Bahria University,

Karachi Campus

COURSE:
ARTIFICIAL INTELLIGENCE
Term: Spring 2024
Class: BSE- 6(B)

Submitted By:

NAME: AIMAN ZIA SATTI

ENROLLNMENT: 02-131212-028

Submitted To:

FAIZ UL HAQ / ENGR.HAMZA

Signed _____ **Remarks:** _____

INDEX

SNO	DATE	LAB NO	LAB OBJECTIVE	SIGN
01	14-2-24	01	python programming with python Interview questions	
02	21-2-24	02	PYTHON PACKAGES	
03				
04				
05				
06				
07				
08				
09				
10				
11				
12				
13				
14				

Bahria University,

Karachi Campus

LAB NO. 02

LIST OF TASKS

TASK NO	OBJECTIVE
01	Develop a Python application to generate data visualizations Scenario: You are a data analyst working with a large dataset containing various types of data. Your task is to create a Python application that uses the Pandas, Matplotlib, and Seaborn libraries to perform exploratory data analysis and generate interactive visualizations. The application should allow users to load their dataset, explore the data, and create suitable charts and plots for visual analysis.
02	Implement a text summarization model using Transformers Scenario: As a natural language processing (NLP) researcher, you have been tasked with developing a text summarization model that can generate concise summaries of long text documents. Your task is to utilize the Transformers library in Python to build and train a summarization model. The model should be able to take a long text document as input and generate a concise summary that captures the key information and main ideas.
03	Implement a text summarization model using Transformers Scenario: As a natural language processing (NLP) researcher, you have been tasked with developing a text summarization model that can generate concise summaries of long text documents. Your task is to utilize the Transformers library in Python to build and train a summarization model. The model should be able to take a long text document as input and generate a concise summary that captures the key information and main ideas.
04	Build a web scraper using Beautiful Soup Scenario: You are a data engineer working for Pakveels. Your task is to develop a Python script that uses the Beautiful Soup library to scrape product information from competitor websites. The script should be able to extract data such as product names, descriptions, prices, and images from the target websites and store the data in a structured format (e.g., CSV or JSON) for further analysis.
05	Automate WhatsApp messaging using PyWhatKit Scenario: You are a software developer working on a project to automate communication for a small business. Your task is to create a Python script that uses the PyWhatKit library to automate the sending of messages and images through WhatsApp. The script should allow users to schedule the sending of messages or images to one or more contacts at specific times or intervals.
06	Develop a text-to-speech application using pyttsx3 Scenario: You are a developer working on an accessibility project to help visually impaired users interact with digital content. Your task is to create a Python application that uses the pyttsx3 library to convert text into spoken words. The application should allow users to input text, select voice settings (e.g., language, gender, rate), and generate audio output that can be played or saved to a file.

Submitted On:

22nd Feb, 2024

AIMAN ZIA SATTI **BSE-6B** **02-131212-028**

TASK # 1:

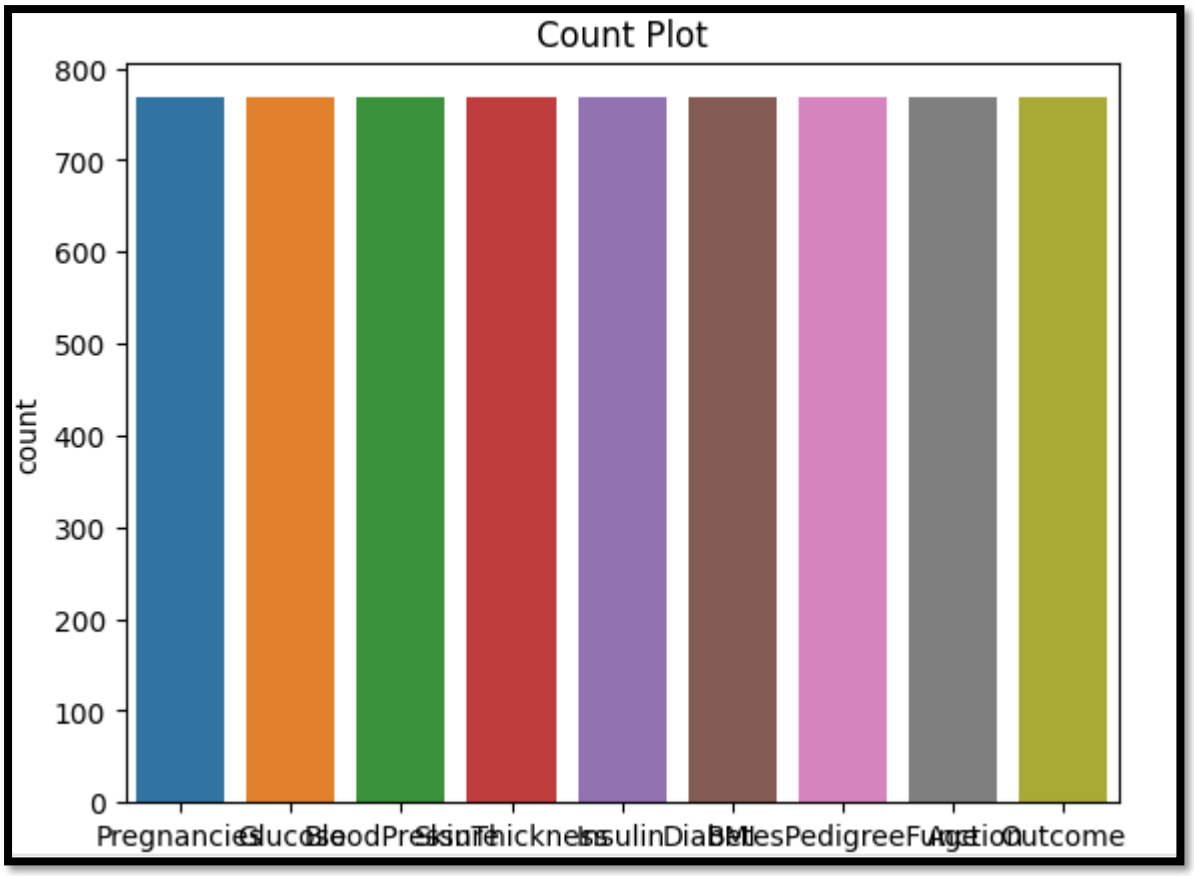
Develop a Python application to generate data visualizations Scenario: You are a data analyst working with a large dataset containing various types of data. Your task is to create a Python application that uses the Pandas, Matplotlib, and Seaborn libraries to perform exploratory data analysis and generate interactive visualizations. The application should allow users to load their dataset, explore the data, and create suitable charts and plots for visual analysis.

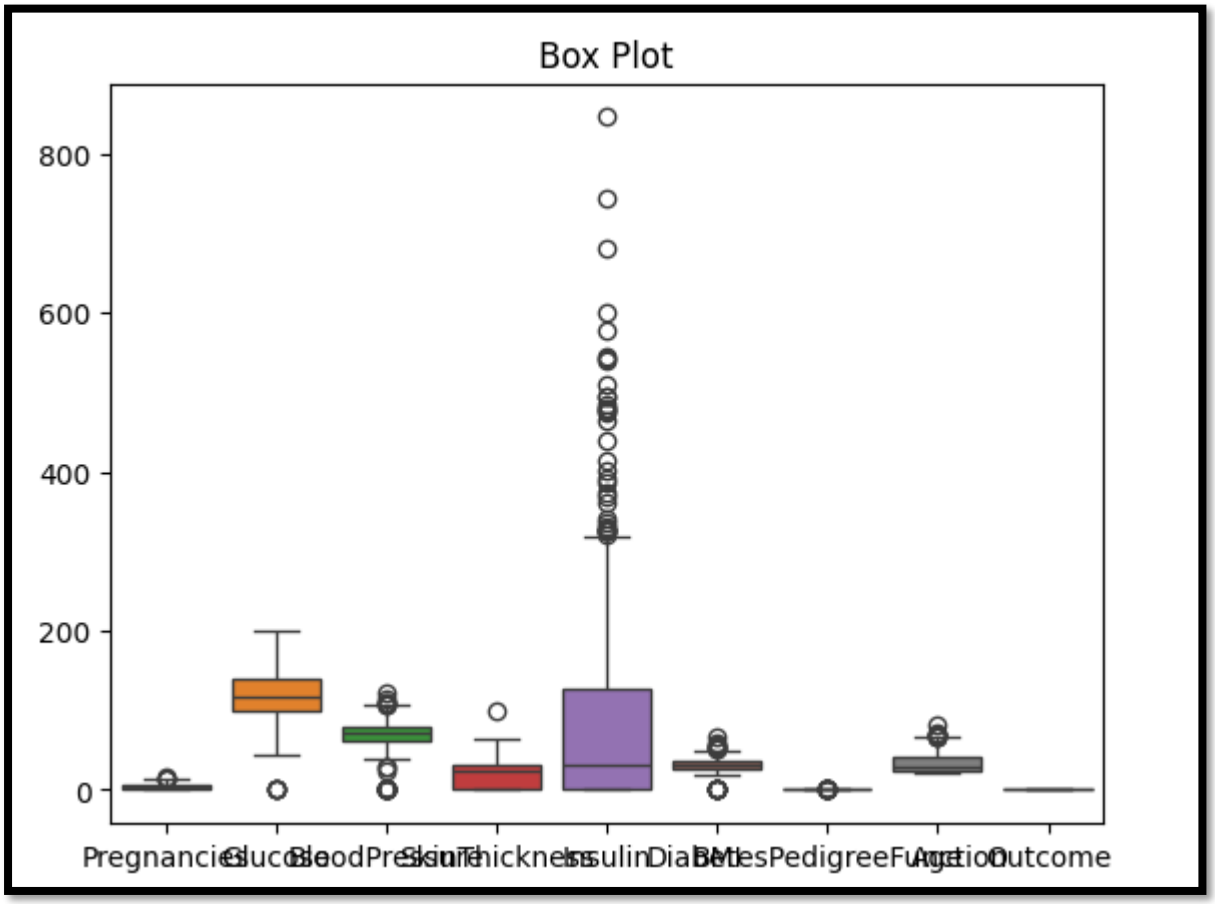
SOLUTION:

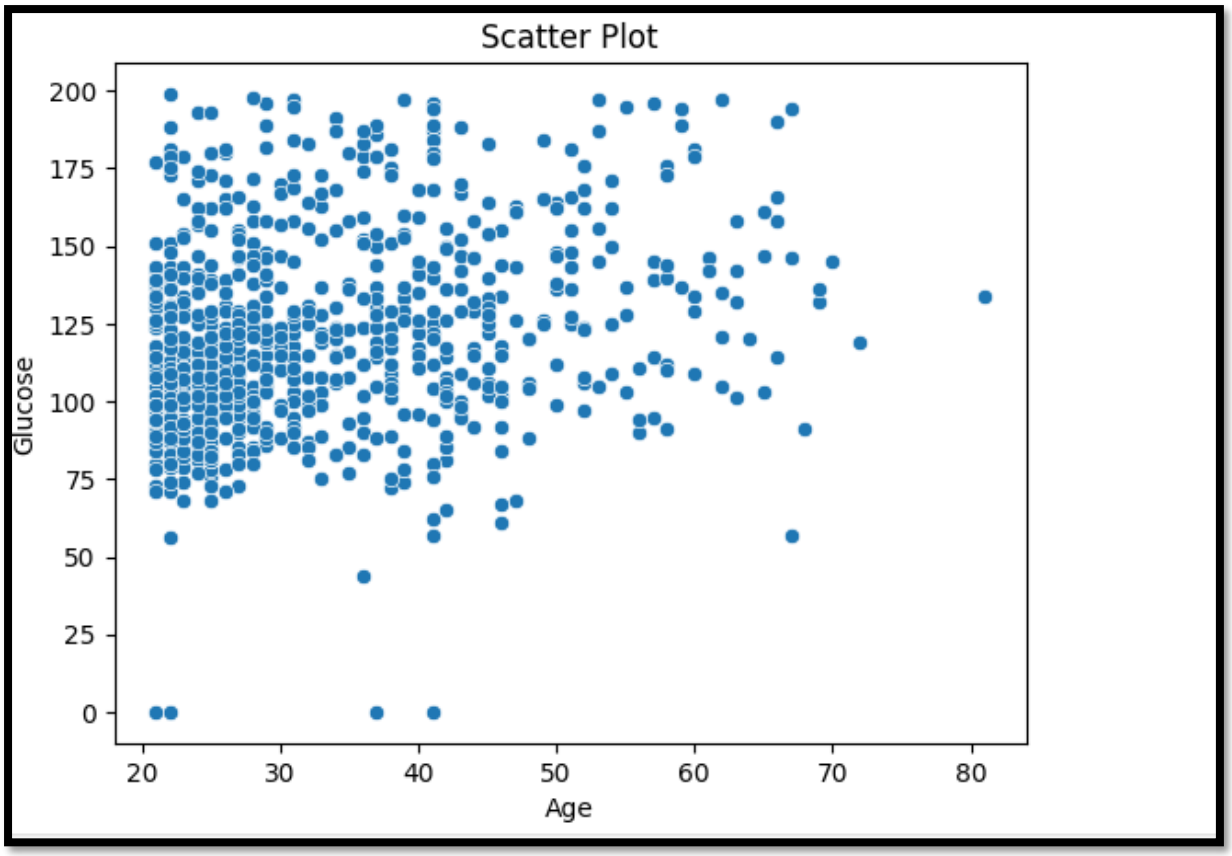
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

class DataVisualizer:
    def __init__(self):
        self.data = None
    def load_dataset(self, file_path):
        try:
            self.data = pd.read_csv("diabetes.csv")
            print("Dataset loaded successfully!")
        except FileNotFoundError:
            print("File not found. Please check the file path.")
    def explore_data(self):
        if self.data is not None:
            print("Dataset Information:")
            print(self.data.info())
            print("\nPreview of the dataset:")
            print(self.data.head())
            print("\nBasic Statistics:")
            print(self.data.describe())
        else:
            print("No dataset loaded yet.")
    def generate_plots(self):
        if self.data is not None:
            sns.countplot(data=self.data)
            plt.title("Count Plot")
            plt.show()
            sns.pairplot(data=self.data)
            plt.title("Pair Plot")
            plt.show()
            corr = self.data.corr()
            sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
            plt.title("Correlation Heatmap")
            plt.show()
        else:
            print("No dataset loaded yet.")
```

