**Bahria University,**

**Karachi Campus**

**A logo with text on it

Description automatically generated**

**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**

**A logo with text on it

Description automatically generated**

**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***

**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.")

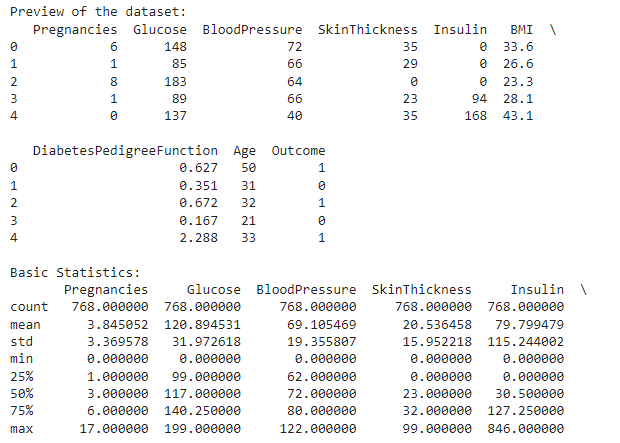
if \_\_name\_\_ == "\_\_main\_\_":

    visualizer = DataVisualizer()

    visualizer.load\_dataset("your\_dataset.csv")

    visualizer.explore\_data()

    visualizer.generate\_plots()

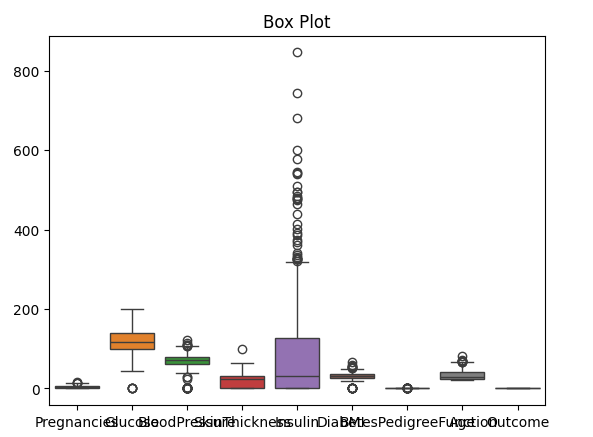


A screenshot of a computer

Description automatically generated

A graph showing different colored lines

Description automatically generated with medium confidence



A diagram of a scatter plot

Description automatically generated

**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.

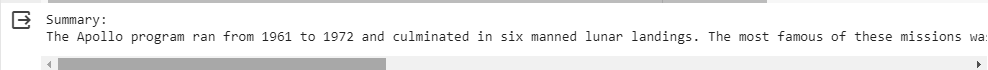
    """

