TW1

```
'''
   Develop a menu driven program to implement a queue. The operation would be
   a.Add an item to the queue
   b.Delete an item from queue
   c.Display the queue
'''


li=[]
val=1
print("1->.Add an item to the queue\n2->.Delete an item from queue\n3->.Display the queue\n4->exit")
while val!=4:
   val=int(input("Select one option: "))
   if val==1:
      a=input("enter a value to add: ")
      li.append(a)
   elif val==2:
      print("the value ",li.pop(0)," is removed")
   elif val==3:
      print("Queue: ",*li)
```

TW2

```
''''
   Store the following information in a dictionary:
   Course code: course Name,faculty,Number of registerations
   Perform the following operations using functions
   a.Find Course Name that has highest number of registration
   b.Given the Ciurse Code, Display the associated details
   Display details of all courses.
'''
```

```
def find_highest(detail):
    li=[[detail[a][2], detail[a][0]] for a in detail]
    val=li[0]
    for a in li[1:]:
        if val[0]<a[0]:
            val=a
    return val[1]


def find_course(details,course):
    val=details.get(course,0)
    if val==0:
        return "NO such course Found"
    return val


course_details={"MAT134":["Math","Jayesh",13],"DSA132":["DSA","Thapa",15],"PYT34":["Python","
Abhishek",30],"SED32":["SED","Maggie",86]}


print("Available courses are: ",course_details)
print("Highest registered courese is: ",find_highest(course_details))
course=input("enter course code: ")
print(find_course(course_details,course))
```

TW3

'''

write a python program to read the book information from the user and store in a CSV file containing rows in the following format:

bookNo,title,author,price

Develop a menu-driven program(with functions for each) with the following options:

1:Search Book by author

2:Search Books below specified price (Rise an exception if price entered is <=0

3:Search Books where tilte contains the specified word

```
    4:Exit
'''
import csv
# fields=['bookNo','title','author','price']
with open("books.csv",'w',newline="") as file:
    writer = csv.writer(file)
    count=int(input("Enter the number of books: "))
    # writer.writerow(fields)
    data=[]
    for i in range(count):
        li=[]
        print("enter details for item ",i+1)
        li.append(i+1)
        li.append(input("Enter book title: "))
        li.append(input("author name: "))
        li.append(int(input("Enter Price: ")))
        data.append(li)
    print(data)
    writer.writerows(data)

def search_by_author(value):
    with open("books.csv", "r") as file:
        reader = csv.reader(file)
        for a in reader:
            if a[2].lower() == value.lower():
                return a
        return []

def search_by_price(value):
    if value<=0:
        raise Exception("Value of price can't be less than or equal to 0")
```

```
    with open("books.csv", "r") as file:
        reader = csv.reader(file)
        li=[]
        for a in reader:
            if int(a[3]) < value:
                li.append(a)
        return li


def search_by_word(value):
    with open("books.csv", "r") as file:
        reader = csv.reader(file)
        li=[]
        for a in reader:
            if value in a[1].split():
                li.append(a)
        return li


print("1:Search Book by author\n2:Search Books below specified price (Rise an exception if price entered is <=0\n3:Search Books where tilte contains the specified word\n4:Exit")
while True:
    val=int(input("enter choice: "))
    if val == 1:
        value = input("Enter books author: ")
        print("book is: ",search_by_author(value))
    elif val==2:
        value=int(input("enter price: "))
        print("books are: \n",search_by_price(value))
    elif val==3:
        value = input("enter word: ")
        print("books are: \n",search_by_word(value))
    else:
```

```
        print("Bye!!")
        exit(0)
```