```
if __name__ == "__main__":
    visualizer = DataVisualizer()
    visualizer.load_dataset("your_dataset.csv")
    visualizer.explore_data()
    visualizer.generate_plots()
```

```
Preview of the dataset:
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1

Basic Statistics:
       Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin  \
count   768.000000  768.000000     768.000000     768.000000  768.000000
mean      3.845052  120.894531      69.105469      20.536458   79.799479
std       3.369578   31.972618      19.355807      15.952218  115.244002
min       0.000000    0.000000       0.000000       0.000000    0.000000
25%       1.000000   99.000000      62.000000       0.000000    0.000000
50%       3.000000  117.000000      72.000000      23.000000   30.500000
75%       6.000000  140.250000      80.000000      32.000000  127.250000
max      17.000000  199.000000     122.000000      99.000000  846.000000

              BMI  DiabetesPedigreeFunction         Age     Outcome
count  768.000000                768.000000  768.000000  768.000000
mean    31.992578                  0.471876   33.240885    0.348958
std      7.884160                  0.331329   11.760232    0.476951
min      0.000000                  0.078000   21.000000    0.000000
25%     27.300000                  0.243750   24.000000    0.000000
50%     32.000000                  0.372500   29.000000    0.000000
75%     36.600000                  0.626250   41.000000    1.000000
max     67.100000                  2.420000   81.000000    1.000000
```


Count Plot
count
800
700
600
500
400
300
200
100
0
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
Diabetes
BMI
PedigreeFunction
Age
Outcome


Box Plot
800
600
400
200
0
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
Diabetes
BMI
PedigreeFunction
Age
Outcome

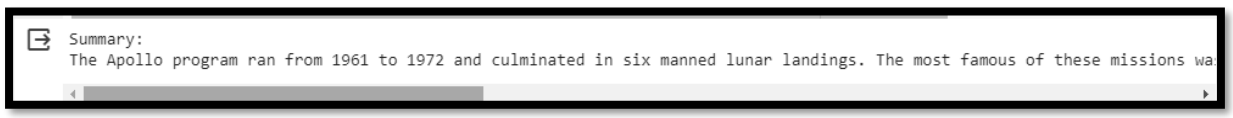


TASK # 2: Implement a text summarization model using Transformers Scenario: As a natural language processing (NLP) researcher, you have been tasked with developing a text summarization model that can generate concise summaries of long text documents. Your task is to utilize the Transformers library in Python to build and train a summarization model. The model should be able to take a long text document as input and generate a concise summary that captures the key information and main ideas.

SOLUTION:

```
from transformers import BartForConditionalGeneration, BartTokenizer

class TextSummarizer:
    def __init__(self):
        self.model =
BartForConditionalGeneration.from_pretrained("facebook/bart-large-cnn")
        self.tokenizer = BartTokenizer.from_pretrained("facebook/bart-
large-cnn")

    def summarize(self, text, max_length=5000, min_length=40):
        inputs = self.tokenizer.encode("summarize: " + text,
return_tensors="pt", max_length=1024, truncation=True)
        summary_ids = self.model.generate(inputs,
max_length=max_length, min_length=min_length, length_penalty=2.0,
num_beams=4, early_stopping=True)
```

```
        summary = self.tokenizer.decode(summary_ids[0],
skip_special_tokens=True)
        return summary
if __name__ == "__main__":
    text = """
    The Apollo program was a series of space missions undertaken by
NASA (National Aeronautics and Space Administration)
    with the goal of landing humans on the Moon and returning them
safely to Earth. It was initiated in response to the
    Soviet Union's early successes in space exploration, including the
launch of the first artificial satellite, Sputnik 1,
    in 1957. The Apollo program ran from 1961 to 1972 and culminated in
six manned lunar landings between 1969 and 1972.
    The most famous of these missions was Apollo 11, during which
astronauts Neil Armstrong and Buzz Aldrin became the first
    humans to set foot on the Moon on July 20, 1969. The program
achieved its goal of demonstrating American
    technological and scientific prowess in space, but it also faced
challenges, including the tragic loss of Apollo 1
    crew members in a cabin fire during a pre-launch test. Despite
these setbacks, the Apollo program remains one of
    the most significant achievements in human space exploration
history.
    """
    summarizer = TextSummarizer()
    summary = summarizer.summarize(text)
    print("Summary:")
    print(summary)
```




