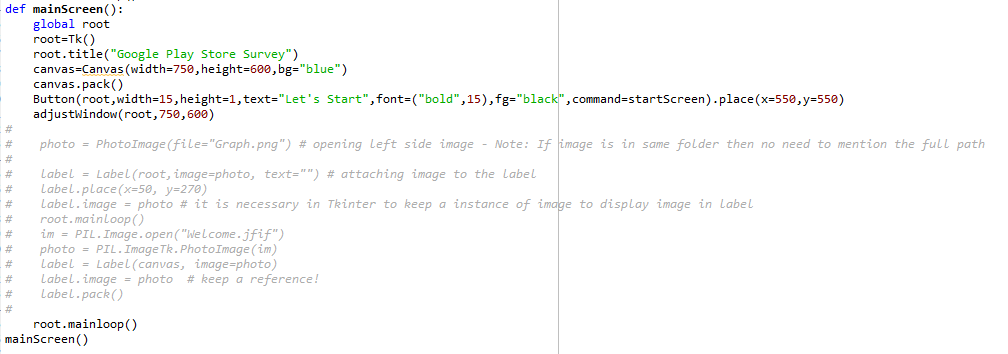
Beginning of the code: importing packages and reading csv files



Designing GUI frame:

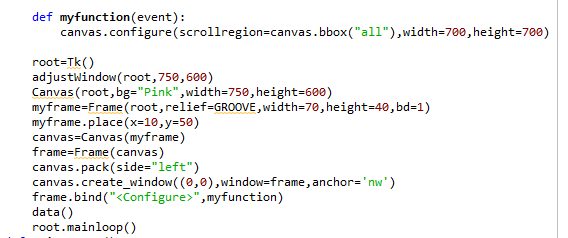


First screen:



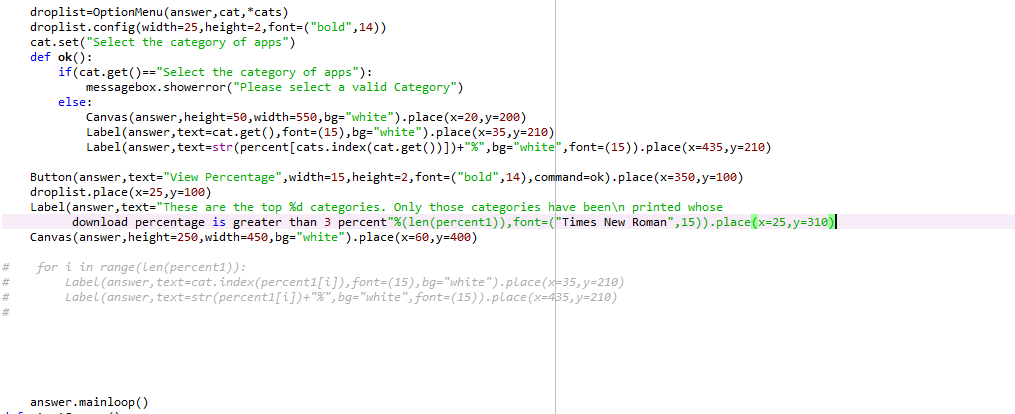
Second screen:



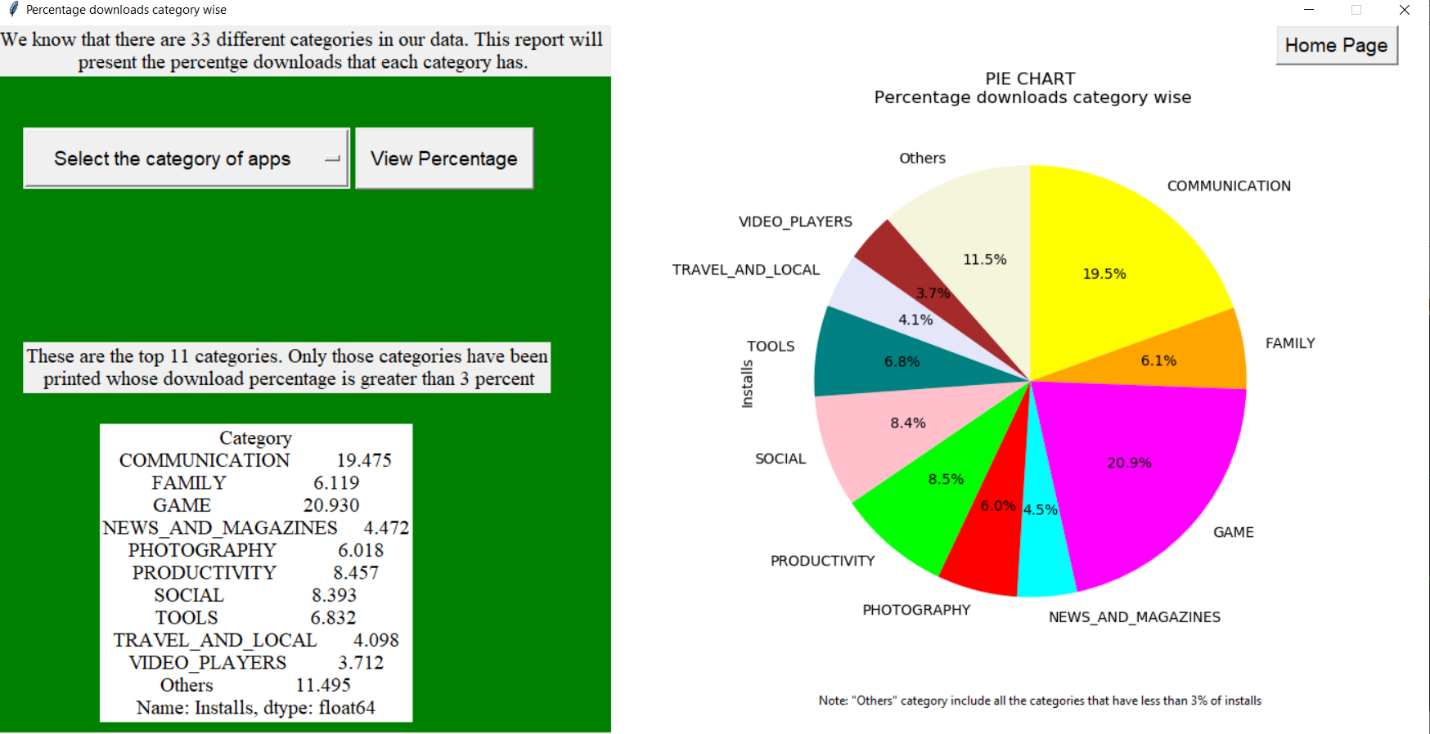


Question 1:





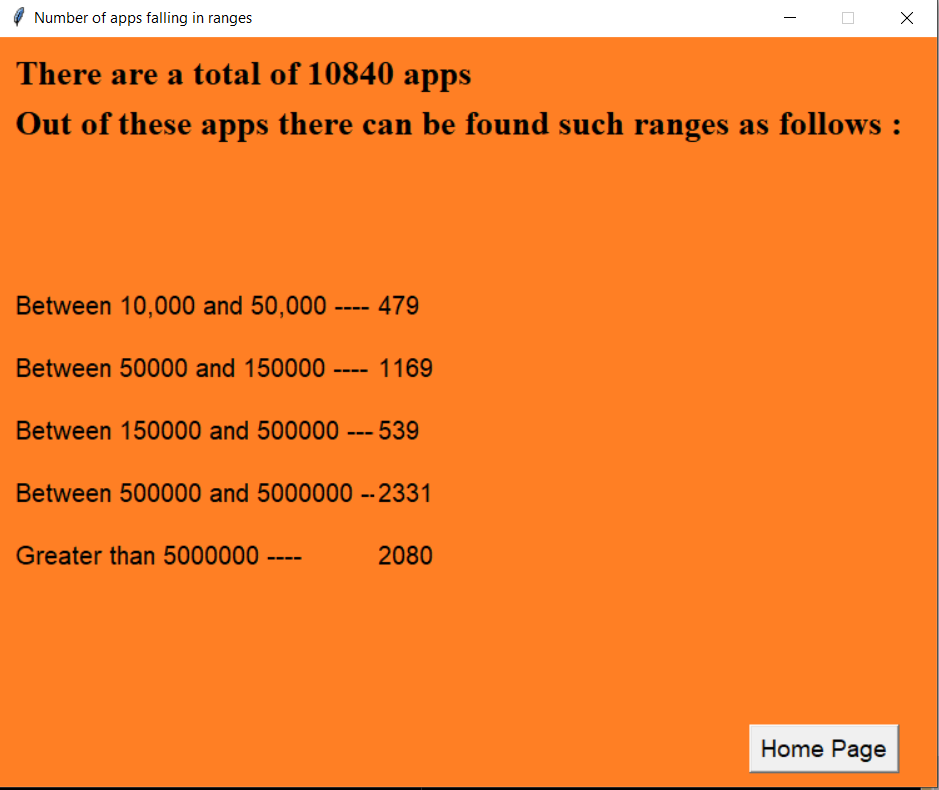
Output:



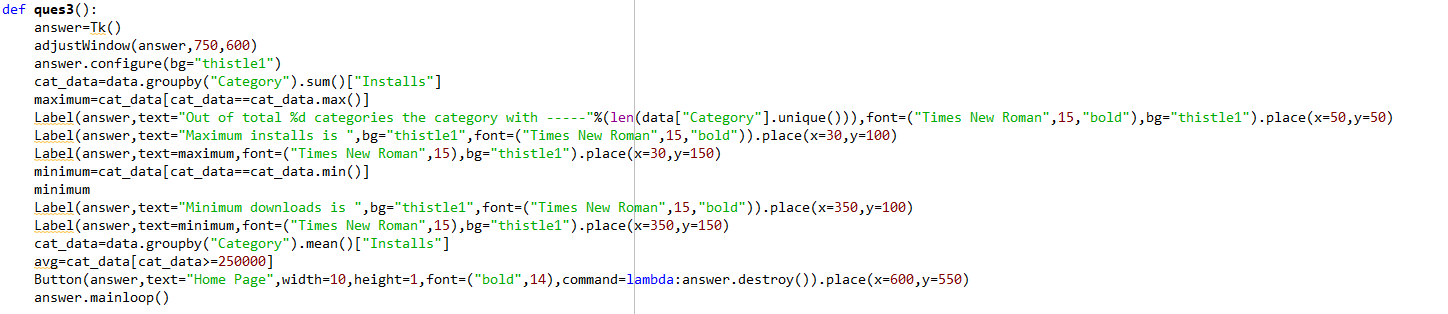
Question 2:



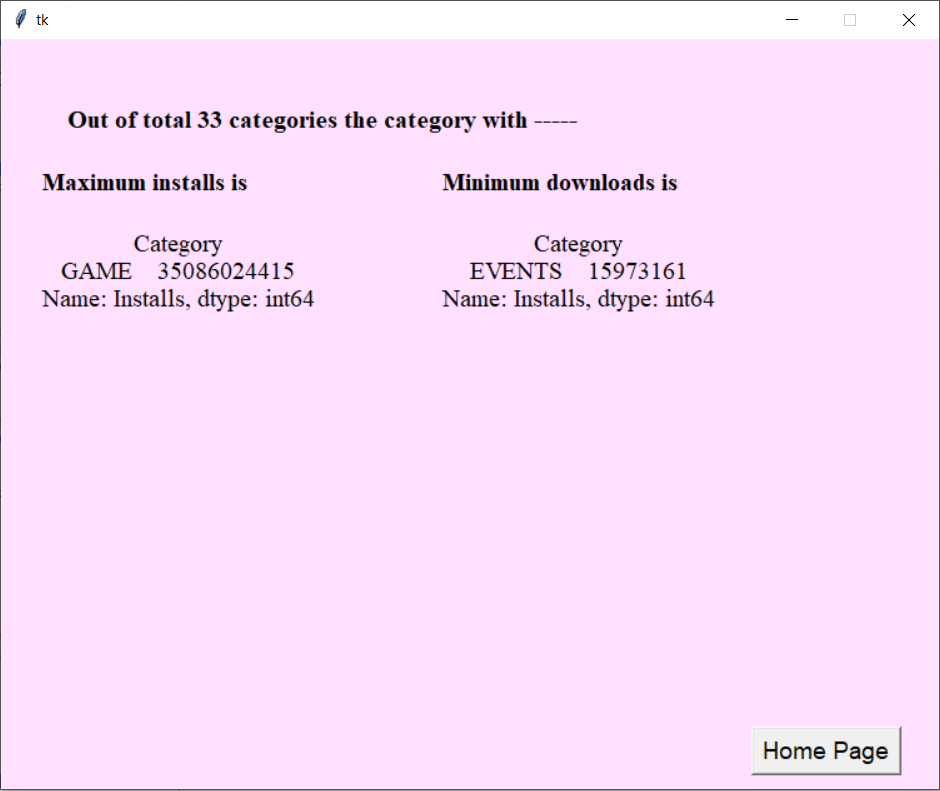
Output:



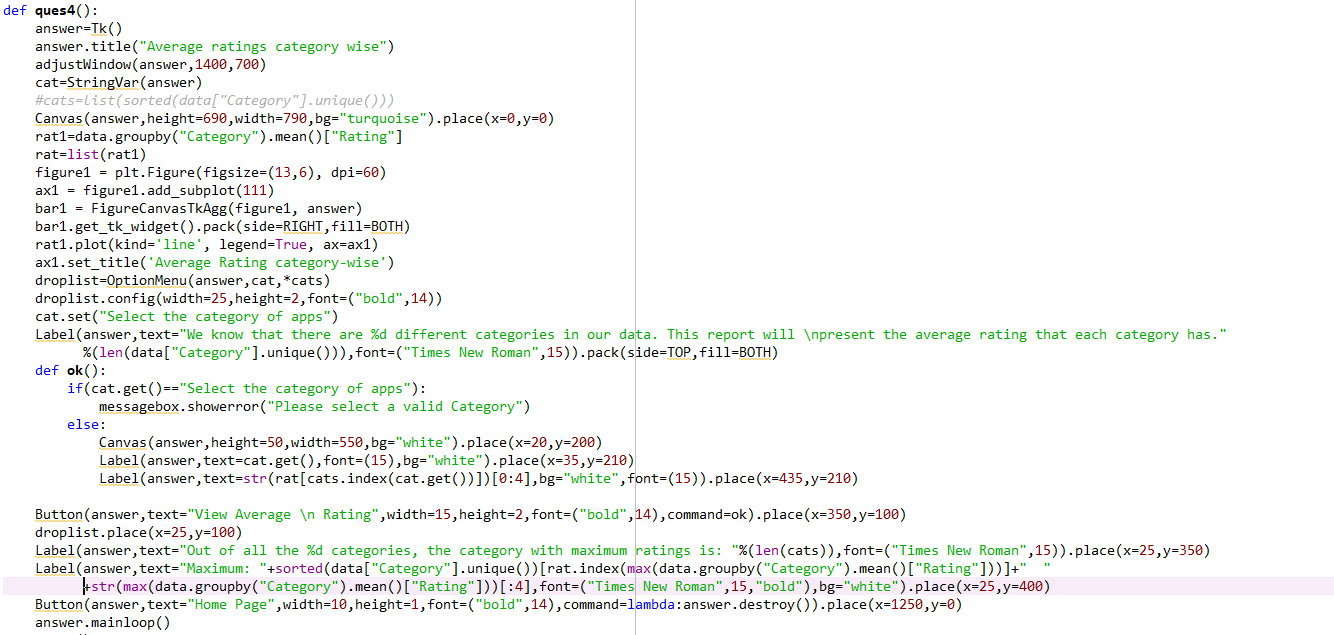
Question 3:

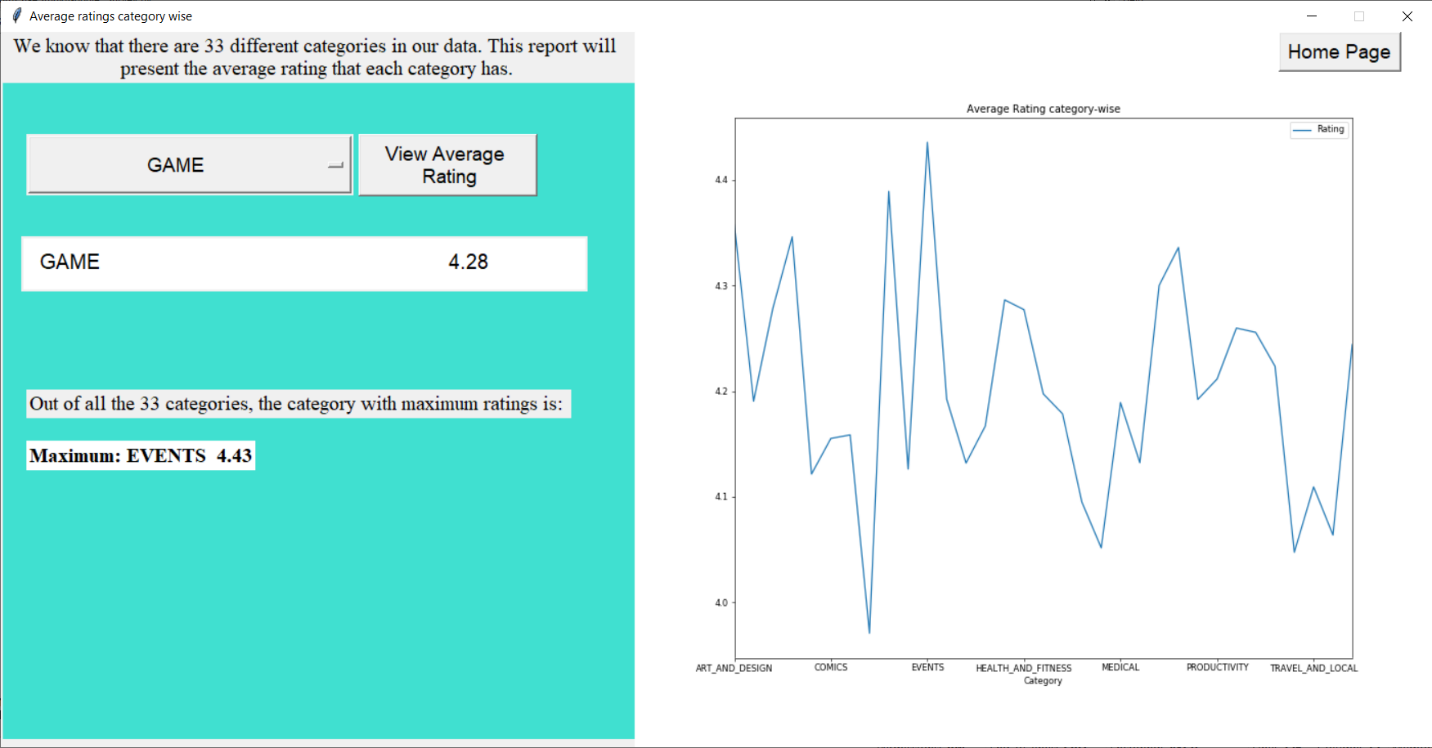


Output:



Question 4:

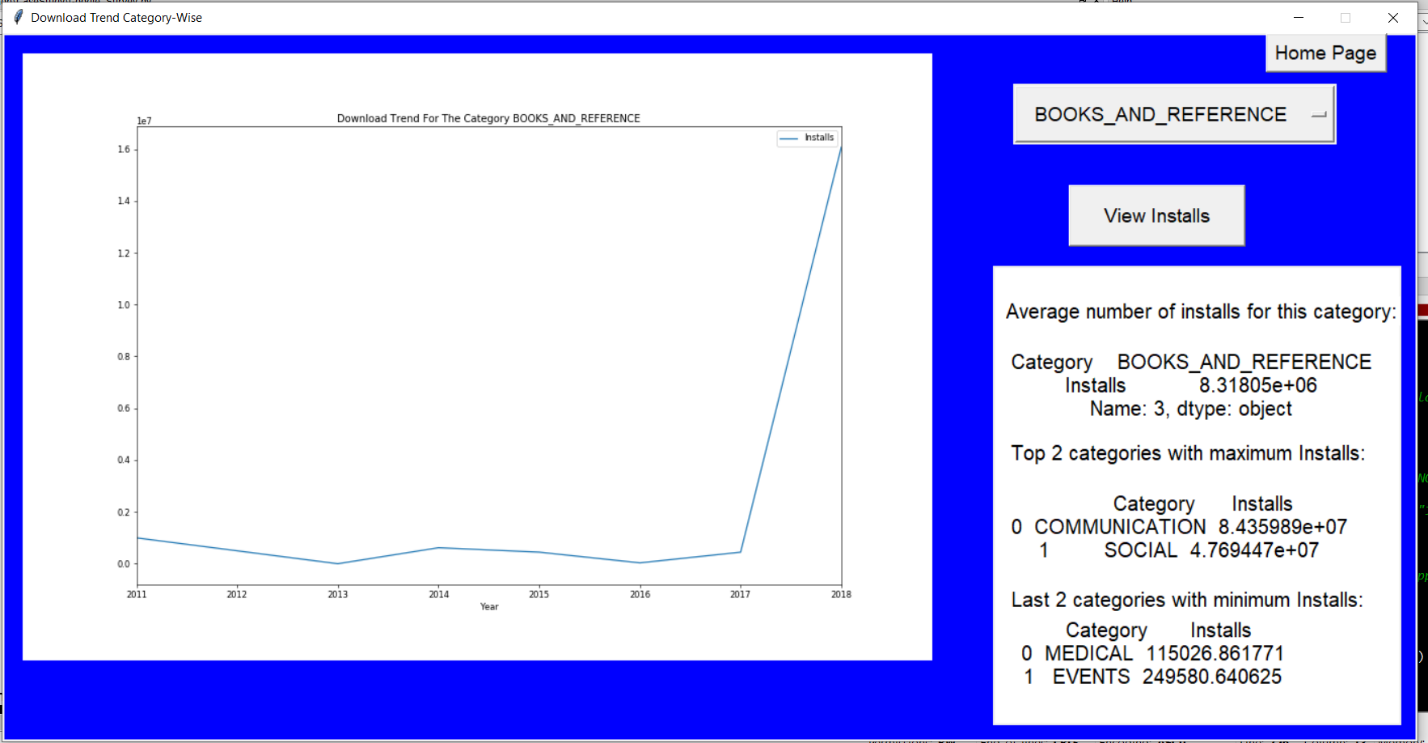




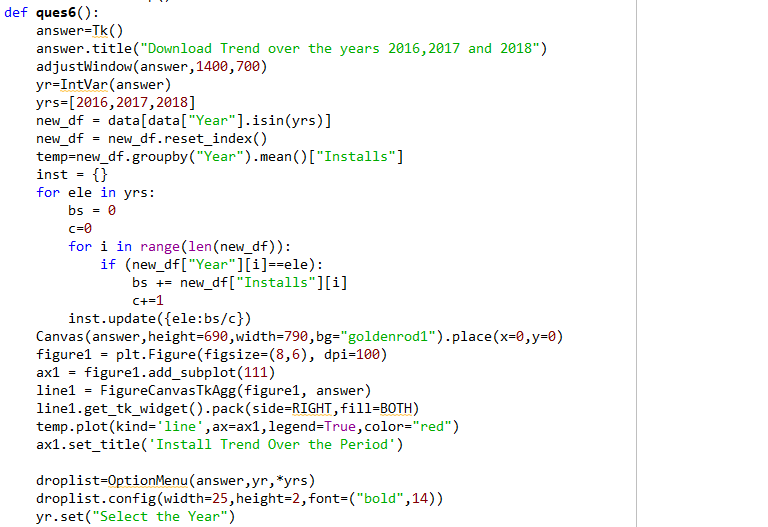
Question 5:



Output:

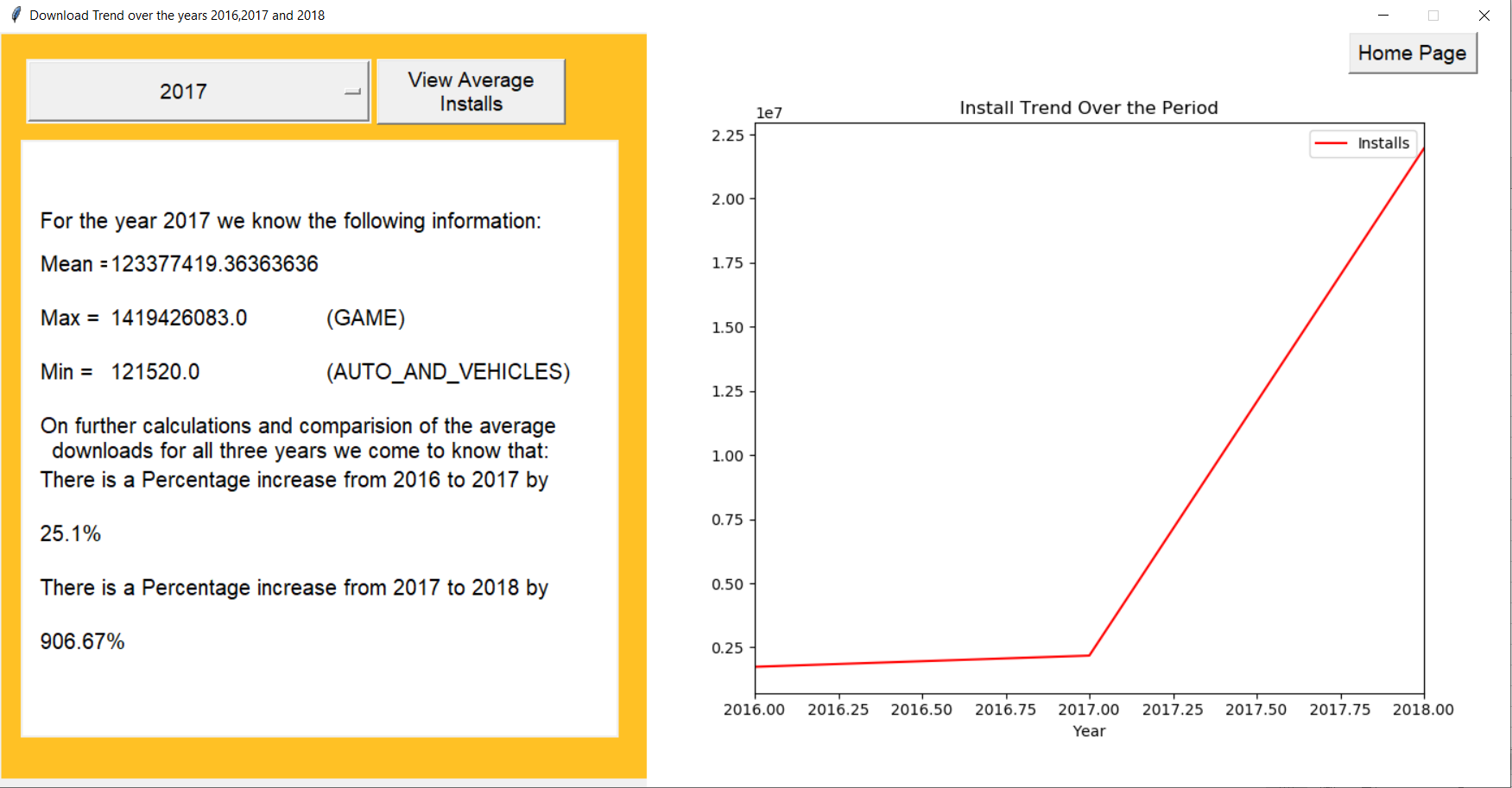


Question 6:

