Course Learning Outcomes:
Upon completion of this assignment you should be able to:

CLO1	Explain the essential facts, concepts, principles, strategies and theories relating to Information Technology applications. (C2, PLO1)	Class Test
CLO2	**Demonstrate intellectual independence, logical and analytical thinking skills to develop creative and innovative solutions for a range of Information management and IT problems. (C3, PLO2)**	**Individual Assignment**
CLO3	**Communicate effectively and professionally with peers, clients, superiors and society at large both in written and spoken form. (A3, PLO5)**	**Individual Assignment**

1.0 INDIVIDUAL ASSIGNMENT DESCRIPTION

VACCINE INVENTORY MANAGEMENT SYSTEM

Coronavirus or severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) is a deadly virus that has caused global pandemic. The virus causes respiratory tract infections that can range from mild to lethal. The World Health Organization (WHO) has named the disease caused by coronavirus as coronavirus disease (COVID-19).

As of now, there is no specific medication available nor drugs discovered for treating COVID-19 patients. Thus, hospitals usually provide supportive care to COVID-19 patients as part of their treatment procedure. This includes treatment to relieve symptoms, fluid therapy, oxygen support and prone positioning as needed, and medications or devices to support other affected vital organs.

The COVID-19 vaccination is found to be a safer way to help build protection against the virus and end the pandemic. This has made pharmaceutical companies to race against time and develop vaccine to save the lives of mankind. At present, the vaccination process has started in many countries including Malaysia. The vaccine warehousing and distribution in Malaysia is done by an authorised pharmaceutical company. The types of vaccine planned for people in Malaysia is shown in Table 1 as follows:

Table 1: Types of Vaccine

Name of Vaccine	Vaccine Code	Producing Country	Dosage Required	Population Covered (%)
Pfizer	PF	USA	2	50
Sinovac	SV	China	2	18.8
AstraZeneca	AZ	UK	2	10
Sputnik V	SP	Russia	2	10
CanSinoBio	CS	China	1	10.9

Assume that all these vaccines have been approved by the Malaysian government and the above-mentioned pharmaceutical company needs a Vaccine Inventory Management System to allow its employees to carry out the following:

1. ***Inventory Creation***. The system should provide a feature for the employees to permanently record vaccine details shown in Table 1 into a text file named as *vaccine.txt*. Initial quantity of each vaccine (in millions) also needs to be recorded in this file.

   Note: Initial quantity of vaccine is to be decided by the programmer. The records in the *vaccine.txt* file should be available every time the program is executed.

2. ***Update vaccine quantities***. The system should allow the employees to select a particular vaccine and indicate either received or distributed quantity. In either case, the quantity of the selected vaccine needs to be updated accordingly in the *vaccine.txt* file.

   E.g. Assume that the initial quantity of Pfizer vaccine in *vaccine.txt* file is 1 million. When the company receives a new stock, this quantity has to be added to the existing quantity of 1 million in the *vaccine.txt* file. In the case where the vaccines are distributed to hospitals for vaccination, the distributed quantity has to be subtracted from the quantity available in the *vaccine.txt* file.

   Note: Whenever a vaccine is distributed to hospitals, its code and the quantity distributed need to be recorded into a text file named as *dist.txt*. Each vaccine is expected to be distributed more than once. Hence, while testing the program, there should be at least 10 records created in the *dist.txt* file.

3. ***Search vaccine and its available quantity by using vaccine code***. The system should have a feature for employees to query a particular vaccine's existing quantity from the *vaccine.txt* file using vaccine code.

4. ***Produce a list of all vaccines and their distributed quantities*** . The system should allow the employees to list all distributed vaccines and their accumulated quantities read from the *dist.txt* file.

   Note: The vaccines and their distributed quantities need to be sorted ascendingly (with highest quantity listed first followed by second highest and so on) using Bubble sort before displaying on the screen.