TW4

```
'''
Write a Python program to perform the following:
a. Create a database named “products.db”
b. Create a table named “products” that has the following fields:
prodID: int
name: text
quantity: int
price: real
c. Insert n records into the table reading the values for each item from the user.
d. Display the recordset after fetching all the rows.
e. Delete a product whose product ID is entered by the user.
f. Increase the price of all products whose current price is less than Rs.50, by 10%.
g. Display all the products whose quantity is less than 40.
'''


import sqlite3

# Create a database and connect to it
conn = sqlite3.connect("products.db")
cursor = conn.cursor()

# Create the 'products' table
cursor.execute('''CREATE TABLE IF NOT EXISTS products (
                prodID INTEGER PRIMARY KEY,
                name TEXT,
```

```
            quantity INTEGER,
            price REAL
        )''')


# Function to insert records into the table
def insert_record():
    n = int(input("Enter the number of records to insert: "))
    for _ in range(n):
        prodID = int(input("Enter product ID: "))
        name = input("Enter product name: ")
        quantity = int(input("Enter quantity: "))
        price = float(input("Enter price: "))
        cursor.execute("INSERT INTO products (prodID, name, quantity, price) VALUES (?, ?, ?, ?)",
                       (prodID, name, quantity, price))
        conn.commit()


# Function to display all records
def display_records():
    cursor.execute("SELECT * FROM products")
    records = cursor.fetchall()
    for record in records:
        print(record)


# Function to delete a product by product ID
def delete_product():
    prodID = int(input("Enter the product ID to delete: "))
    cursor.execute("DELETE FROM products WHERE prodID = ?", (prodID,))
    conn.commit()
```

```
    print("Product deleted successfully.")


# Function to increase the price of products under Rs.50 by 10%
def increase_price():
    cursor.execute("UPDATE products SET price = price * 1.1 WHERE price < 50")
    conn.commit()
    print("Prices increased for eligible products.")


# Function to display products with quantity less than 40
def display_low_quantity_products():
    cursor.execute("SELECT * FROM products WHERE quantity < 40")
    records = cursor.fetchall()
    for record in records:
        print(record)


# Main program
while True:
    print("\nMenu:")
    print("1. Insert records")
    print("2. Display records")
    print("3. Delete a product")
    print("4. Increase price of products under Rs.50")
    print("5. Display products with quantity less than 40")
    print("6. Exit")

    choice = int(input("Enter your choice: "))

    if choice == 1:
```

```
        insert_record()
    elif choice == 2:
        display_records()
    elif choice == 3:
        delete_product()
    elif choice == 4:
        increase_price()
    elif choice == 5:
        display_low_quantity_products()
    elif choice == 6:
        break
    else:
        print("Invalid choice. Please select a valid option.")

# Close the connection
conn.close()
```

TW5

```
class Product:
    def __init__(self,name='',price=0.0,discount_percentage=0.0):
        self.name=name
        self.price=price
        self.discount_percentage=discount_percentage

    def getDiscountAmount(self):
        return round(self.price * self.discount_percentage /100,2)

    def getDiscountPrice(self):
        return round(self.price - self.getDiscountAmount(),2)

    def __str__(self):
```

```
        return f'Name:{self.name}\nPrice:{self.price}\nDiscount percentage{self.price}%\nDiscount amount:{self.getDiscountPrice()}\nDiscount price:{self.getDiscountPrice()}\n'
li=[]
li.append(Product("Stanley 13 Ounce Wood Hammer",12.99,62))
li.append(Product("National Hardware 3/4\" Wire Nails",100,30))
li.append(Product("Economy Duct Yape, 60 yds, Silver",1000,25))


val=True
print("Products:")
for a in li:
    print(a.name)
while val:
    value=int(input("Enter product number: "))
    print(li[value-1])
    temp=input("View another product? (y/n): ")
    val=temp.lower()=='y'
```

TW6

'''

Create an object-oriented program that allows you to enter data for customers and employees.

Problem Details

Create a Person class that provides attributes for first name, last name, and emailaddress. This class should provide a property or method that returns the person's fullname.

Create a Customer class that inherits the Person class. This class should add anattribute for a customer number.

Create an Employee class that inherits the Person class. This class should add anattribute for a PAN number.

