CONFIDENTIAL

UTM Fakulti Komputeran
UNIVERSITI TEKNOLOGI MALAYSIA

UNIVERSITI TEKNOLOGI MALAYSIA
FINAL EXAMINATION (PRACTICAL)
SEMESTER 1, 2024/2025

SUBJECT CODE	**:**	**SECJ1013**
SUBJECT NAME	**:**	**PROGRAMMING TECHNIQUE I**
SECTION	**:**	**01,02,03,04,05,06,07,08,09**
TIME	**:**	**2 HOURS 30 MINUTES**
DATE/DAY	**:**	
VENUES	**:**	

INSTRUCTIONS:

- This exam consists of **TWO (2)** questions with a total of 100 marks. You must **answer all questions.**
- This is a **CLOSED-BOOK** practical exam. You will complete the exam by coding on a computer.
- You are provided with reference resources. **ONLY** the provided resources are **allowed** for reference.
- Referencing any other resources, including source codes, websites, or external materials, is **STRICTLY PROHIBITED.**
- The use of any artificial intelligence tools (e.g., ChatGPT, AI-based IDE extensions like Co-Pilot) is **STRICTLY PROHIBITED**.
- Perform all coding **offline** using a C++ IDE such as VS Code, DevCPP, etc.
- You must use the PC provided in the lab. Using your own computer is **NOT ALLOWED.**
- Your program will be screened for **plagiarism detection**. Ensure your work is original.
- **NO technical assistance** will be provided during the exam. This includes issues such as IDE malfunctions or compilation errors. You are responsible for resolving these issues on your own.
- Questions related to the exam question will not be entertained. Please read the questions carefully.
- **COMMENT STATEMENTS** in your submitted program **WILL NOT BE EVALUATED**.

EXAM SETUP:

This exam provides essential resources, including starter code and slides, which are the only materials you may use during the exam.

1. Download the **secj1013_exam2.zip** file from the link available on eLearning.
2. The ZIP file is password-protected, and the password will be given at the start of the exam.
3. Once the password is received, extract the ZIP file. Inside, you will find a folder named **FULLNAME MATRICNUMBER** .
4. Rename this folder using your matric number and full name (spaces are allowed). Example: *NUR AINA BINTI AZMAN A23CS4567*
5. Use the renamed folder as your project directory for all exam questions.
6. Launch your IDE from this folder to begin working.

SUBMISSION:

1. When the exam ends, step away from your computer and wait for your turn to submit.
2. The exam invigilator will manually collect your program files.
3. During submission, copy only the renamed folder (as instructed earlier) to the provided portable storage device using the "**Send to**" option.
4. Leaving the exam room is strictly prohibited until the invigilator grants permission.

QUESTION 1 - ERROR DEBUGGING [35 MARKS]

Notes: Question 1 provides a code file named **program1.cpp** that contains errors. Modify this program to meet the requirements specified in the question.

You are given a C++ program containing **TEN (10) errors**, which may include syntax and/or logical errors. The program is designed to assist hospital staff in managing patient records. It contains **FOUR (4) user-defined functions**, as listed in **Table 1** below:

Table 1: User-defined functions

Function Name	Description
`addPatient`	Add new patient records to the program and store them as **`struct`**.
`displayPatients`	Display the current list of patients stored in the program as **`struct`**.
`loadFromFile`	Read the patient data from **"raw_data.txt"** and store the information in the program as **`struct`**.
`saveToFile`	Write and save the current patient records stored in the program to **"output_data.txt"**.

The main function of the program allows users to select **FIVE (5)** main options, enabling them to add new patient records, display the list of current patients, load patient data from a file, save the current patient list to a file, and exit the program

You are required to debug the program by identifying and fixing all errors, including both syntax and logical errors. Once all errors have been resolved, compile and execute the program to ensure it functions as intended. You are **NOT ALLOWED TO REMOVE** any existing statements from the program. You are only **ALLOWED TO UPDATE** the provided statements or add new statements if necessary to correct the program and achieve the desired functionality.

Important:

- Comment on every line of code where you make updates or modifications.
- Use numbered comments (e.g., **`//Error 1`**, **`//Error 2`**) to track your changes clearly.

Figure 1 shows the information contained in **"raw_data.txt"**. The program should produce the outputs shown in **Figure 2** below. ***Note:*** The values displayed in **bold** represent inputs entered by the user. **Figure 3** shows the expected content in **"output_data.txt"**.

```
1
Azman
34
Normal

2
Janice
21
Fever

3
Edward
45
Flu

4
Monica
17
High Fever
```