2.0 REQUIREMENTS

i. You are required to carry out extra research for your system and document any logical assumptions you made after the research.

ii. Your program should use symbolic constants where appropriate. Validations need to be included to ensure the accuracy of the system. State any assumptions that you make under each function.

iii. You are required to store all data in *vaccine.txt* and *dist.txt* files only.

iv. You are expected to use control structures, functions, array, pointers, structures, unions and files in your program. Your program must embrace modular programming technique and should be menu-driven. Functions of similar operations can be grouped (or kept alone) and stored as separate C files. Header files are to be stored separately as .h files.

v. You may include any extra features which you may feel relevant and that add value to the system.

vi. There should be no need for graphics (user interface) in your program, as what is being assessed, is your programming skill not the interface design.

vii. You should include the good programming practice such as comments, variable naming conventions and indentation**.**

viii. In a situation where a student:
- ***Failed to attempt the assignment demonstration, overall marks awarded for the assignment will be adjusted to 50% of the overall existing marks.***
- ***Found to be involved in plagiarism, the offence will be dealt in accordance to APU regulations on plagiarism.***

ix. You are required to use portable ANSI C programming language to implement the solution. Use of any other language like C++/Java and etc. is not allowed. Global variable is not allowed.

x. Results of a comprehensive testing is to be included in your document in the form of Input/Output screenshots with sufficient explanation. The tests conducted shall take into consideration of all valid inputs and negative test cases.

3.0 DELIVERABLES

You are required to submit:

i. A softcopy of the program coded in C – submitted in Moodle. The program should include the following:
   - Basic C concepts such as displaying and reading of text, variables, and assignment of values, comments – to explain various parts of the program, etc.
   - Intermediate C concepts such as control structures – selection and iteration control structures, use of arrays – single / double scripted, string.
   - Advanced C concepts such as functions – programmer defined and library functions, pointers, structures, unions, linked list and files.
   - Any other features of C that has not been covered.

ii. A documentation of the system, that incorporates basic documentation standards such as header and footer, page numbering and which includes
   - Cover page
   - Table of contents
   - Introduction and assumptions
   - Design of the program – using pseudocode **and** flowchart – which adheres to the requirements provided above
   - Additional features which have been incorporated in the solution in terms of design and C codes (sample segment of source code from the system created)
   - Sample outputs when the program is executed with some explanation of the outputs / sections of the program
   - Conclusion
   - References – Harvard Name Referencing

iii. Files to be uploaded to Moodle (ONLY FOLLOWING 3 FILES):

1. **Documentation file** (.pdf)
2. **Program / Source files** (.c files), **Header files** (.h files), ***vaccine.txt*** and ***dist.txt*** **files** (all compressed as single .zip or .rar file)
3. **Presentation video file** (refer to Appendix 1 for guidelines on making video presentation)

iv. Submission

- All three files to be uploaded to Moodle by **28th June 2021** latest by 7**.00 p.m.**

4.0 ASSESSMENT CRITERIA

i. Design solution (Pseudocode and Flowchart) 20%

Detailed, logical and application of appropriate idea.

ii. Coding / Implementation 30%

Appropriate application of C concepts (from basic to advance), good solution implemented with validation and met all the requirements with additional features.

iii. Documentation 20%

Overall standard and layout, referencing (Harvard), Input/Output screen capture and assumptions.

iv. Demonstration 20%

Know how to execute and able to trace the system.

v. Question and Answer 10%

Answered the questions based on the assignment submitted during presentation.

5.0PERFORMANCE CRITERIA

Distinction (75% and above)

This grade will be assigned to work which meets all requirements stated in the question. The program runs smoothly when executed. There is clear evidence and application of C concepts up to advanced level. The program solution is unique with excellent coding styles and validation. The program implemented maps completely against the design (pseudocode and flowchart) as seen in the documentation. The design of the solution varies in styles and has unique logic with hardly any errors / omissions. The documentation does not have any missing components. Sample outputs documented have clear explanation. All work is referenced according to Harvard Name Referencing convention. Student must be able to provide excellent explanation of the codes and work done, show additional concepts / new ideas used in the solution, able to answer all questions posed with accurate / logical answers / explanation provided with sound arguments and clear discussion. Overall an excellent piece of work submitted.