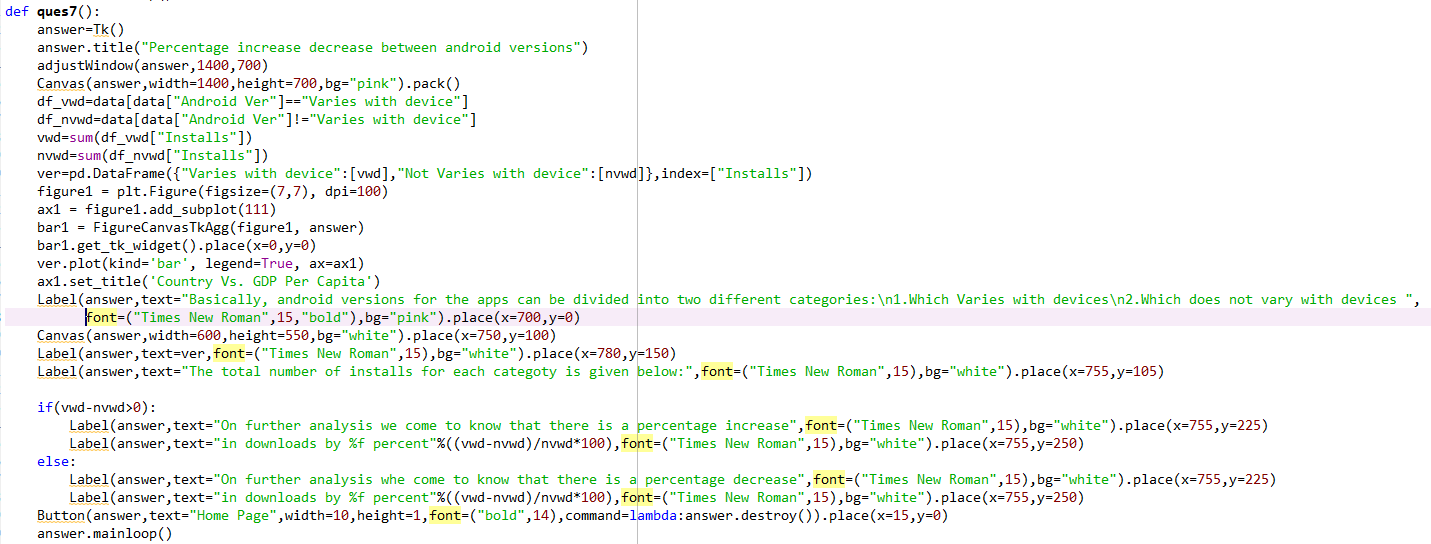




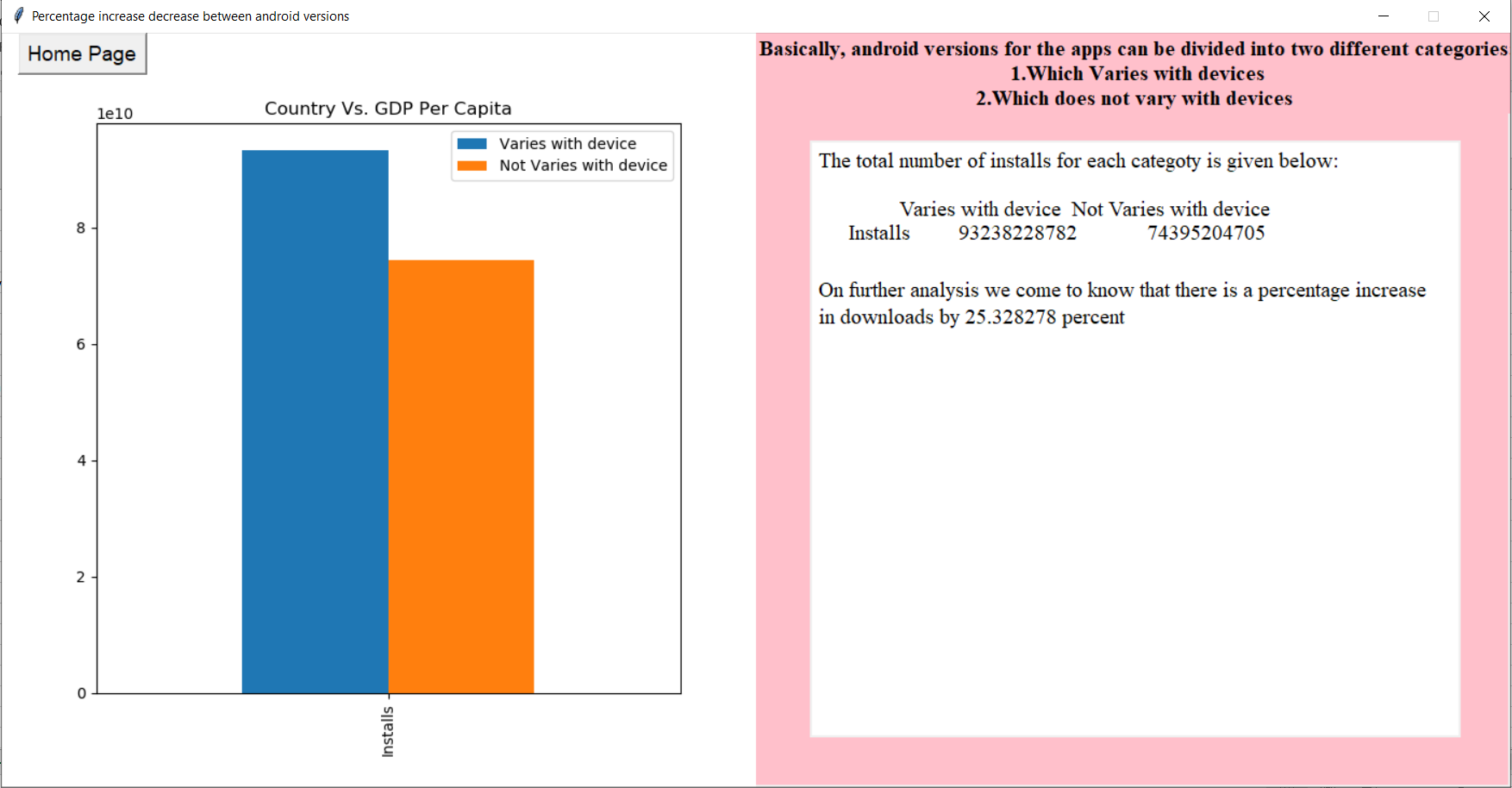
Output:



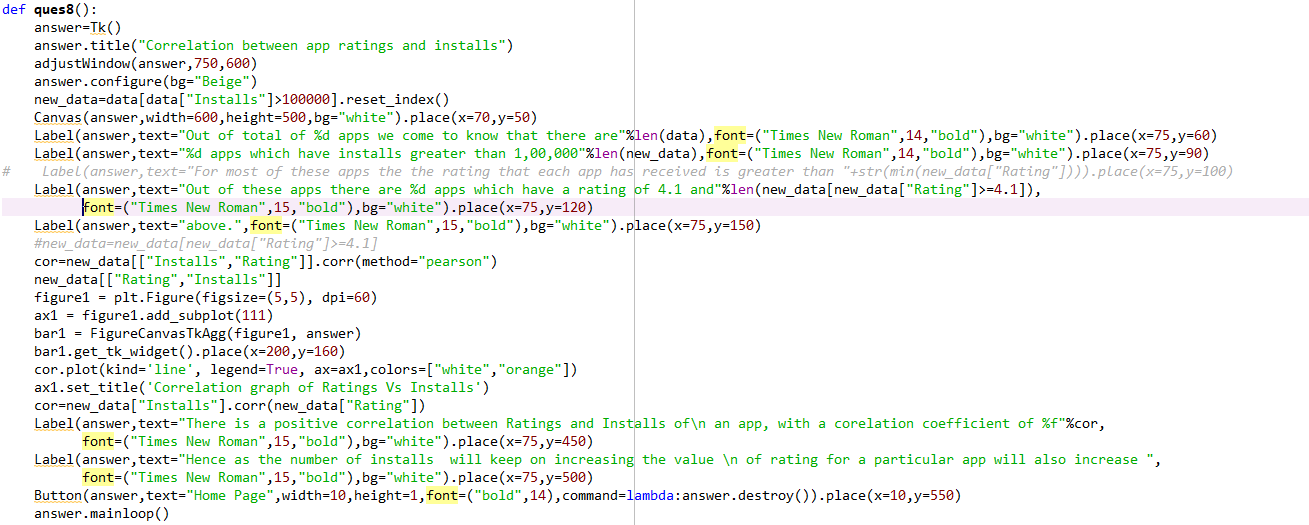
Question 7:



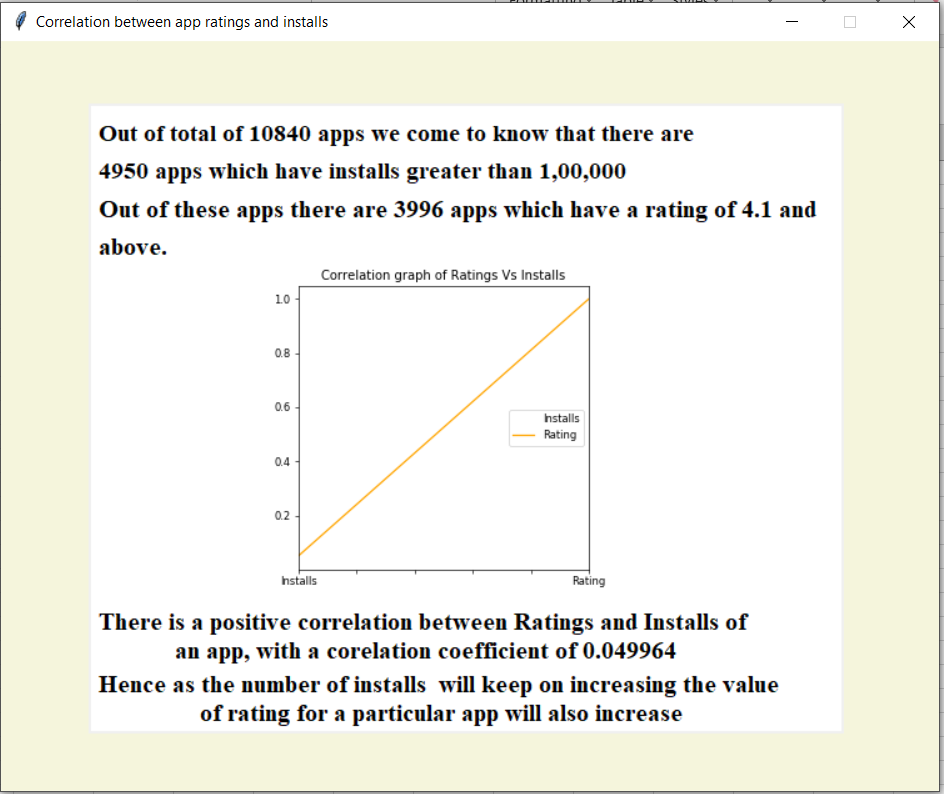
Output:



Question 8:

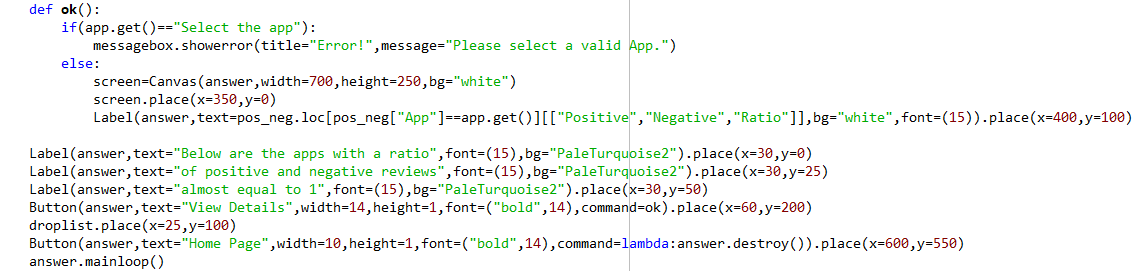


Output:

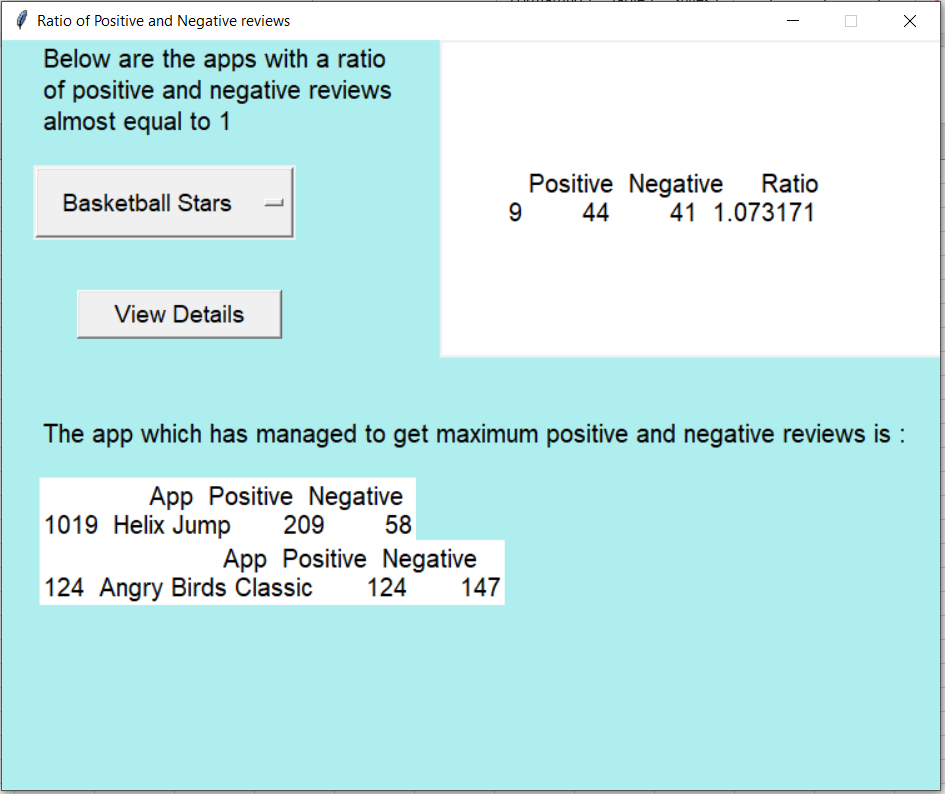


Question 9:





Output:



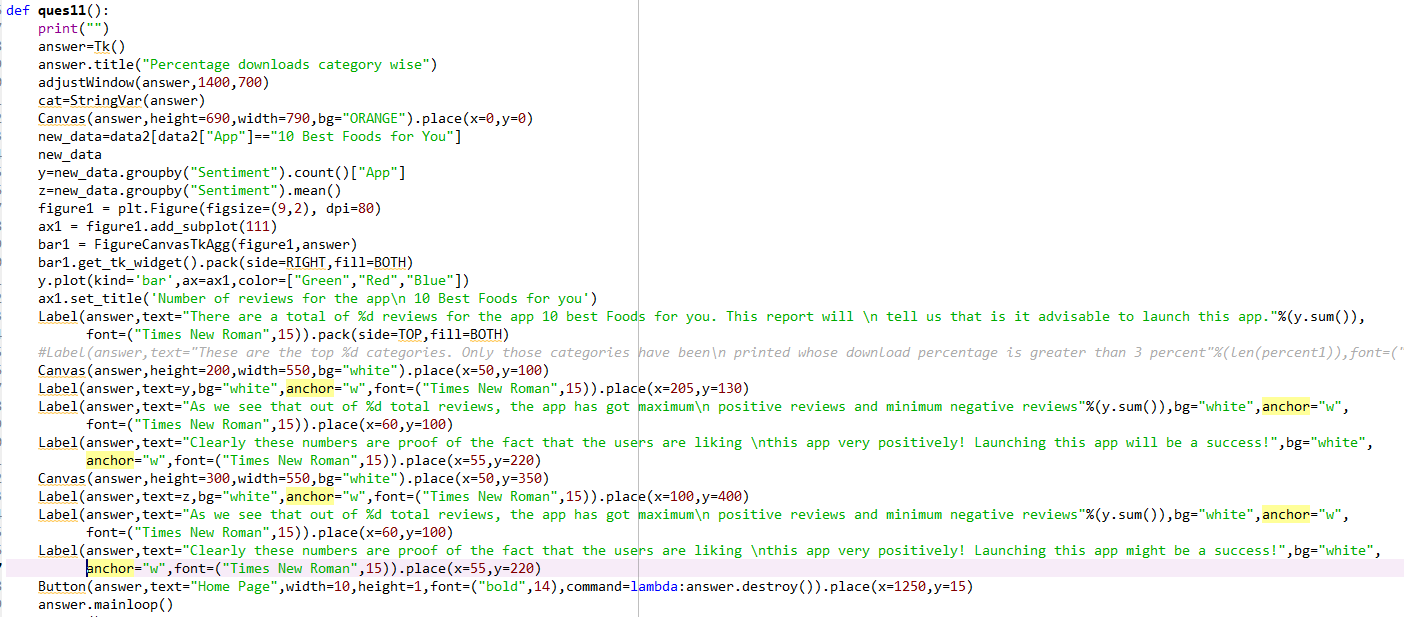
Question 10:



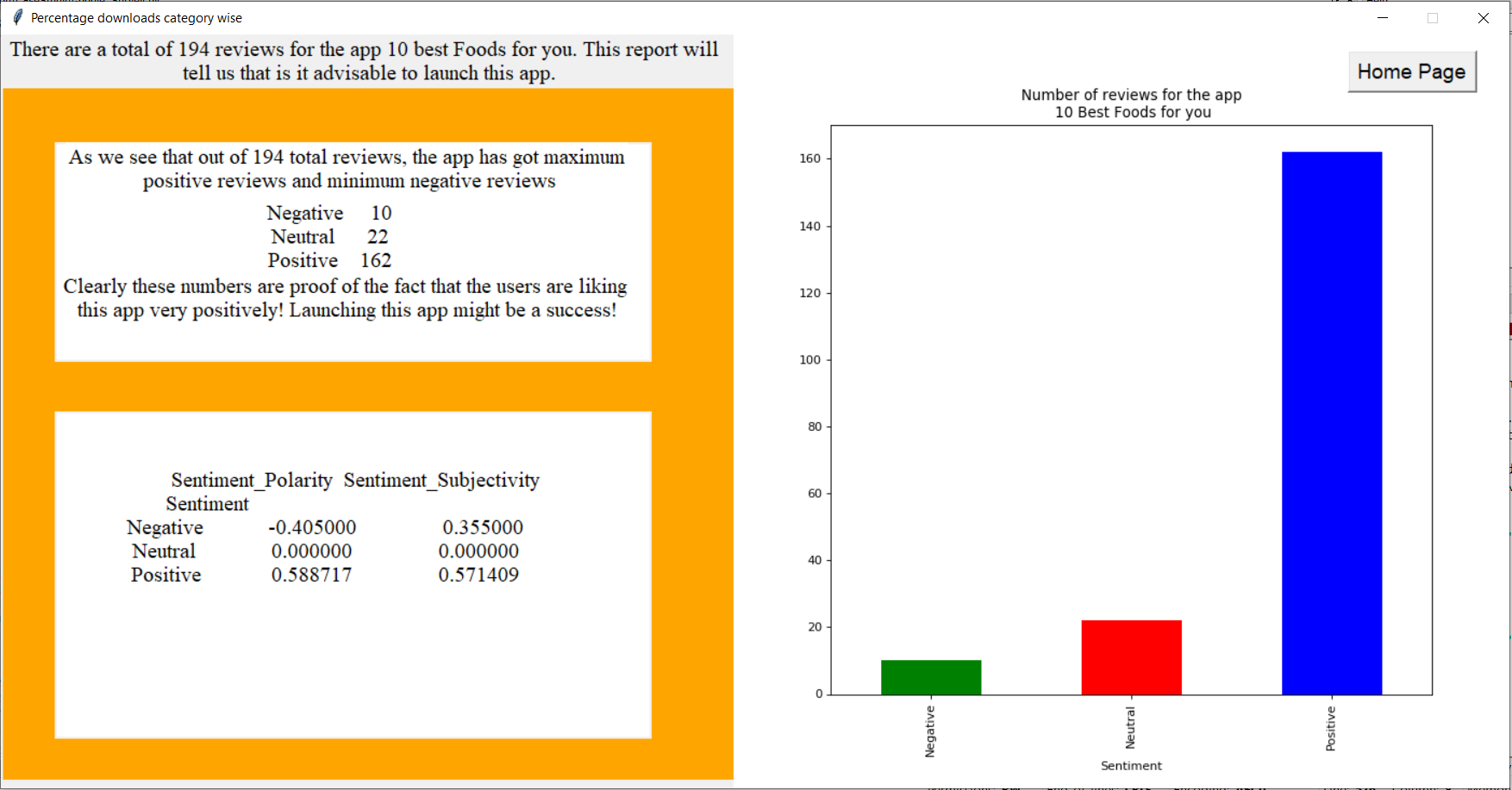


Output:

Question 11:



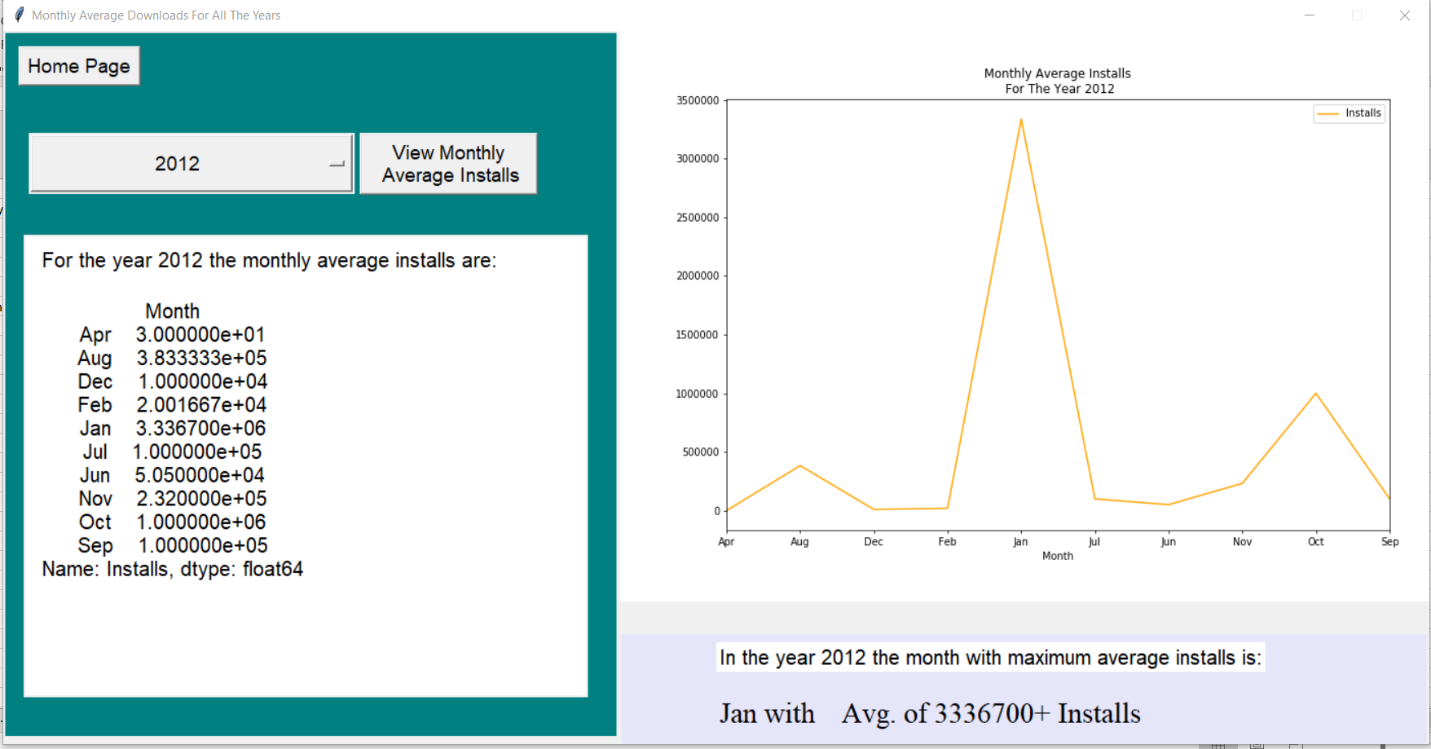
Output:



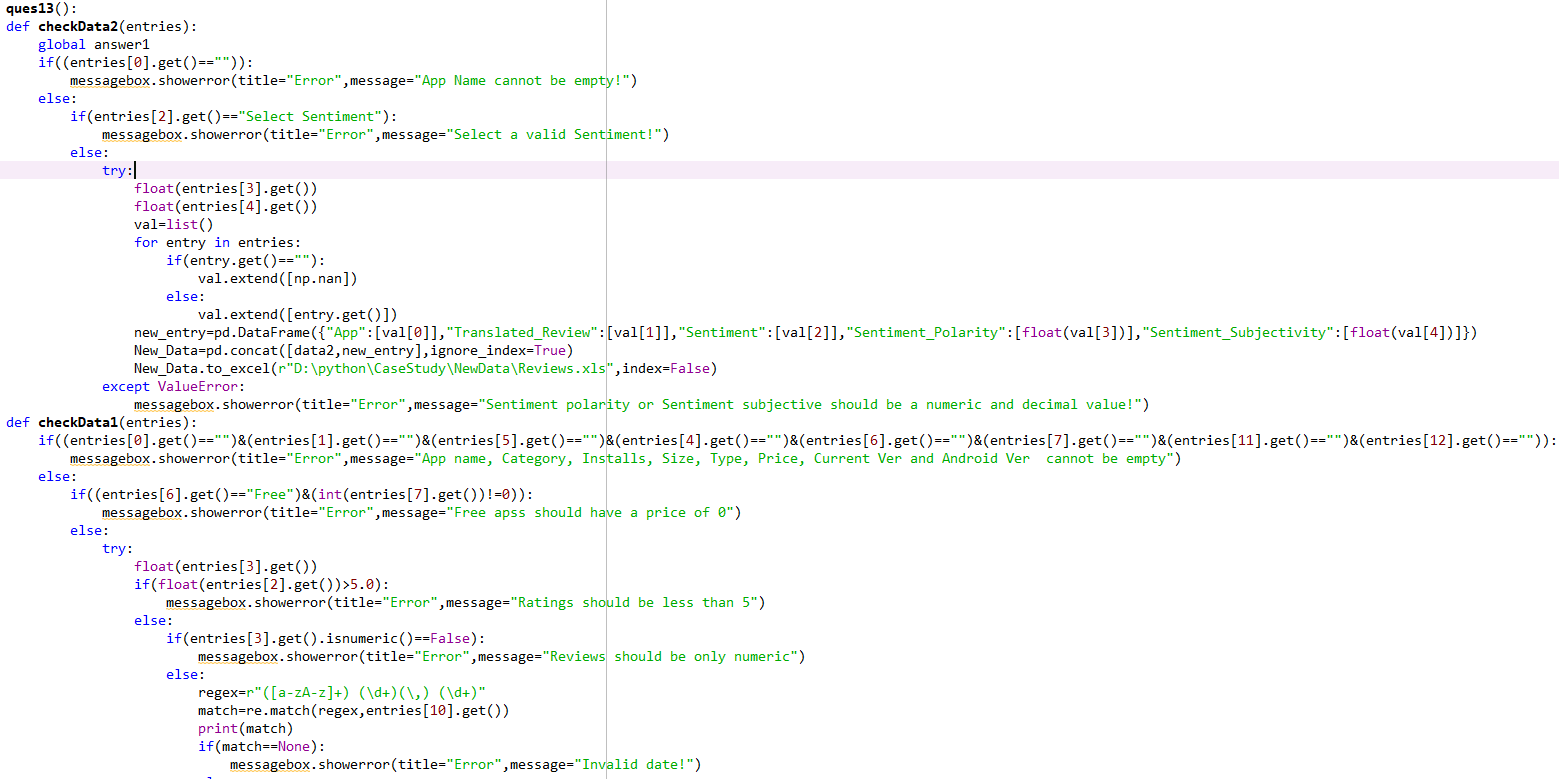
Question 12:

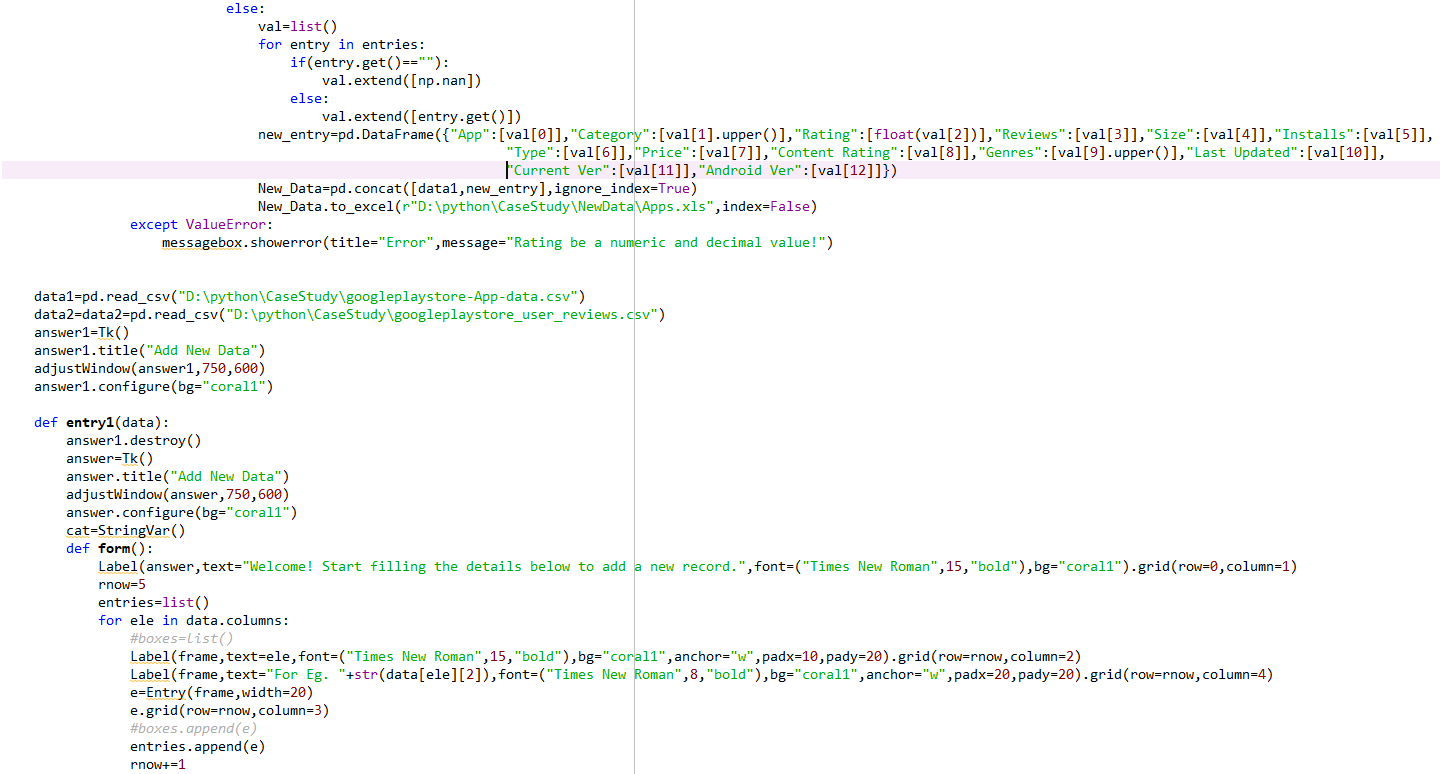


Output:



Question 13:

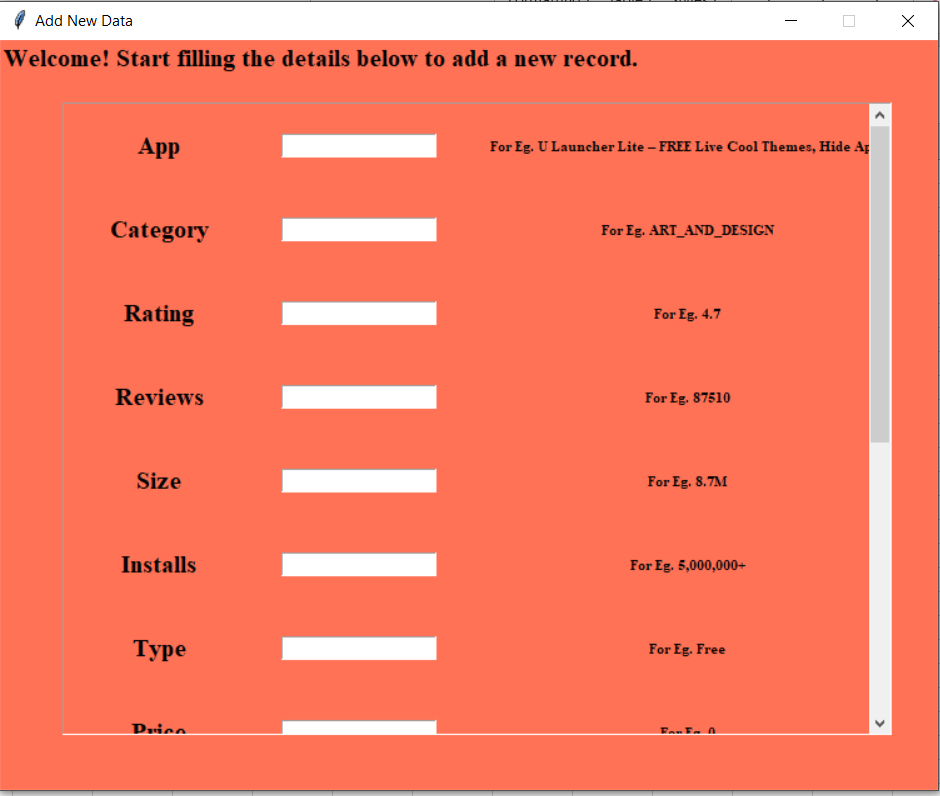


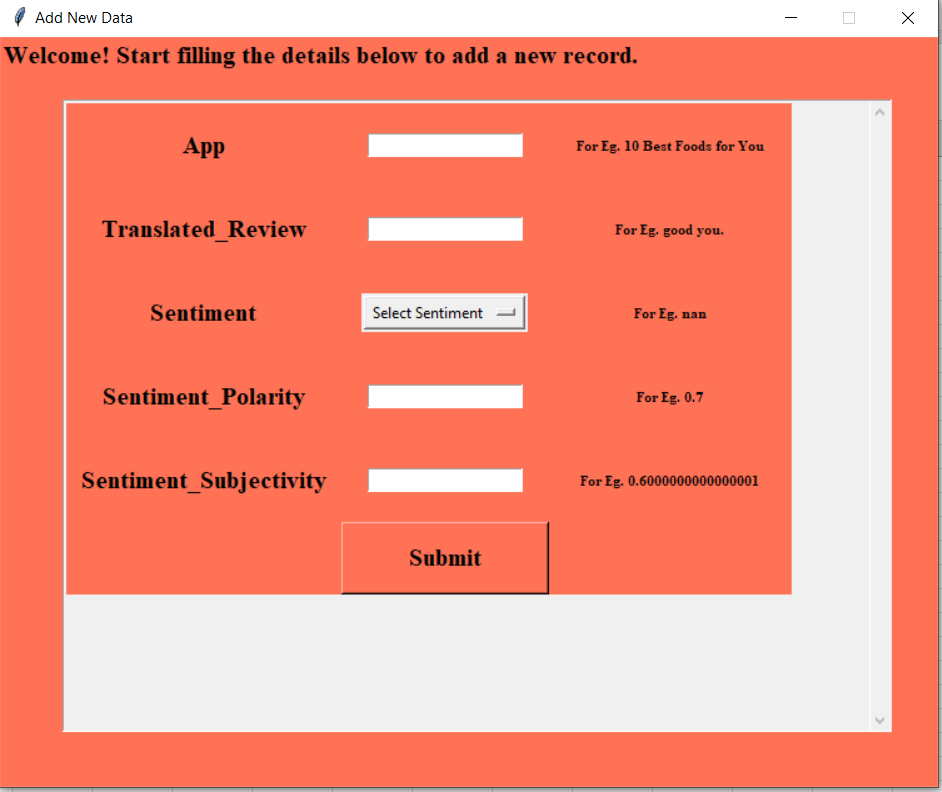






Output:





Section 5:

Section 6:

Final code:

from tkinter import \*

import PIL.Image

import PIL.ImageTk

import matplotlib.pyplot as plt

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg,NavigationToolbar2Tk

import pandas as pd

import re

import pandas as pd

import numpy as np

data=pd.read\_csv("D:\python\CaseStudy\googleplaystore-App-data.csv")

data.head()

num=[]

for entry in data["Installs"]:

num.append(int((re.sub(r"\D","",entry))))

data["Installs"]=num #assumed installs to ne the minimum of every value for eg 10000+ is taken as 10000

data2=pd.read\_csv("D:\python\CaseStudy\googleplaystore\_user\_reviews.csv")

data2.head()

data2.columns

data.drop(index=10472,inplace=True)

data=data.reset\_index()

cats=sorted(data["Category"].unique())

month=[]

year=[]

for i in range(len(data["Last Updated"])):

month.append(re.findall(r"([A-Z][a-z]\*)",data["Last Updated"][i])[0])

year.append(int(re.findall(r"( \d\*)",data["Last Updated"][i])[1].strip()))

data["Month"]=month

data["Year"]=year

data.columns

def adjustWindow(screen,wi,ht):

global ws,hs

w = wi

h = ht

ws = root.winfo\_screenwidth()

hs = root.winfo\_screenheight()

x = (ws-w)/2

y = (hs-h)/2

screen.geometry('%dx%d+%d+%d' % (w, h, x, y))

screen.resizable(False,False)

def ques13():

def checkData2(entries):

global answer1

if((entries[0].get()=="")):

messagebox.showerror(title="Error",message="App Name cannot be empty!")

else:

if(entries[2].get()=="Select Sentiment"):

messagebox.showerror(title="Error",message="Select a valid Sentiment!")