```
The Apollo program ran from 1961 to 1972 and culminated in six manned
lunar landings. The most famous of these missions was Apollo 11, during
which astronauts Neil Armstrong and Buzz Aldrin became the first humans
to set foot on the Moon. The program achieved its goal of demonstrating
American technological and scientific prowess in space.
```

TASK # 3: Convert images to sketches using OpenCV Scenario: You are a computer vision enthusiast working on a project to develop a photo editing application. Your task is to create a Python script that uses the OpenCV library to convert regular images into sketches. The script should allow users to select an image file, apply appropriate filters and transformations to convert it into a sketch-like image, and save the resulting image to disk.

SOLUTION:

```
import cv2
from google.colab.patches import cv2_imshow
```

```
def convert_to_sketch(image_path, save_path):
    image = cv2.imread(image_path)
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    inverted_gray_image = 255 - gray_image
    blurred_image = cv2.GaussianBlur(inverted_gray_image, (21, 21), 0)
    inverted_blurred_image = 255 - blurred_image
    sketch = cv2.divide(gray_image, inverted_blurred_image,
scale=256.0)
    cv2.imwrite(save_path, sketch)
    print("Sketch saved successfully!")
    cv2_imshow(sketch)
if __name__ == "__main__":
    input_image_path = "1688459626_newbabarazam.jpg"
    output_sketch_path = "1688459626_newbabarazam.jpg"
    convert_to_sketch(input_image_path, output_sketch_path)
```

ORIGINAL IMAGE:

SKETCHED IMAGE:

TASK # 4: Build a web scraper using Beautiful Soup Scenario: You are a data engineer working for Pakveels. Your task is to develop a Python script that uses the Beautiful Soup library to scrape product information from competitor websites. The script should be able to extract data such as product names, descriptions, prices, and images from the target websites and store the data in a structured format (e.g., CSV or JSON) for further analysis.

SOLUTION:

FORCES CATEGORY:

```
r = requests.get('https://www.pakwheels.com/used-cars/army-auction-jeep/430586')
html = r.text
soup = BeautifulSoup(html, "html.parser")
divs = soup.find_all("ul",{"list-unstyled search-vehicle-info fs13"})
car_detail = soup.find_all("ul",{"class":"list-unstyled search-vehicle-info-2 fs13"})
divsd = soup.find_all("a",{"class":"car-name ad-detail-path"})
divsd
key = "car name"
key0 = "category"
key1 = "city "
key2 = "make year"
key3= " total km"
key4 = "engine type"
```

```
key5 = "engine_cc"
list_of_dicts_armyacution = []
f = 0;
for i in divsd:
    detail = i.text
    detail = detail.strip("\n ")
    detailed  = detail.split()
    detailed.pop()
    detailed.pop()
    detail = ' '.join(detailed)
#     dictionary = {key: detail}
    details = divs[f].text
    details = details.strip("\n ")
    further_detail = car_detail[f].text
    further_detail = further_detail.strip("\n")
    further_details = further_detail.split()
    year = further_details[0]
    km = further_details[1]
    engine = further_details[3]
    e_type = further_details[4]
    dictionary = {key: detail,key0:"army-auction-jeep",key1:details,key2: year,key3:km,key4:
engine,key5:e_type}
    list_of_dicts_armyacution.append(dictionary)
    f +=1;
for i in list_of_dicts_armyacution:
    print(i)
```

MECHANIZED:

```
r = requests.get('https://www.pakwheels.com/used-cars/automatic/57336')
html = r.text
soup = BeautifulSoup(html, "html.parser")
divs = soup.find_all("ul",{"list-unstyled search-vehicle-info fs13"})
car_detail = soup.find_all("ul",{"class":"list-unstyled search-vehicle-info-2 fs13"})
divsd = soup.find_all("a",{"class":"car-name ad-detail-path"})
divsd
key = "car name"
key0 = "category"
key1 = "city "
key2 = "make year"
key3= " total km"
key4 = "engine type"
key5 = "engine_cc"
list_of_dicts = []
f = 0;
for i in divsd
    detail = i.text
    detail = detail.strip("\n ")
    detailed  = detail.split()
    detailed.pop()
    detailed.pop()
    detail = ' '.join(detailed)
#     dictionary = {key: detail}
    details = divs[f].text
    details = details.strip("\n ")
    further_detail = car_detail[f].text
    further_detail = further_detail.strip("\n")
```