Figure 1: Contents of "raw_data.txt"

```
-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 1

Please load the data from the file before proceeding to add a patient.

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 2

No patients data to display.

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
```

```
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 6

Invalid choice! Enter valid choice again.

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 3

Reading patient record from file...
Successfully load the data from file.

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 2

Patient Record:
ID: 1   Name: Azman     Age: 34 Condition: Normal
ID: 2   Name: Janice    Age: 21 Condition: Fever
ID: 3   Name: Edward    Age: 45 Condition: Flu
ID: 4   Name: Monica    Age: 17 Condition: High Fever

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 1

Enter patient ID: 5
Enter patient name: Halim
Enter patient age: 26
Enter patient condition: Fever
Successfully add patient.

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
```

```
4. Save to File
5. Exit
-------------------------
Enter your choice: 2

Patient Record:
ID: 1   Name: Azman     Age: 34 Condition: Normal
ID: 2   Name: Janice    Age: 21 Condition: Fever
ID: 3   Name: Edward    Age: 45 Condition: Flu
ID: 4   Name: Monica    Age: 17 Condition: High Fever
ID: 5   Name: Halim     Age: 26 Condition: Fever

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 1

Enter patient ID: 6
Enter patient name: Kumar
Enter patient age: 29
Enter patient condition: Flu
Successfully add patient.

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 4

Saving patient record to file...
Successfully save the data to the file.

-------------------------
Patient Management Program
Program Menu:
1. Add Patient
2. Display Patients
3. Load from File
4. Save to File
5. Exit
-------------------------
Enter your choice: 5

Program End.
```

Figure 2: Example of program output

```
Patient Record
ID.  Name      Age     Condition
1    Azman     34      Normal
2    Janice    21      Fever
3    Edward    45      Flu
4    Monica    17      High Fever
5    Halim     26      Fever
6    Kumar     29      Flu
```

Figure 3: Expected contents of "output_data.txt"

QUESTION 2 - PROBLEM SOLVING **[65 MARKS]**

Notes: Question 2 provides a code file named **program2.cpp** and input files. Modify this program to address the requirements of the question.

Develop a C++ program to generate a sales report for a grocery mini-market chain with multiple branches. The company collects quarterly sales data for each branch, stored in the following input files:

- **`branches.txt`**: Contains the list of all branch names (one branch name per line).
- **`first.txt`**: Sales data for all branches for the first quarter (Quarter 1).
- **`second.txt`**: Sales data for the second quarter (Quarter 2)
- **`third.txt`**: Sales data for the third quarter (Quarter 3).
- **`fourth.txt`**: Sales data for the fourth quarter (Quarter 4).

Each line in **`branches.txt`** corresponds to a branch, and the same line in the other input files corresponds to that branch's sales for the respective quarter. Your solution must dynamically handle any number of branches as the company may expand its business. Avoid relying on fixed loop limits. For example, consider the sample datasets in Figures 4 and 5.

branches.txt	first.txt	second.txt	third.txt	fourth.txt
Green Valley Market	1200	1300	1250	1350
FreshMart Superstore	1800	1900	1850	2000
CityCenter Grocery	1400	1500	1450	1600

Figure 4: Sample data, Set 1

branches.txt	first.txt	second.txt	third.txt	fourth.txt
Green Valley Market	1200	1300	1250	1350
Green Valley Market	1200	1300	1250	1350
FreshMart Superstore	1800	1900	1850	2000
CityCenter Grocery	1400	1500	1450	1600
UrbanGrocer	1100	1150	1200	1250
EcoFresh Markets	2200	2300	2400	2500

Figure 5: Sample data, Set 4

Based on the given requirements, write a program that accomplishes the following tasks. Include the question number in your program.

1. Declare a structured data type to hold the branch's sales data, consisting of the branch name and its quarterly sales. (5 marks)

2. Read data from input files and store it into an array of the structured type.

(20 marks)

3. Identify the following figures:
    a. **Annual sales for each branch**: Calculate the total sales for each branch across all four quarters and store these values in another array.
    b. **Grand total sales**: Calculate the total annual sales from all branches, representing the overall sales generated by the company across all branches.
    c. **Highest branch sales**: Identify the branch with the highest annual sales and determine the value of those sales.
    d. Use an appropriate loop to calculate and identify the figures mentioned above.

(20 marks)

4. Generate a report as screen output consisting of the following:
    a. Quarter sales for each branch along with their annual sales.
    b. The grand total sales made by the company.
    c. The highest branch sales along with the branch name.
    d. The report should be displayed appropriately including the formats.

(20 marks)

Figures 6 and 7 show sample outputs for the provided datasets. To test your program, manually copy the input files.

```
BRANCH                  QUARTER 1  QUARTER 2  QUARTER 3  QUARTER 4  ANNUAL SALE

Green Valley Market     1200       1300       1250       1350       5100
FreshMart Superstore    1800       1900       1850       2000       7550
CityCenter Grocery      1400       1500       1450       1600       5950


OVERALL REPORT
==============
GRAND TOTAL SALES = 18600
HIGHEST BRANCH SALE = 7550
BRANCH NAME: FreshMart Superstore
```

Figure 6: Example output of the program using the Set 1 dataset

```
BRANCH                QUARTER 1  QUARTER 2  QUARTER 3  QUARTER 4  ANNUAL SALE

Green Valley Market   1200       1300       1250       1350       5100
Green Valley Market   1200       1300       1250       1350       5100
FreshMart Superstore  1800       1900       1850       2000       7550
CityCenter Grocery    1400       1500       1450       1600       5950
UrbanGrocer           1100       1150       1200       1250       4700
EcoFresh Markets      2200       2300       2400       2500       9400


OVERALL REPORT
==============
GRAND TOTAL SALES = 37800
HIGHEST BRANCH SALE = 9400
BRANCH NAME: EcoFresh Markets
```

Figure 7: Example output of the program using the Set 2 dataset