else:

try:

float(entries[3].get())

float(entries[4].get())

val=list()

for entry in entries:

if(entry.get()==""):

val.extend([np.nan])

else:

val.extend([entry.get()])

new\_entry=pd.DataFrame({"App":[val[0]],"Translated\_Review":[val[1]],"Sentiment":[val[2]],"Sentiment\_Polarity":[float(val[3])],"Sentiment\_Subjectivity":[float(val[4])]})

New\_Data=pd.concat([data2,new\_entry],ignore\_index=True)

New\_Data.to\_excel(r"D:\python\CaseStudy\NewData\Reviews.xls",index=False)

except ValueError:

messagebox.showerror(title="Error",message="Sentiment polarity or Sentiment subjective should be a numeric and decimal value!")

def checkData1(entries):

if((entries[0].get()=="")&(entries[1].get()=="")&(entries[5].get()=="")&(entries[4].get()=="")&(entries[6].get()=="")&(entries[7].get()=="")&(entries[11].get()=="")&(entries[12].get()=="")):

messagebox.showerror(title="Error",message="App name, Category, Installs, Size, Type, Price, Current Ver and Android Ver cannot be empty")

else:

if((entries[6].get()=="Free")&(int(entries[7].get())!=0)):

messagebox.showerror(title="Error",message="Free apss should have a price of 0")

else:

try:

float(entries[3].get())

if(float(entries[2].get())>5.0):

messagebox.showerror(title="Error",message="Ratings should be less than 5")

else:

if(entries[3].get().isnumeric()==False):

messagebox.showerror(title="Error",message="Reviews should be only numeric")

else:

regex=r"([a-zA-z]+) (\d+)(\,) (\d+)"

match=re.match(regex,entries[10].get())

print(match)

if(match==None):

messagebox.showerror(title="Error",message="Invalid date!")

else:

val=list()

for entry in entries:

if(entry.get()==""):

val.extend([np.nan])

else:

val.extend([entry.get()])

new\_entry=pd.DataFrame({"App":[val[0]],"Category":[val[1].upper()],"Rating":[float(val[2])],"Reviews":[val[3]],"Size":[val[4]],"Installs":[val[5]],"Type":[val[6]],"Price":[val[7]],"Content Rating":[val[8]],"Genres":[val[9].upper()],"Last Updated":[val[10]],"Current Ver":[val[11]],"Android Ver":[val[12]]})

New\_Data=pd.concat([data1,new\_entry],ignore\_index=True)

New\_Data.to\_excel(r"D:\python\CaseStudy\NewData\Apps.xls",index=False)

except ValueError:

messagebox.showerror(title="Error",message="Rating be a numeric and decimal value!")

data1=pd.read\_csv("D:\python\CaseStudy\googleplaystore-App-data.csv")

data2=data2=pd.read\_csv("D:\python\CaseStudy\googleplaystore\_user\_reviews.csv")

answer1=Tk()

answer1.title("Add New Data")

adjustWindow(answer1,750,600)

answer1.configure(bg="coral1")

def entry1(data):

answer1.destroy()

answer=Tk()

answer.title("Add New Data")

adjustWindow(answer,750,600)

answer.configure(bg="coral1")

cat=StringVar()

def form():

Label(answer,text="Welcome! Start filling the details below to add a new record.",font=("Times New Roman",15,"bold"),bg="coral1").grid(row=0,column=1)

rnow=5

entries=list()

for ele in data.columns:

#boxes=list()

Label(frame,text=ele,font=("Times New Roman",15,"bold"),bg="coral1",anchor="w",padx=10,pady=20).grid(row=rnow,column=2)

Label(frame,text="For Eg. "+str(data[ele][2]),font=("Times New Roman",8,"bold"),bg="coral1",anchor="w",padx=20,pady=20).grid(row=rnow,column=4)

e=Entry(frame,width=20)

e.grid(row=rnow,column=3)

#boxes.append(e)

entries.append(e)

rnow+=1

Button(frame,text="Submit",width=13,font=("Times New Roman",15,"bold"),bg="coral1",pady=10,command=lambda:checkData1(entries)).grid(row=rnow,column=3)

def myfunction(event):

canvas.configure(scrollregion=canvas.bbox("all"),width=640,height=500)

myframe=Frame(answer,relief=GROOVE,width=150,height=100,bd=1)

myframe.place(x=50,y=50)

canvas=Canvas(myframe)

frame=Frame(canvas)

myscrollbar=Scrollbar(myframe,orient="vertical",command=canvas.yview)

canvas.configure(yscrollcommand=myscrollbar.set)

myscrollbar.pack(side="right",fill="y")

canvas.pack(side="left")

canvas.create\_window((0,0),window=frame,anchor='nw')

frame.bind("<Configure>",myfunction)

frame.configure(bg="coral1")

form()

answer.mainloop()

def entry2(data):

answer1.destroy()

answer=Tk()

answer.title("Add New Data")

adjustWindow(answer,750,600)

answer.configure(bg="coral1")

cat=StringVar()

def form2():

Label(answer,text="Welcome! Start filling the details below to add a new record.",font=("Times New Roman",15,"bold"),bg="coral1").grid(row=0,column=1)

rnow=5

entries=list()

for ele in data.columns:

#boxes=list()

if (ele=="Sentiment"):

e=StringVar(answer)

Label(frame,text=ele,font=("Times New Roman",15,"bold"),bg="coral1",anchor="w",padx=10,pady=20).grid(row=rnow,column=2)

Label(frame,text="For Eg. "+str(data[ele][2]),font=("Times New Roman",8,"bold"),bg="coral1",anchor="w",padx=20,pady=20).grid(row=rnow,column=4)

droplist=OptionMenu(frame,e,\*["Positive","Neutral","Negative"])

droplist.grid(row=rnow,column=3)

e.set("Select Sentiment")

else:

Label(frame,text=ele,font=("Times New Roman",15,"bold"),bg="coral1",anchor="w",padx=10,pady=20).grid(row=rnow,column=2)

Label(frame,text="For Eg. "+str(data[ele][10]),font=("Times New Roman",8,"bold"),bg="coral1",anchor="w",padx=20,pady=20).grid(row=rnow,column=4)

e=Entry(frame,width=20)

e.grid(row=rnow,column=3)

#boxes.append(e)

entries.append(e)

rnow+=1

Button(frame,text="Submit",width=13,font=("Times New Roman",15,"bold"),bg="coral1",pady=10,command=lambda:checkData2(entries)).grid(row=rnow,column=3)

def myfunction(event):

canvas.configure(scrollregion=canvas.bbox("all"),width=640,height=500)

myframe=Frame(answer,relief=GROOVE,width=150,height=100,bd=1)

myframe.place(x=50,y=50)

canvas=Canvas(myframe)

frame=Frame(canvas)

myscrollbar=Scrollbar(myframe,orient="vertical",command=canvas.yview)

canvas.configure(yscrollcommand=myscrollbar.set)

myscrollbar.pack(side="right",fill="y")

canvas.pack(side="left")

canvas.create\_window((0,0),window=frame,anchor='nw')

frame.bind("<Configure>",myfunction)

frame.configure(bg="coral1")

form2()

answer.mainloop()

Button(answer1,text="Enter",width=10,height=10,command=lambda : entry1(data1)).pack()

Button(answer1,text="Enter",width=10,height=10,command=lambda : entry2(data2)).pack()

answer1.mainloop()

def ques12():

answer=Tk()

answer.title("Monthly Average Downloads For All The Years ")

adjustWindow(answer,1400,700)

month=[]

year=[]

yr=IntVar(answer)

Canvas(answer,height=690,width=600,bg="Teal").place(x=0,y=0)

Canvas(answer,height=150,width=790,bg="Lavender").place(x=605,y=590)

# for i in range(len(data["Last Updated"])):

# month.append(re.findall(r"([A-Z][a-z][a-z])",data["Last Updated"][i])[0])

# year.append(int(re.findall(r"( \d\*)",data["Last Updated"][i])[1].strip()))

# data["Month"]=month

# data["Year"]=year

droplist=OptionMenu(answer,yr,\*sorted(data["Year"].unique()))

droplist.config(width=25,height=2,font=("bold",14))

yr.set("Select the year")

def ok():

Canvas(answer,height=450,width=550,bg="white").place(x=20,y=200)

Label(answer,text="For the year %d the monthly average installs are:"%yr.get(),font=(15),bg="white").place(x=35,y=210)

new\_df=data[data["Year"]==yr.get()]

installs=new\_df.groupby("Month").mean()["Installs"]

figure1 = plt.Figure(figsize=(12,8), dpi=70)

ax1 = figure1.add\_subplot(111)

line1 = FigureCanvasTkAgg(figure1,answer)

line1.get\_tk\_widget().place(x=605,y=0)

installs.plot(kind='line', legend=True, ax=ax1,color="Orange")

ax1.set\_title('Monthly Average Installs\n For The Year %d'%yr.get())

Label(answer,text=installs,bg="white",font=(15),anchor="w").place(x=35,y=260)

Canvas(answer,height=150,width=790,bg="Lavender").place(x=605,y=590)

Label(answer,text="In the year %d the month with maximum average installs is:"%yr.get(),bg="white",font=(15),anchor="w").place(x=700,y=600)

installs=list(new\_df.groupby("Month").mean()["Installs"])

Label(answer,text=sorted(new\_df["Month"].unique())[installs.index(max(new\_df.groupby("Month").mean()["Installs"]))]+" with " ,font=("Times New Roman",22),bg="Lavender").place(x=700,y=650)

Label(answer,text="Avg. of %d+ Installs"%max(new\_df.groupby("Month").mean()["Installs"]),font=("Times New Roman",22),bg="Lavender").place(x=820,y=650)

Button(answer,text="View Monthly\n Average Installs",width=15,height=2,font=("bold",14),command=ok).place(x=350,y=100)

droplist.place(x=25,y=100)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=15,y=15)

answer.mainloop()

def ques11():

print("")

answer=Tk()

answer.title("Percentage downloads category wise")

adjustWindow(answer,1400,700)

cat=StringVar(answer)

Canvas(answer,height=690,width=790,bg="ORANGE").place(x=0,y=0)

new\_data=data2[data2["App"]=="10 Best Foods for You"]

new\_data

y=new\_data.groupby("Sentiment").count()["App"]

z=new\_data.groupby("Sentiment").mean()

figure1 = plt.Figure(figsize=(9,2), dpi=80)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1,answer)

bar1.get\_tk\_widget().pack(side=RIGHT,fill=BOTH)

y.plot(kind='bar',ax=ax1,color=["Green","Red","Blue"])

ax1.set\_title('Number of reviews for the app\n 10 Best Foods for you')

Label(answer,text="There are a total of %d reviews for the app 10 best Foods for you. This report will \n tell us that is it advisable to launch this app."%(y.sum()),font=("Times New Roman",15)).pack(side=TOP,fill=BOTH)

#Label(answer,text="These are the top %d categories. Only those categories have been\n printed whose download percentage is greater than 3 percent"%(len(percent1)),font=("Times New Roman",15)).place(x=25,y=310)

Canvas(answer,height=200,width=550,bg="white").place(x=50,y=100)

Label(answer,text=y,bg="white",anchor="w",font=("Times New Roman",15)).place(x=205,y=130)

Label(answer,text="As we see that out of %d total reviews, the app has got maximum\n positive reviews and minimum negative reviews"%(y.sum()),bg="white",anchor="w",font=("Times New Roman",15)).place(x=60,y=100)

Label(answer,text="Clearly these numbers are proof of the fact that the users are liking \nthis app very positively! Launching this app will be a success!",bg="white",anchor="w",font=("Times New Roman",15)).place(x=55,y=220)

Canvas(answer,height=300,width=550,bg="white").place(x=50,y=350)

Label(answer,text=z,bg="white",anchor="w",font=("Times New Roman",15)).place(x=100,y=400)

Label(answer,text="As we see that out of %d total reviews, the app has got maximum\n positive reviews and minimum negative reviews"%(y.sum()),bg="white",anchor="w",font=("Times New Roman",15)).place(x=60,y=100)

Label(answer,text="Clearly these numbers are proof of the fact that the users are liking \nthis app very positively! Launching this app might be a success!",bg="white",anchor="w",font=("Times New Roman",15)).place(x=55,y=220)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=1250,y=15)

answer.mainloop()

def ques10():

def myfunction(event):

canvas1.configure(scrollregion=canvas1.bbox("all"),width=700,height=300)

def data(rev):

c=1

i=0

while(i!=len(data2)):

if((data2["App"][i]==e1.get())&(data2["Sentiment"][i]==rev)):

#print(data2["Translated\_Review"][i])

Label(frame,text=str(c),anchor="w").pack(side="top",fill="x")

c+=1

Label(frame,text=rev,anchor="w",font=("Times New Roman",14)).pack(side="top",fill="x")

Label(frame,text=data2["Translated\_Review"][i]+"\n",anchor="w",font=("Times New Roman",14)).pack(side="top",fill="x")

i+=1

answer=Tk()

adjustWindow(answer,800,600)

answer.configure(bg="Brown")

myframe=Frame(answer,relief=GROOVE,width=700,height=300,bd=2)

myframe.place(x=50,y=200)

canvas1=Canvas(myframe)

frame=Frame(canvas1)

myscrollbar2=Scrollbar(myframe,orient="vertical",command=canvas1.yview)

canvas1.configure(yscrollcommand=myscrollbar2.set)

myscrollbar2.pack(side="right",fill="y")

myscrollbar21=Scrollbar(myframe,orient="horizontal",command=canvas1.xview)

canvas1.configure(xscrollcommand=myscrollbar21.set)

myscrollbar21.pack(side="bottom",fill="x")

canvas1.pack()

canvas1.create\_window((0,0),window=frame,anchor='nw')

frame.bind("<Configure>",myfunction)