Credit (65%-74%)

This grade will be assigned to work which of good standard and meets most of the requirements stated in the question. The program runs smoothly when executed. There is clear evidence and application of C concepts up to at least intermediate level. The program solution is unique with good coding styles and validation. The program implemented maps well against the design (pseudocode and flowchart) as seen in the documentation. The design of the solution varies in styles and has unique logic with minor errors / omissions. The documentation does not have any missing components. Sample outputs documented with some explanation. All work is referenced according to Harvard Name Referencing convention but with some minor errors / omissions. Student must be able to provide good explanation of the codes and work done, answer most questions posed with mostly accurate / logical answers / explanation. Overall a good assignment submitted.

Pass (50%-64%)

This grade will be assigned to work which meets at least half of the basic requirements (approximately 50%) stated in the questions. The program runs smoothly when executed. There is clear evidence and application of C concepts at basic level. The program solution is common with basic coding styles and validation. The program implemented somewhat maps with the design (pseudocode and flowchart) as seen in the documentation. The design of the solution is average in terms of logic and style with some errors / omissions. The documentation has some missing components. Sample outputs documented but without any explanation. Did some referencing but not according to Harvard Name Referencing convention and with some minor errors / omissions. Student must be able to explain some codes and work done and able to answer some questions posed with some accurate / logical answers / explanation. Overall an average piece of work submitted.

Fail (Below 50%)

This grade will be assigned to work which achieved less than half of the requirements stated in the question. The program is able to compile but not able to execute or with major error. The program solution has only basic coding styles with no validation. The program solution has little or no mapping with the design. The design of the solution has major / obvious errors / omissions. The documentation has some missing essential components. No referencing. Student is barely able to explain the codes / work done and answer given on the questions posed but with mostly inaccurate / illogical answers / explanation. Overall a poor piece of work submitted.

APPENDIX 1: Guidelines for Making Video Presentation

Steps before recording:

1. Make sure your camera and mic are in working condition.

2. Make sure the mic volume is adjusted or set to an appropriate level.

3. Make sure the camera is facing you and your face is fully visible throughout the recording session.

4. Open ALL your program files in Visual Studio or Visual Studio Code or Dev C++ or Code Blocks or any other suitable IDE and ensure the font size is not too small.

5. Open the folder where the text file(s) is/are created and keep the folder minimized.

START Recording

1. Introduce yourself (name, TP number, intake, programme of study, level and semester) – max 30 sec

2. Indicate the C and header files you created. Tell what each of them contains. You need to open each of them one-by-one while indicating – max 1 minute.

3. Now, show the C program where the program will start its execution. Explain the program starting from the execution point (from menu function, the input expected, selection of function for each input given to menu function). Note: You **need not** have to explain the code but the inputs to menu function and the selection of functions according to the user input – max 1 minute.

4. Now, show the C file (one-by-one) and explain the internal working of each function. You need to walk through the code (with mouse pointer precisely pointing the line of code you are explaining) - max 6 minutes.

5. Now, compile and run the program (pause the recording if computer takes a lot of time to compile and run) – max 30 sec.

6. Type your first option input to the menu function. Provide all data required. Exit the program. Open the text file created. Display and explain what data written in it. Close the text file (do not minimize).

7. Repeat step 6 for other options one-by-one. Display the changes that takes place in the text file(s). Max 1 minute for each option.

Important: Make sure you share all your screens while recording.

STOP Recording

If required, you may need to use the video editors to edit the unwanted frames out. If needed, use online video editors to fast forward your video to keep the duration short (10 to 15 minutes).

YOU WILL BE REQUIRED TO COME ONLINE (AS THE PER AGREED DATE & TIME) AND ATTEND THE Q&A SESSION (UNLESS ADVISED ACCORDINGLY) TO COMPLETE THE DEMO AND Q&A PROCESS. THE ACCEPTANCE OF YOUR VIDOE PRESENTATION IS BASED ON THE ATTENDANCE TO Q&A SESSION.