The program should create a Customer or Employee object from the data entered bythe user, and it should use this object to display the data to the user. To do that, theprogram can use the

```
isinstance() function to check whether an object is a Customer orEmployee object.
'''

class Person:
    def __init__(self, first_name, last_name, email_address):
        self.first_name = first_name
        self.last_name = last_name
        self.email_address = email_address

    def get_fullname(self):
        return f"{self.first_name} {self.last_name}"


class Customer(Person):
    def __init__(self, first_name, last_name, email_address, customer_number):
        super().__init__(first_name, last_name, email_address)
        self.customer_number = customer_number


class Employee(Person):
    def __init__(self, first_name, last_name, email_address, pan_number):
        super().__init__(first_name, last_name, email_address)
        self.pan_number = pan_number


def main():
    people = []  # List to store created objects

    while True:
        print("Menu:")
        print("1. Add Customer")
```

```
print("2. Add Employee")
print("3. Display All")
print("4. Exit")

choice = int(input("Enter your choice: "))

if choice == 1:
    first_name = input("Enter customer's first name: ")
    last_name = input("Enter customer's last name: ")
    email_address = input("Enter customer's email address: ")
    customer_number = input("Enter customer number: ")

    customer = Customer(first_name, last_name, email_address, customer_number)
    people.append(customer)  # Add the object to the list
    print("Customer added successfully.")
elif choice == 2:
    first_name = input("Enter employee's first name: ")
    last_name = input("Enter employee's last name: ")
    email_address = input("Enter employee's email address: ")
    pan_number = input("Enter PAN number: ")

    employee = Employee(first_name, last_name, email_address, pan_number)
    people.append(employee)  # Add the object to the list
    print("Employee added successfully.")
elif choice == 3:
    print("\nAll Created Objects:")
    for person in people:
        print(type(person).__name__)  # Display the class name (Customer or Employee)
        print(f"Full Name: {person.get_fullname()}")
        print(f"Email Address: {person.email_address}")
        if isinstance(person, Customer):
```

```
            print(f"Customer Number: {person.customer_number}")
        elif isinstance(person, Employee):
            print(f"PAN Number: {person.pan_number}")
        print("-" * 20)
    elif choice == 4:
        break
    else:
        print("Invalid choice. Please select a valid option.")


if __name__ == "__main__":
    main()
```

TW7

```
'''
Develop the following GUI application.
'''

import tkinter as tk
from tkinter import messagebox

def add():
    try:
        result.set(float(entry_num1.get()) + float(entry_num2.get()))
    except ValueError:
        messagebox.showerror("Error", "Please enter valid numbers.")

def subtract():
    try:
```

```
        result.set(float(entry_num1.get()) - float(entry_num2.get()))
    except ValueError:
        messagebox.showerror("Error", "Please enter valid numbers.")

def multiply():
    try:
        result.set(float(entry_num1.get()) * float(entry_num2.get()))
    except ValueError:
        messagebox.showerror("Error", "Please enter valid numbers.")

def divide():
    try:
        num2 = float(entry_num2.get())
        if num2 == 0:
            messagebox.showerror("Error", "Cannot divide by zero.")
        else:
            result.set(float(entry_num1.get()) / num2)
    except ValueError:
        messagebox.showerror("Error", "Please enter valid numbers.")

# Create the main window
window = tk.Tk()
window.title("Simple Calculator")

# Entry fields for numbers
entry_num1 = tk.Entry(window)
entry_num1.pack()

entry_num2 = tk.Entry(window)
entry_num2.pack()
```

```
# Result display
result = tk.StringVar()
result_label = tk.Label(window, textvariable=result)
result_label.pack()

# Buttons for operations
add_button = tk.Button(window, text="Add", command=add)
add_button.pack()

subtract_button = tk.Button(window, text="Subtract", command=subtract)
subtract_button.pack()

multiply_button = tk.Button(window, text="Multiply", command=multiply)
multiply_button.pack()

divide_button = tk.Button(window, text="Divide", command=divide)
divide_button.pack()

# Start the GUI event loop
window.mainloop()
```

TW8

```
'''

Problem Statement: Exam Score Analysis and Visualization
An exam has been conducted for a class of students. The exam data is stored in a CSV file, containing the student names and their scores. Develop a Python program to analyse the exam scores, calculate key statistics, and visualize the data to gain insights into the students&amp;#39; performance.

'''
```

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

data = pd.read_csv("exam_scores.csv")
mean_score = data['Score'].mean()
median_score = data['Score'].median()
std_dev = data['Score'].std()

print(f"Mean Score: {mean_score:.2f}")
print(f"Median Score: {median_score:.2f}")
print(f"Standard Deviation: {std_dev:.2f}")

# Histogram
plt.hist(data['Score'], bins=10, edgecolor='black')
plt.title("Exam Score Distribution")
plt.xlabel("Score")
plt.ylabel("Frequency")
plt.show()

# Box plot
plt.boxplot(data['Score'])
plt.title("Exam Score Distribution")
plt.ylabel("Score")
plt.show()
```