# data2=pd.read\_csv("D:\python\CaseStudy\googleplaystore\_user\_reviews.csv")

# data2.head()

# data2["App"].unique()

Label(answer,text="Enter the app you want to see: ",font=("Times New Roman",20,"bold"),bg="brown").place(x=100,y=50)

e1=Entry(answer,width=20)

e1.place(x=500,y=60)

Label(answer,text="Which type of reviews \n do you want to see",font=("Times New Roman",15,"bold"),bg="brown").place(x=100,y=130)

Button(answer,text="Positive",font=("Times New Roman",15,"bold"),command=lambda: data("Positive")).place(x=350,y=140)

Button(answer,text="Neutral",font=("Times New Roman",15,"bold"),command=lambda: data("Neutral")).place(x=450,y=140)

Button(answer,text="Negative",font=("Times New Roman",15,"bold"),command=lambda: data("Negative")).place(x=550,y=140)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=10,y=550)

answer.mainloop()

def ques9():

answer=Tk()

app=StringVar(answer)

answer.title("Ratio of Positive and Negative reviews")

adjustWindow(answer,750,600)

answer.configure(bg="PaleTurquoise2")

pos=[]

neg=[]

for i in range(len(data2)):

p=0

n=0

if(data2["Sentiment"][i]=="Positive"):

p+=1

elif(data2["Sentiment"][i]=="Negative"):

n+=1

pos.append(p)

neg.append(n)

data2["Positive"]=pos

data2["Negative"]=neg

# data2[["Positive","Negative"]]

pos\_neg=data2.groupby("App")[["Positive","Negative"]].sum()

# Label(answer,text=pos\_neg.max(),font=(15),bg="white").place(x=300,y=150)

pos\_neg["Ratio"]=pos\_neg["Positive"]/pos\_neg["Negative"]

pos\_neg=pos\_neg[pos\_neg["Ratio"]<=1.1]

pos\_neg=pos\_neg[pos\_neg["Ratio"]>0.9].reset\_index()

apps=sorted(pos\_neg["App"].unique())

droplist=OptionMenu(answer,app,\*apps)

droplist.config(width=15,height=2,font=("bold",14))

app.set("Select the app")

pos\_neg1=data2.groupby("App")[["Positive","Negative"]].sum().reset\_index()

new\_pos=pos\_neg1[pos\_neg1["Positive"]==pos\_neg1["Positive"].max()]

new\_neg=pos\_neg1[pos\_neg1["Negative"]==pos\_neg1["Negative"].max()]

Label(Label(answer,text=new\_pos,font=(15),bg="white").place(x=30,y=350))

Label(Label(answer,text=new\_neg,font=(15),bg="white").place(x=30,y=400))

#Label(answer,text=pos\_neg1[["Positive","Negative"]].max(),font=(15),bg="white").place(x=50,y=350)

Label(answer,text="The app which has managed to get maximum positive and negative reviews is :",bg="PaleTurquoise2",font=(15)).place(x=30,y=300)

def ok():

if(app.get()=="Select the app"):

messagebox.showerror(title="Error!",message="Please select a valid App.")

else:

screen=Canvas(answer,width=700,height=250,bg="white")

screen.place(x=350,y=0)

Label(answer,text=pos\_neg.loc[pos\_neg["App"]==app.get()][["Positive","Negative","Ratio"]],bg="white",font=(15)).place(x=400,y=100)

Label(answer,text="Below are the apps with a ratio",font=(15),bg="PaleTurquoise2").place(x=30,y=0)

Label(answer,text="of positive and negative reviews",font=(15),bg="PaleTurquoise2").place(x=30,y=25)

Label(answer,text="almost equal to 1",font=(15),bg="PaleTurquoise2").place(x=30,y=50)

Button(answer,text="View Details",width=14,height=1,font=("bold",14),command=ok).place(x=60,y=200)

droplist.place(x=25,y=100)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=600,y=550)

answer.mainloop()

def ques8():

answer=Tk()

answer.title("Correlation between app ratings and installs")

adjustWindow(answer,750,600)

answer.configure(bg="Beige")

new\_data=data[data["Installs"]>100000].reset\_index()

Canvas(answer,width=600,height=500,bg="white").place(x=70,y=50)

Label(answer,text="Out of total of %d apps we come to know that there are"%len(data),font=("Times New Roman",14,"bold"),bg="white").place(x=75,y=60)

Label(answer,text="%d apps which have installs greater than 1,00,000"%len(new\_data),font=("Times New Roman",14,"bold"),bg="white").place(x=75,y=90)

# Label(answer,text="For most of these apps the rating that each app has received is greater than "+str(min(new\_data["Rating"]))).place(x=75,y=100)

Label(answer,text="Out of these apps there are %d apps which have a rating of 4.1 and"%len(new\_data[new\_data["Rating"]>=4.1]),font=("Times New Roman",15,"bold"),bg="white").place(x=75,y=120)

Label(answer,text="above.",font=("Times New Roman",15,"bold"),bg="white").place(x=75,y=150)

#new\_data=new\_data[new\_data["Rating"]>=4.1]

cor=new\_data[["Installs","Rating"]].corr(method="pearson")

new\_data[["Rating","Installs"]]

figure1 = plt.Figure(figsize=(5,5), dpi=60)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, answer)

bar1.get\_tk\_widget().place(x=200,y=160)

cor.plot(kind='line', legend=True, ax=ax1,colors=["white","orange"])

ax1.set\_title('Correlation graph of Ratings Vs Installs')

cor=new\_data["Installs"].corr(new\_data["Rating"])

Label(answer,text="There is a positive correlation between Ratings and Installs of\n an app, with a corelation coefficient of %f"%cor,font=("Times New Roman",15,"bold"),bg="white").place(x=75,y=450)

Label(answer,text="Hence as the number of installs will keep on increasing the value \n of rating for a particular app will also increase ",font=("Times New Roman",15,"bold"),bg="white").place(x=75,y=500)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=10,y=550)

answer.mainloop()

def ques7():

answer=Tk()

answer.title("Percentage increase decrease between android versions")

adjustWindow(answer,1400,700)

Canvas(answer,width=1400,height=700,bg="pink").pack()

df\_vwd=data[data["Android Ver"]=="Varies with device"]

df\_nvwd=data[data["Android Ver"]!="Varies with device"]

vwd=sum(df\_vwd["Installs"])

nvwd=sum(df\_nvwd["Installs"])

ver=pd.DataFrame({"Varies with device":[vwd],"Not Varies with device":[nvwd]},index=["Installs"])

figure1 = plt.Figure(figsize=(7,7), dpi=100)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, answer)

bar1.get\_tk\_widget().place(x=0,y=0)

ver.plot(kind='bar', legend=True, ax=ax1)

ax1.set\_title('Country Vs. GDP Per Capita')

Label(answer,text="Basically, android versions for the apps can be divided into two different categories:\n1.Which Varies with devices\n2.Which does not vary with devices ",font=("Times New Roman",15,"bold"),bg="pink").place(x=700,y=0)

Canvas(answer,width=600,height=550,bg="white").place(x=750,y=100)

Label(answer,text=ver,font=("Times New Roman",15),bg="white").place(x=780,y=150)

Label(answer,text="The total number of installs for each categoty is given below:",font=("Times New Roman",15),bg="white").place(x=755,y=105)

if(vwd-nvwd>0):

Label(answer,text="On further analysis we come to know that there is a percentage increase",font=("Times New Roman",15),bg="white").place(x=755,y=225)

Label(answer,text="in downloads by %f percent"%((vwd-nvwd)/nvwd\*100),font=("Times New Roman",15),bg="white").place(x=755,y=250)

else:

Label(answer,text="On further analysis whe come to know that there is a percentage decrease",font=("Times New Roman",15),bg="white").place(x=755,y=225)

Label(answer,text="in downloads by %f percent"%((vwd-nvwd)/nvwd\*100),font=("Times New Roman",15),bg="white").place(x=755,y=250)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=15,y=0)

answer.mainloop()

def ques6():

answer=Tk()

answer.title("Download Trend over the years 2016,2017 and 2018")

adjustWindow(answer,1400,700)

yr=IntVar(answer)

yrs=[2016,2017,2018]

new\_df = data[data["Year"].isin(yrs)]

new\_df = new\_df.reset\_index()

temp=new\_df.groupby("Year").mean()["Installs"]

inst = {}

for ele in yrs:

bs = 0

c=0

for i in range(len(new\_df)):

if (new\_df["Year"][i]==ele):

bs += new\_df["Installs"][i]

c+=1

inst.update({ele:bs/c})

Canvas(answer,height=690,width=790,bg="goldenrod1").place(x=0,y=0)

figure1 = plt.Figure(figsize=(8,6), dpi=100)

ax1 = figure1.add\_subplot(111)

line1 = FigureCanvasTkAgg(figure1, answer)

line1.get\_tk\_widget().pack(side=RIGHT,fill=BOTH)

temp.plot(kind='line',ax=ax1,legend=True,color="red")

ax1.set\_title('Install Trend Over the Period')

droplist=OptionMenu(answer,yr,\*yrs)

droplist.config(width=25,height=2,font=("bold",14))

yr.set("Select the Year")

def ok():

if(yr.get()=="Select the year"):

messagebox.showerror("Please select a valid year")

else:

data\_yr=data[data["Year"]==yr.get()]

cats=sorted(data\_yr["Category"].unique())

inst\_yr=data\_yr.groupby("Category")["Installs"].sum()

stats=inst\_yr.describe()

inst\_yr=list(inst\_yr)

inst\_yr

Canvas(answer,height=550,width=550,bg="white").place(x=20,y=100)

Label(answer,text="For the year %d we know the following information: "%yr.get(),font=(15),bg="white").place(x=35,y=160)

Label(answer,text="Mean = ",font=(15),bg="white").place(x=35,y=200)

Label(answer,text=str(stats["mean"]),bg="white",font=(15)).place(x=100,y=200)

Label(answer,text="Max = ",font=(15),bg="white").place(x=35,y=250)

Label(answer,text=str(stats["max"]),bg="white",font=(15)).place(x=100,y=250)

Label(answer,text="(%s)"%cats[inst\_yr.index(stats["max"])],bg="white",font=(15)).place(x=300,y=250)

Label(answer,text="Min = ",font=(15),bg="white").place(x=35,y=300)

Label(answer,text=str(stats["min"]),bg="white",font=(15)).place(x=100,y=300)

Label(answer,text="(%s)"%cats[inst\_yr.index(stats["min"])],bg="white",font=(15)).place(x=300,y=300)

Label(answer,text="On further calculations and comparision of the average \ndownloads for all three years we come to know that:",font=(15),bg="white",anchor="w").place(x=35,y=350)

if(inst[2017]-inst[2016]<0):

Label(answer,text="There is a Percentage increase from 2016 to 2017 by",font=(15),bg="white",anchor="w").place(x=35,y=400)

Label(answer,text=str((abs(inst[2016]-inst[2017])/inst[2016]\*100).round(2))+"%",font=(15),bg="white",anchor="w").place(x=35,y=450)

else:

Label(answer,text="There is a Percentage increase from 2016 to 2017 by",font=(15),bg="white",anchor="w").place(x=35,y=400)

Label(answer,text=str((abs(inst[2016]-inst[2017])/inst[2016]\*100).round(2))+"%",font=(15),bg="white",anchor="w").place(x=35,y=450)

if(inst[2018]-inst[2017]<0):

Label(answer,text="There is a Percentage increase from 2017 to 2018 by",font=(15),bg="white",anchor="w").place(x=35,y=500)

Label(answer,text=str((abs(inst[2018]-inst[2017])/inst[2017]\*100).round(2))+"%",font=(15),bg="white",anchor="w").place(x=35,y=550)

else:

Label(answer,text="There is a Percentage increase from 2017 to 2018 by",font=(15),bg="white",anchor="w").place(x=35,y=500)

Label(answer,text=str((abs(inst[2018]-inst[2017])/inst[2017]\*100).round(2))+"%",font=(15),bg="white",anchor="w").place(x=35,y=550)

Button(answer,text="View Average\nInstalls",width=15,height=2,font=("bold",14),command=ok).place(x=350,y=25)

droplist.place(x=25,y=25)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=1250,y=0)

answer.mainloop()

def ques5():

answer=Tk()

answer.title("Download Trend Category-Wise")

adjustWindow(answer,1400,700)

Canvas(answer,height=700,width=1400,bg="blue").pack()

cat=StringVar(answer)

#cats=list(sorted(data["Category"].unique()))

droplist=OptionMenu(answer,cat,\*cats)

droplist.config(width=25,height=2,font=("bold",14))

cat.set("Select the category of apps")

def ok():

if(cat.get()=="Select the category of apps"):

messagebox.showerror("Please select a valid category")

else:

Canvas(answer,height=450,width=400,bg="white").place(x=980,y=230)

Label(answer,text="Average number of installs for this category:",font=(15),bg="white").place(x=990,y=260)

