


If you have been ill or had other adverse circumstances which you believe have affected any of your exams or other assessments, this can be considered through our regulations on 'Good Cause'. Good Cause claims are submitted via MyCampus and further information on the process is explained in our **'Key FAQs for students'**. 

Time left 1:56:37

Hide

Question 1

Not yet answered

Marked out of 40.00

Part 1: Python Question

This exam consists of three tasks. You are required to write the code for each task in the same Python source file. You should name the file **your_name_python_task.py** (fill in your own name).

Python Task 1: Python Basics and Data Manipulation [13 Marks][Download this "sales_data.csv" file](#)

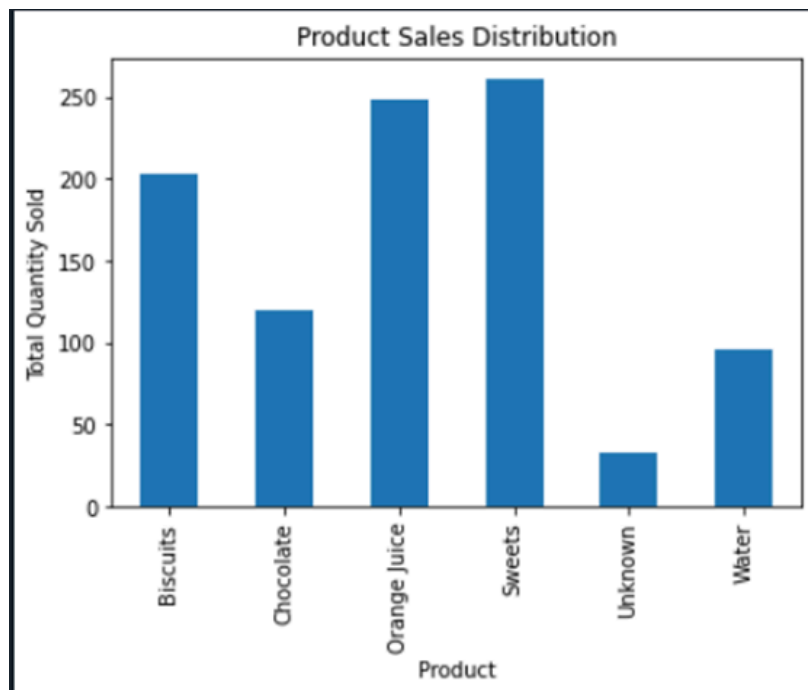
You have been provided with a CSV file **sales_data.csv** containing sales data for a corner shop. The file contains the following columns: Date, Product, Quantity, Price, and Total. Your task is to write a Python program that performs the following operations:

Task 1a: Data Loading and Preprocessing

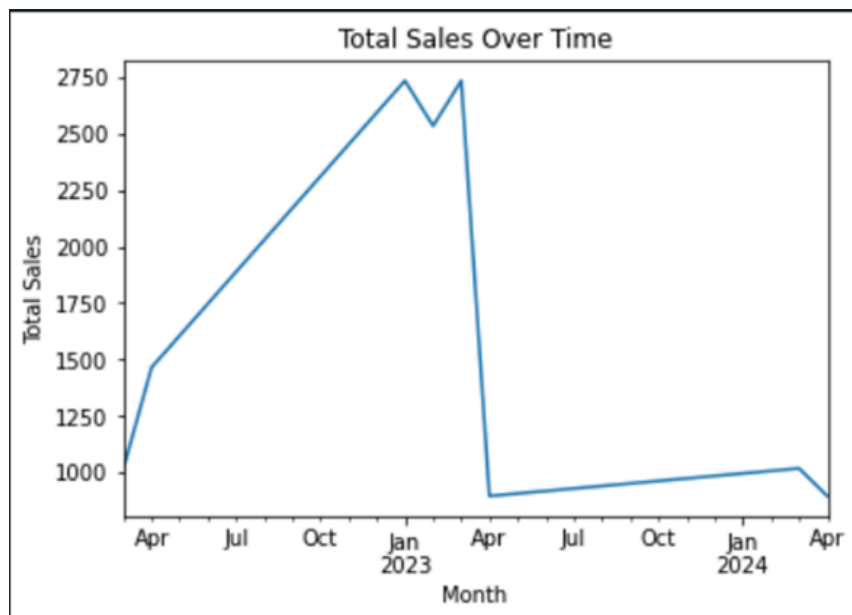
1. Load the Data: Load the CSV file into a Pandas DataFrame. **[2 mark]**
2. Data Cleaning:
 - Ensure there are no missing values. If any are found, fill them with appropriate values. **[3 mark]**
 - Convert the Date column to a datetime object. **[2 mark]**
 - Ensure the Total column correctly represents the product of Quantity and Price. If any discrepancies are found, correct them. **[2 mark]**

Task 1b: Data Visualisation

1. Product Sales Distribution: Create a bar chart showing the total quantity sold for each product. **[2 mark]**



2. Sales Over Time: Create a line plot to show the trend of total sales over the months of 2023. The x-axis should represent the months, and the y-axis should represent the total sales. **[2 mark]**



Python Task 2: Python Database Management [15 marks]

Problem: Managing Sales Data with SQLite. You are required to create a database to store and manage the sales data processed in Part 1.