```
    further_details = further_detail.split()
    year = further_details[0]
    km = further_details[1]
    engine = further_details[3]
    e_type = further_details[4]
    dictionary = {key: detail,key0:"automatic cars",key1:details,key2: year,key3:km,key4: engine,key5:e_type}
    list_of_dicts.append(dictionary)
    f +=1;
for i in list_of_dicts:
    print(i)
merged = list_of_dicts_armyacution +list_of_dicts_automatic + list_of_dicts_armyacution2
+list_of_dicts_automatic2+ list_of_dicts_automatic3 + list_of_dicts_automatic4+ list_of_dicts_japanese +
list_of_dicts_japanese2+ list_of_dicts_japanese3 + list_of_dicts_japanese4 + list_of_dicts_electric +
list_of_dicts_electric2 + list_of_dicts_electric3 + list_of_dicts_electric4 + list_of_dicts_sports +
list_of_dicts_sports2 + list_of_dicts_sports3 + list_of_dicts_sports4

len(merged)
import pandas as pd
df = pd.DataFrame(merged)
```

OUTPUT:

```
{'car name': 'Audi A3 2015 1.2 TFSI Exclusive Line', 'category': 'automatic cars', 'city ': 'Lahore', 'make year': '2015',
' total km': '70,000', 'engine type': 'Petrol', 'engine_cc': '1200'}
{'car name': 'Hyundai Tucson 2022 AWD A/T Ultimate', 'category': 'automatic cars', 'city ': 'Lahore', 'make year': '2022',
' total km': '9,000', 'engine type': 'Petrol', 'engine_cc': '2000'}
{'car name': 'Hyundai Sonata 2022 2.5', 'category': 'automatic cars', 'city ': 'Lahore', 'make year': '2022', ' total km':
'15,000', 'engine type': 'Petrol', 'engine_cc': '2500'}
{'car name': 'Audi A5 2017 1.4 TFSI Sportback for', 'category': 'automatic cars', 'city ': 'Lahore', 'make year': '2017',
' total km': '77,751', 'engine type': 'Petrol', 'engine_cc': '1400'}
{'car name': 'Toyota Land Cruiser 2012 ZX', 'category': 'automatic cars', 'city ': 'Lahore', 'make year': '2012', ' total
km': '82,000', 'engine type': 'Petrol', 'engine_cc': '4600'}
{'car name': 'Toyota Land Cruiser 2013 ZX for', 'category': 'automatic cars', 'city ': 'Lahore', 'make year': '2013', ' to
tal km': '62,938', 'engine type': 'Petrol', 'engine_cc': '4600'}
{'car name': 'Toyota Land Cruiser 2016 ZX for', 'category': 'automatic cars', 'city ': 'Lahore', 'make year': '2016', ' to
tal km': '71,066', 'engine type': 'Petrol', 'engine_cc': '4600'}
{'car name': 'Toyota Hilux 2013 Vigo Champ G', 'category': 'automatic cars', 'city ': 'Faisalabad', 'make year': '2013', '
total km': '110,000', 'engine type': 'Diesel', 'engine_cc': '2500'}
{'car name': 'Honda City 2021 1.5L ASPIRE CVT', 'category': 'automatic cars', 'city ': 'Islamabad', 'make year': '2021', '
total km': '35,000', 'engine type': 'Petrol', 'engine_cc': '1500'}
{'car name': 'Nissan Dayz 2018 Bolero X', 'category': 'automatic cars', 'city ': 'Islamabad', 'make year': '2018', ' total
km': '52,000', 'engine type': 'Petrol', 'engine_cc': '660'}
{'car name': 'Honda Civic Rebirth 2015 VTi Oriel Prosmatec 1.8 i-VTEC', 'category': 'automatic cars', 'city ': 'Lahore',
'make year': '2015', ' total km': '85,000', 'engine type': 'Petrol', 'engine_cc': '1800'}
{'car name': 'Honda Accord 2012 24TL Sports Style', 'category': 'automatic cars', 'city ': 'Karachi', 'make year': '2012',
' total km': '67,224', 'engine type': 'Petrol', 'engine_cc': '2400'}
{'car name': 'Honda Civic Reborn 2010 VTi Oriel Prosmatec 1.8 i-VTEC', 'category': 'automatic cars', 'city ': 'Sahiwal',
'make year': '2010', ' total km': '65,000', 'engine type': 'Petrol', 'engine_cc': '1800'}
{'car name': 'Honda Civic Rebirth 2012 VTi Prosmatec 1.8 i-VTEC', 'category': 'automatic cars', 'city ': 'Karachi', 'make
year': '2012', ' total km': '68,000', 'engine type': 'Petrol', 'engine_cc': '1800'}
```

Out[104]:

	car name	category	city	make year	total km	engine type	engine_cc	price
0	Toyota Prado 2000	army-auction-jeep	Rawalpindi	2000	185,000	Diesel	2800	PKR 45 lacs
1	Toyota Land Cruiser 1987	army-auction-jeep	Islamabad	1987	100,000	Petrol	3500	PKR 27 lacs
2	Suzuki Cultus 2004 VXR (CNG)	army-auction-jeep	Islamabad	2004	128,500	Petrol	1000	PKR 7.85 lacs
3	Toyota Land Cruiser 1990	army-auction-jeep	Islamabad	1990	120,000	Diesel	3500	PKR 41 lacs
4	Toyota Land Cruiser 80 Series 1995 VX 4.2D	army-auction-jeep	Karachi	1995	60,000	Diesel	4200	PKR 38 lacs
...	...	...	...	...	...	...	...	...
501	Daihatsu Copen 2015 Robe S Sports Car	electric	Lahore	2015	48,000	Petrol	660	PKR 31.75 lacs
502	Nissan Note 2021 e-Power Aura Sports Car	electric	Islamabad	2021	21,000	Hybrid	1198	PKR 64.75 lacs
503	Nissan Note 2007 Sports Car	electric	Swabi	2007	98,059	Petrol	1500	PKR 19 lacs
504	Toyota 86 1987 Sports Car	electric	Peshawar	1987	100,000	CNG	1300	PKR 3.2 lacs
505	Daihatsu Copen 2015 Robe Sports Car	electric	Burewala	2015	55,000	Petrol	660	PKR 41 lacs

506 rows × 8 columns