    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:**

A person in a green uniform

Description automatically generated

**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)

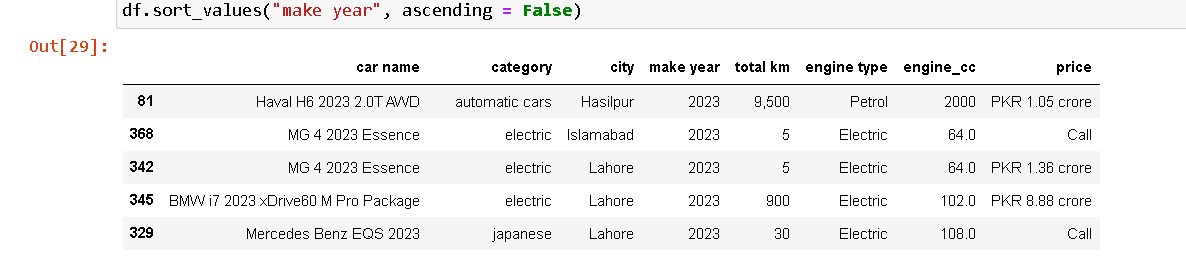
import pandas as pd

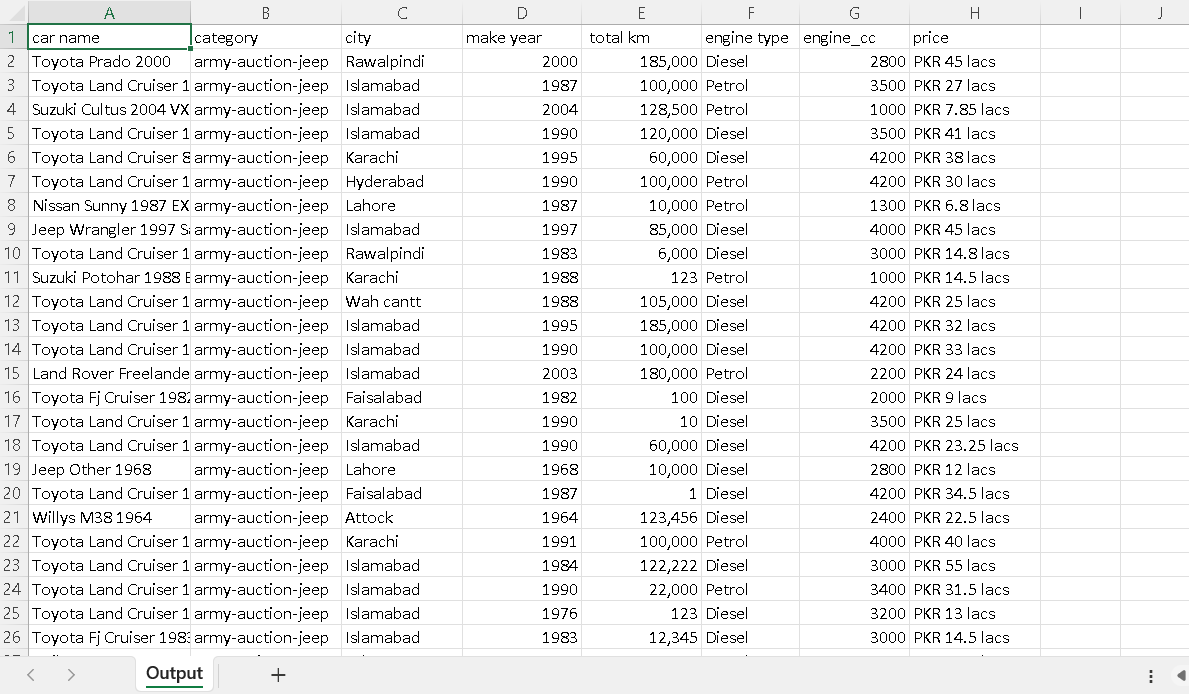
df = pd.DataFrame(merged)

**OUTPUT:**

****

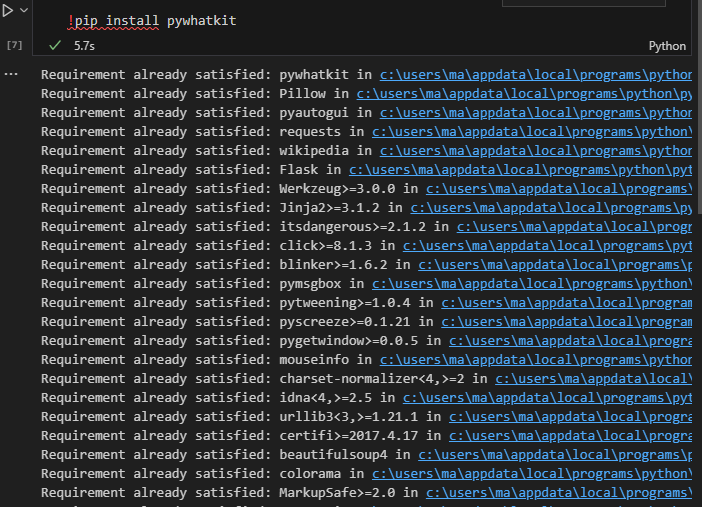
****

****

****

**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:**



A screen shot of a computer code

Description automatically generated

A white background with a white border

Description automatically generated with medium confidence

**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:**

A screenshot of a computer program

Description automatically generated