Label(answer,text=data.groupby("Category")["Installs"].mean().reset\_index().iloc[cats.index(cat.get())],bg="white",font=(15)).place(x=995,y=310)

trend2=data.groupby("Category")["Installs"].mean()

trend2=trend2.sort\_values(ascending=False).reset\_index()

Label(answer,text="Top 2 categories with maximum Installs:",font=(15),bg="white").place(x=995,y=400)

Label(answer,text=trend2.loc[0:1],bg="White",font=(15),anchor="w").place(x=995,y=450)

trend2=data.groupby("Category")["Installs"].mean()

trend2=trend2.sort\_values().reset\_index()

Label(answer,text="Last 2 categories with minimum Installs:",font=(15),bg="white").place(x=995,y=545)

Label(answer,text=trend2.loc[0:1],bg="White",font=(15),anchor="w").place(x=1005,y=575)

new\_data=data[data["Category"]==cat.get()]

trend=new\_data.groupby("Year").mean()["Installs"]

# trend.set\_index("Year")

figure1 = plt.Figure(figsize=(15,10), dpi=60)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, answer)

bar1.get\_tk\_widget().place(x=20,y=20)

trend.plot(kind='line', legend=True, ax=ax1)

ax1.set\_title('Download Trend For The Category %s'%cat.get())

Button(answer,text="View Installs",width=15,height=2,font=("bold",14),command=ok).place(x=1055,y=150)

droplist.place(x=1000,y=50)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=1250,y=0)

answer.mainloop()

def ques4():

answer=Tk()

answer.title("Average ratings category wise")

adjustWindow(answer,1400,700)

cat=StringVar(answer)

#cats=list(sorted(data["Category"].unique()))

Canvas(answer,height=690,width=790,bg="turquoise").place(x=0,y=0)

rat1=data.groupby("Category").mean()["Rating"]

rat=list(rat1)

figure1 = plt.Figure(figsize=(13,6), dpi=60)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, answer)

bar1.get\_tk\_widget().pack(side=RIGHT,fill=BOTH)

rat1.plot(kind='line', legend=True, ax=ax1)

ax1.set\_title('Average Rating category-wise')

droplist=OptionMenu(answer,cat,\*cats)

droplist.config(width=25,height=2,font=("bold",14))

cat.set("Select the category of apps")

Label(answer,text="We know that there are %d different categories in our data. This report will \npresent the average rating that each category has."%(len(data["Category"].unique())),font=("Times New Roman",15)).pack(side=TOP,fill=BOTH)

def ok():

if(cat.get()=="Select the category of apps"):

messagebox.showerror("Please select a valid Category")

else:

Canvas(answer,height=50,width=550,bg="white").place(x=20,y=200)

Label(answer,text=cat.get(),font=(15),bg="white").place(x=35,y=210)

Label(answer,text=str(rat[cats.index(cat.get())])[0:4],bg="white",font=(15)).place(x=435,y=210)

Button(answer,text="View Average \n Rating",width=15,height=2,font=("bold",14),command=ok).place(x=350,y=100)

droplist.place(x=25,y=100)

Label(answer,text="Out of all the %d categories, the category with maximum ratings is: "%(len(cats)),font=("Times New Roman",15)).place(x=25,y=350)

Label(answer,text="Maximum: "+sorted(data["Category"].unique())[rat.index(max(data.groupby("Category").mean()["Rating"]))]+" "+str(max(data.groupby("Category").mean()["Rating"]))[:4],font=("Times New Roman",15,"bold"),bg="white").place(x=25,y=400)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=1250,y=0)

answer.mainloop()

def ques3():

answer=Tk()

adjustWindow(answer,750,600)

answer.configure(bg="thistle1")

cat\_data=data.groupby("Category").sum()["Installs"]

maximum=cat\_data[cat\_data==cat\_data.max()]

Label(answer,text="Out of total %d categories the category with -----"%(len(data["Category"].unique())),font=("Times New Roman",15,"bold"),bg="thistle1").place(x=50,y=50)

Label(answer,text="Maximum installs is ",bg="thistle1",font=("Times New Roman",15,"bold")).place(x=30,y=100)

Label(answer,text=maximum,font=("Times New Roman",15),bg="thistle1").place(x=30,y=150)

minimum=cat\_data[cat\_data==cat\_data.min()]

minimum

Label(answer,text="Minimum downloads is ",bg="thistle1",font=("Times New Roman",15,"bold")).place(x=350,y=100)

Label(answer,text=minimum,font=("Times New Roman",15),bg="thistle1").place(x=350,y=150)

cat\_data=data.groupby("Category").mean()["Installs"]

avg=cat\_data[cat\_data>=250000]

avg=pd.DataFrame(avg)

avg

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=600,y=550)

answer.mainloop()

def ques2():

answer=Tk()

adjustWindow(answer,750,600)

answer.configure(bg="chocolate1")

answer.title("Number of apps falling in ranges")

Label(answer,text="There are a total of %d apps "%len(data),font=("Times New Roman",20,"bold"),bg="chocolate1").place(x=10,y=10)

Label(answer,text="Out of these apps there can be found such ranges as follows :",font=("Times New Roman",20,"bold"),bg="chocolate1").place(x=10,y=50)

one=0

two=0

three=0

four=0

five=0

for i in range(len(data)):

if ((data["Installs"][i]>10000)&(data["Installs"][i]<=50000)):

one+=1

elif((data["Installs"][i]>50000)&(data["Installs"][i]<=150000)):

two+=1

elif((data["Installs"][i]>150000)&(data["Installs"][i]<=500000)):

three+=1

elif ((data["Installs"][i]>500000)&(data["Installs"][i]<=5000000)) :

four+=1

elif((data["Installs"][i]>5000000)) :

five+=1

Label(answer,text="Between 10,000 and 50,000 ----",font=("Tinmes New Roman",15),bg="chocolate1").place(x=10,y=200)

Label(answer,text="Between 50000 and 150000 ----",font=("Tinmes New Roman",15),bg="chocolate1").place(x=10,y=250)

Label(answer,text="Between 150000 and 500000 ----",font=("Tinmes New Roman",15),bg="chocolate1").place(x=10,y=300)

Label(answer,text="Between 500000 and 5000000 ----",font=("Tinmes New Roman",15),bg="chocolate1").place(x=10,y=350)

Label(answer,text="Greater than 5000000 ----",font=("Tinmes New Roman",15),bg="chocolate1").place(x=10,y=400)

Label(answer,text=str(one),font=("Tinmes New Roman",15),bg="chocolate1").place(x=300,y=200)

Label(answer,text=str(two),font=("Tinmes New Roman",15),bg="chocolate1").place(x=300,y=250)

Label(answer,text=str(three),font=("Tinmes New Roman",15),bg="chocolate1").place(x=300,y=300)

Label(answer,text=str(four),font=("Tinmes New Roman",15),bg="chocolate1").place(x=300,y=350)

Label(answer,text=str(five),font=("Tinmes New Roman",15),bg="chocolate1").place(x=300,y=400)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=600,y=550)

answer.mainloop()

def ques1():

global cats

global cat

answer=Tk()

answer.title("Percentage downloads category wise")

adjustWindow(answer,1400,700)

cat=StringVar(answer)

# cats=list(sorted(data["Category"].unique()))

Canvas(answer,height=690,width=790,bg="green").place(x=0,y=0)

percent=((data.groupby("Category").sum()["Installs"]/data.groupby("Category").sum()["Installs"].sum())\*100).round(decimals=3)

percent.reset\_index()

#Label(answer,text=percent,anchor="w",bg="green",font=("Times New Roaman",12)).pack()

percent1=percent[percent>3]

percent1["Others"]=sum(percent[percent<3])

percent1.reset\_index()

percent1

color=["yellow","Orange","Magenta","Cyan","Red","Lime","Pink","Teal","Lavender","Brown","Beige"]

figure1 = plt.Figure(figsize=(8,6), dpi=100)

ax1 = figure1.add\_subplot(111)

pie = FigureCanvasTkAgg(figure1, answer)

pie.get\_tk\_widget().pack(side=RIGHT,fill=BOTH)

percent1.plot(kind='pie',ax=ax1,autopct="%1.1f%%",counterclock=False,startangle=90,colors=color)

ax1.set\_title('PIE CHART\n Percentage downloads category wise')

Label(answer,text="Note: \"Others\" category include all the categories that have less than 3% of installs",bg="White").place(x=800,y=650)

Label(answer,text="We know that there are %d different categories in our data. This report will \npresent the percentge downloads that each category has."%(len(data["Category"].unique())),font=("Times New Roman",15)).pack(side=TOP,fill=BOTH)

percent1=percent[percent>3]

percent1["Others"]=sum(percent[percent<3])

percent1

Label(answer,text=percent1,font=("Times New Roman",15),bg="White").place(x=100,y=390)

droplist=OptionMenu(answer,cat,\*cats)

droplist.config(width=25,height=2,font=("bold",14))

cat.set("Select the category of apps")

def ok():

if(cat.get()=="Select the category of apps"):

messagebox.showerror(title="Error!",message="Please select a valid Category.")

else:

Canvas(answer,height=50,width=550,bg="white").place(x=20,y=200)

Label(answer,text=cat.get(),font=(15),bg="white").place(x=35,y=210)

Label(answer,text=str(percent[cats.index(cat.get())])+"%",bg="white",font=(15)).place(x=435,y=210)

Button(answer,text="View Percentage",width=15,height=2,font=("bold",14),command=ok).place(x=350,y=100)

droplist.place(x=25,y=100)

Label(answer,text="These are the top %d categories. Only those categories have been\n printed whose download percentage is greater than 3 percent"%(len(percent1)),font=("Times New Roman",15)).place(x=25,y=310)

Button(answer,text="Home Page",width=10,height=1,font=("bold",14),command=lambda:answer.destroy()).place(x=1250,y=0)

answer.mainloop()

def startScreen():

global root

root.destroy()

def ques():

Label(root,text="This report presents the following: ",font=("bold",20),bg="pink").pack()

Label(frame,text="1. Percentage downloads of all the apps category-wise ---",font=("Times New Roman",15,"bold")).grid(row=0,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques1).grid(row=0,column=5)

Label(frame,text="2. Total number of all the apps in a range of downloads ---",font=("Times New Roman",15,"bold")).grid(row=1,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques2).grid(row=1,column=5)

Label(frame,text="3. Most, Least and an Average of 2,50,000 downloads ---",font=("Times New Roman",15,"bold")).grid(row=2,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques3).grid(row=2,column=5)

Label(frame,text="4. Category of apps with maximum average rating ---",font=("Times New Roman",16,"bold")).grid(row=3,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques4).grid(row=3,column=5)

Label(frame,text="5. Download trend category-wise for the given data -----",font=("Times New Roman",15,"bold")).grid(row=4,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques5).grid(row=4,column=5)

Label(frame,text="6. Download trend over the years 2016,2017 & 2018 ---",font=("Times New Roman",15,"bold")).grid(row=5,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques6).grid(row=5,column=5)

Label(frame,text="7. Increase,Decrease in downloads over Android Version ---",font=("Times New Roman",15,"bold")).grid(row=6,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques7).grid(row=6,column=5)

Label(frame,text="8. Correlation between Installs and the Rating for the apps ---",font=("Times New Roman",14,"bold")).grid(row=7,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques8).grid(row=7,column=5)

Label(frame,text="9. Number of Positive & Negative sentiments the apps got ---",font=("Times New Roman",14,"bold")).grid(row=8,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques9).grid(row=8,column=5)

Label(frame,text="10. All Positive, Neutral, Negative reviews for the apps --- ",font=("Times New Roman",15,"bold")).grid(row=9,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques10).grid(row=9,column=5)

Label(frame,text="11. Analysis of 10 Best foods for you (Do user like it?) ---",font=("Times New Roman",15,"bold")).grid(row=10,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques11).grid(row=10,column=5)

Label(frame,text="12. Average Monthly downloads of apps for all the years ---",font=("Times New Roman",15,"bold")).grid(row=11,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques12).grid(row=11,column=5)

Label(frame,text="13. Add new records to datasets (App data or App review) ---",font=("Times New Roman",14,"bold")).grid(row=12,column=3)

Button(frame,text="Click Here",font=("bold",12),padx=25,pady=2,command=ques13).grid(row=12,column=5)

def myfunction(event):

canvas.configure(scrollregion=canvas.bbox("all"),width=700,height=700)

root=Tk()

adjustWindow(root,750,600)

myframe=Frame(root,relief=GROOVE,width=70,height=40,bd=1)

myframe.place(x=10,y=50)

canvas=Canvas(myframe)

frame=Frame(canvas)

canvas.pack(side="left")

canvas.create\_window((0,0),window=frame,anchor='nw')

frame.bind("<Configure>",myfunction)

ques()

root.mainloop()

def mainScreen():

global root

root=Tk()

root.title("Google Play Store Survey")

canvas=Canvas(width=750,height=600,bg="blue")

canvas.pack()

Button(root,width=15,height=1,text="Let's Start",font=("bold",15),fg="black",command=startScreen).place(x=550,y=550)

adjustWindow(root,750,600)

im = PIL.Image.open("D:\python\CaseStudy\Welcome.png")

photo = PIL.ImageTk.PhotoImage(im)

label = Label(canvas, image=photo)

label.image = photo # keep a reference!

label.pack()

root.mainloop()

mainScreen()