```
df.sort_values("make year", ascending = False)
```

Out[29]:

	car name	category	city	make year	total km	engine type	engine_cc	price
81	Haval H6 2023 2.0T AWD	automatic cars	Hasilpur	2023	9,500	Petrol	2000	PKR 1.05 crore
368	MG 4 2023 Essence	electric	Islamabad	2023	5	Electric	64.0	Call
342	MG 4 2023 Essence	electric	Lahore	2023	5	Electric	64.0	PKR 1.36 crore
345	BMW i7 2023 xDrive60 M Pro Package	electric	Lahore	2023	900	Electric	102.0	PKR 8.88 crore
329	Mercedes Benz EQS 2023	japanese	Lahore	2023	30	Electric	108.0	Call

	A	B	C	D	E	F	G	H	I	J
1	car name	category	city	make year	total km	engine type	engine_cc	price		
2	Toyota Prado 2000	army-auction-jeep	Rawalpindi	2000	185,000	Diesel	2800	PKR 45 lacs		
3	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1987	100,000	Petrol	3500	PKR 27 lacs		
4	Suzuki Cultus 2004 VX	army-auction-jeep	Islamabad	2004	128,500	Petrol	1000	PKR 7.85 lacs		
5	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1990	120,000	Diesel	3500	PKR 41 lacs		
6	Toyota Land Cruiser 8	army-auction-jeep	Karachi	1995	60,000	Diesel	4200	PKR 38 lacs		
7	Toyota Land Cruiser 1	army-auction-jeep	Hyderabad	1990	100,000	Petrol	4200	PKR 30 lacs		
8	Nissan Sunny 1987 EX	army-auction-jeep	Lahore	1987	10,000	Petrol	1300	PKR 6.8 lacs		
9	Jeep Wrangler 1997 S	army-auction-jeep	Islamabad	1997	85,000	Diesel	4000	PKR 45 lacs		
10	Toyota Land Cruiser 1	army-auction-jeep	Rawalpindi	1983	6,000	Diesel	3000	PKR 14.8 lacs		
11	Suzuki Potohar 1988 E	army-auction-jeep	Karachi	1988	123	Petrol	1000	PKR 14.5 lacs		
12	Toyota Land Cruiser 1	army-auction-jeep	Wah cantt	1988	105,000	Diesel	4200	PKR 25 lacs		
13	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1995	185,000	Diesel	4200	PKR 32 lacs		
14	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1990	100,000	Diesel	4200	PKR 33 lacs		
15	Land Rover Freelande	army-auction-jeep	Islamabad	2003	180,000	Petrol	2200	PKR 24 lacs		
16	Toyota Fj Cruiser 198	army-auction-jeep	Faisalabad	1982	100	Diesel	2000	PKR 9 lacs		
17	Toyota Land Cruiser 1	army-auction-jeep	Karachi	1990	10	Diesel	3500	PKR 25 lacs		
18	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1990	60,000	Diesel	4200	PKR 23.25 lacs		
19	Jeep Other 1968	army-auction-jeep	Lahore	1968	10,000	Diesel	2800	PKR 12 lacs		
20	Toyota Land Cruiser 1	army-auction-jeep	Faisalabad	1987	1	Diesel	4200	PKR 34.5 lacs		
21	Willys M38 1964	army-auction-jeep	Attock	1964	123,456	Diesel	2400	PKR 22.5 lacs		
22	Toyota Land Cruiser 1	army-auction-jeep	Karachi	1991	100,000	Petrol	4000	PKR 40 lacs		
23	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1984	122,222	Diesel	3000	PKR 55 lacs		
24	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1990	22,000	Petrol	3400	PKR 31.5 lacs		
25	Toyota Land Cruiser 1	army-auction-jeep	Islamabad	1976	123	Diesel	3200	PKR 13 lacs		
26	Toyota Fj Cruiser 198	army-auction-jeep	Islamabad	1983	12,345	Diesel	3000	PKR 14.5 lacs		

Output

TASK # 5: Automate WhatsApp messaging using PyWhatKit Scenario: You are a software developer working on a project to automate communication for a small business. Your task is to create a Python script that uses the PyWhatKit library to automate the sending of messages and

images through WhatsApp. The script should allow users to schedule the sending of messages or images to one or more contacts at specific times or intervals.

SOLUTION:

```
!pip install pywhatkit
```

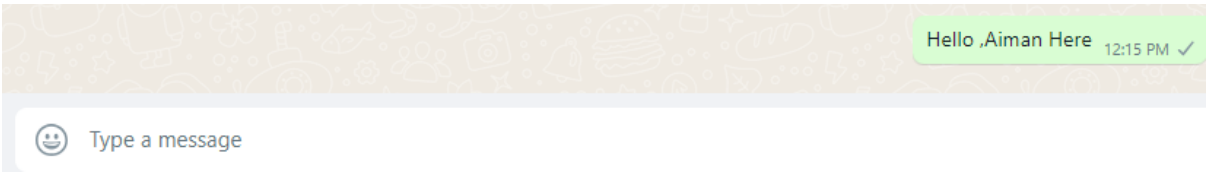
[7] ✓ 5.7s Python

```
Requirement already satisfied: pywhatkit in c:\users\ma\appdata\local\programs\pythor
Requirement already satisfied: Pillow in c:\users\ma\appdata\local\programs\python\py
Requirement already satisfied: pyautogui in c:\users\ma\appdata\local\programs\pythor
Requirement already satisfied: requests in c:\users\ma\appdata\local\programs\python\
Requirement already satisfied: wikipedia in c:\users\ma\appdata\local\programs\pythor
Requirement already satisfied: Flask in c:\users\ma\appdata\local\programs\python\pyt
Requirement already satisfied: Werkzeug>=3.0.0 in c:\users\ma\appdata\local\programs\
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\ma\appdata\local\programs\py
Requirement already satisfied: itsdangerous>=2.1.2 in c:\users\ma\appdata\local\progr
Requirement already satisfied: click>=8.1.3 in c:\users\ma\appdata\local\programs\pyt
Requirement already satisfied: blinker>=1.6.2 in c:\users\ma\appdata\local\programs\p
Requirement already satisfied: pymsgbox in c:\users\ma\appdata\local\programs\python\
Requirement already satisfied: pytweening>=1.0.4 in c:\users\ma\appdata\local\progran
Requirement already satisfied: pyscreeze>=0.1.21 in c:\users\ma\appdata\local\progran
Requirement already satisfied: pygetwindow>=0.0.5 in c:\users\ma\appdata\local\progra
Requirement already satisfied: mouseinfo in c:\users\ma\appdata\local\programs\pythor
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\ma\appdata\local\
Requirement already satisfied: idna<4,>=2.5 in c:\users\ma\appdata\local\programs\pyt
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\ma\appdata\local\progra
Requirement already satisfied: certifi>=2017.4.17 in c:\users\ma\appdata\local\progra
Requirement already satisfied: beautifulsoup4 in c:\users\ma\appdata\local\programs\p
Requirement already satisfied: colorama in c:\users\ma\appdata\local\programs\python\
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\ma\appdata\local\programs\
```

```
import pywhatkit as kit
import time

kit.sendwhatmsg_instantly("+923102264316","Hello ,Aiman Here")
time.sleep(1)
print("message sent successfully")
```

[5] ✓ 16.7s Python

```
message sent successfully
```



TASK # 6: Develop a text-to-speech application using pyttsx3

Scenario: You are a developer working on an accessibility project to help visually impaired users interact with digital content. Your task is to create a Python application that uses the pyttsx3 library to convert text into spoken words. The application should allow users to input text, select voice settings (e.g., language, gender, rate), and generate audio output that can be played or saved to a file.

SOLUTION:

```
import pyttsx3
engine=pyttsx3.init()
engine.setProperty('rate',150)
engine.setProperty('volume',0.9)
voices=engine.getProperty('voices')
engine.setProperty('voice',voices[1].id)
text="HELLO ! I AM AIMAN ZIA SATTI"
engine.say(text)
engine.runAndWait()
engine.save_to_file(text,'output.mp3')
print("Voice Successfully Run")
engine.runAndWait()
```

```
Voice Successfully Run
```