Task 2a: Database Creation and Data Insertion

1. Create an SQLite database named **SalesDB**. [2 mark]
2. Create a table named **Sales** with appropriate columns to store the data (Date, Product, Quantity, Price, Total). [3 mark]
3. Insert the cleaned and processed **data from Part 1** into the Sales table. Ensure no duplicate entries are added. [4 marks]

Task 2b: Querying the Database

1. Total Sales Calculation: Write a query to calculate the total sales for the year 2023 and display the result. [3 marks]

Data Loaded and Cleaned:

	Date	Product	Quantity	Price	Total	Date_original	Month
0	2023-01-01	Chocolate	5	10.0	50.0	2023-01-01	2023-01
1	2023-01-02	Biscuits	7	20.0	140.0	2023-01-02	2023-01
2	2023-01-03	Sweets	9	15.0	135.0	2023-01-03	2023-01
3	2023-01-04	Water	3	7.5	22.5	2023-01-04	2023-01
4	2023-01-05	Orange Juice	8	12.5	100.0	2023-01-05	2023-01

2. Product Sales Summary: Write a query to list each product and its total quantity sold in 2023. Display the result in descending order of quantity. [3 marks]

```
Data inserted into SalesDB.
Total Sales in 2023: 8900.0
Product Sales Summary in 2023:
Product: Sweets, Total Quantity Sold: 180
Product: Orange Juice, Total Quantity Sold: 152
Product: Biscuits, Total Quantity Sold: 140
Product: Chocolate, Total Quantity Sold: 95
Product: Water, Total Quantity Sold: 60
Product: Unknown, Total Quantity Sold: 13
```

Python Task 3: Basic Neural Network Implementation [12 marks]

In this task, you will define a simple fully connected neural network using PyTorch. You will train the network on synthetic data, evaluate its performance by tracking the loss during training, and visualize the results.

Instructions:

Neural Network Definition: Define a class SimpleNN that extends torch.nn.Module.

The network should have the following layers:

Input Layer: 10 input features.

Hidden Layer 1: A fully connected layer with 5 neurons and ReLU activation.

Hidden Layer 2: A fully connected layer with 3 neurons and ReLU activation.

Output Layer: A fully connected layer with 1 neuron and Sigmoid activation.

Task 3a: Model Initialization [4 marks]

Instantiate the SimpleNN model.

Use `torch.nn.MSELoss()` as the loss function.

Use `torch.optim.SGD` as the optimizer with a learning rate of 0.01.

Task 3b: Data Generation [2 mark]

Generate synthetic data with 100 samples, each having 10 features. The target should be a tensor with 100 values.

Use `torch.randn` to generate the synthetic input data and target values.

Task 3c: Training the Network [3 marks]

Train the network for 20 epochs.

During each epoch, compute the loss, perform backpropagation, and update the model's weights.

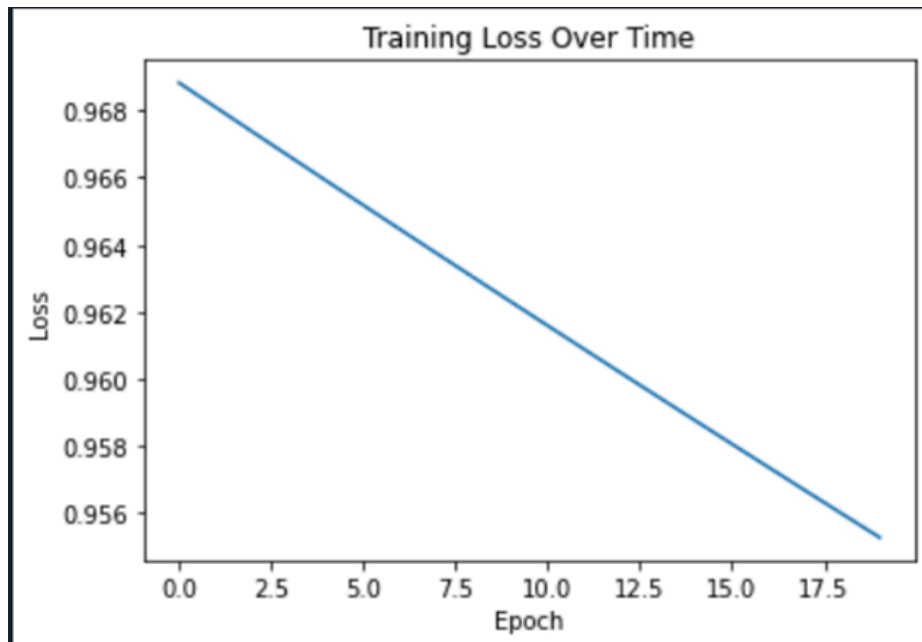
Record the training loss for each epoch.

Task 3d: Loss Visualisation [3 marks]

Plot the training loss over epochs using matplotlib.

Example Output:

Your plot should display the training loss decreasing over time, indicating that the model is learning.


**Python submission**

Ensure your submission file is named exactly as it is supposed to (i.e. `your_name_python_task.py`)

Upload your Python file below.

Maximum file size: 100 MB, maximum number of files: 1

Files



You can drag and drop files here to add them.