Final Examination 2016

Exam Period:	16SP8W1
Exam Code:	1SED1, 1SED2, 1PSE1
Exam:	Diploma Software Engineering;
	MSc Software Engineering & Database Technologies
Module Code:	MCT 609
Module Name:	Fundamentals of Programming
External Examiner:	Dr. John Power
Internal Examiners:	Professor Gerard Lyons
	Dr. James Duggan
	Dr. Sam Redfern
	Mr. Julian Fitzgibbon
	Mr. Chris O'Toole

Instructions:

- Please ensure your name is on all your examination documents.
- Only one examination submission will be accepted from each student.
- No submissions will be accepted after the final submission date.
- All work must be your own. Collaboration is not permitted.
- Include all .c code files as well as one Word document in .doc or .docx format using the naming convention: jbloggs_wk8_exam, where j is your first initial and bloggs is your last name. The file should include a copy of all code, design, development and test documentation as well as a user manual for your system.

Submission Date:

Exam must be submitted by 23:59 on Sunday 6th of March to the module dropbox in WorldClass.

No. of Pages:	5
Department:	Information Technology, NUI, Galway

Part A

Introduction

An ice-cream distributor has requested your help with software to manage its fleet of ice cream delivery trucks. The delivery trucks come in autonomous and manual versions. The majority of the trucks are of the manual variety as the autopilot feature in the autonomous trucks is restricted to the vicinity of the distributor headquarters.

Orders arrive online via the distributor's website, the items are picked in the warehouse and then loaded on to the trucks for shipping to the customer.

The company's management team explain that they need a way to monitor and communicate the delivery status of the orders between a central site and the trucks.

They are interested in having an app developed so that they can assign, manage and monitor the progress of the delivery operations. Eventually the app will automatically use GPS data to match orders with retailers and automatically update the delivery status when the job is completed.

As a first step, they wish to analyse data regarding truck delivery jobs and truck information to see how efficiently delivery is performing. You are required to implement (using the C programming language) a prototype management system for the trucks. Make use of techniques covered in the course material: functions, file handling, dynamic memory allocation, data structures and sorting algorithms.

The Ice-cream Trucks App Management System (ITAMS) will have 2 important files:
 File 1: to record information about the trucks
 File 2: to record information about the deliveries by the trucks.
The truck's type and number are kept on file. If the truck is not autonomous the driver name will also be recorded. Each delivery (job) will have a truck number, delivery number, location zip code, date (and time) and delivery status on file. A truck can have many delivery jobs assigned to it.

ITAMS should provide functions to add trucks and deliveries, view truck information, sort truck information by type, view delivery information by date, sort delivery information by location, and view delivery information by truck.

Create a C program to implement the functions described below. Note that you may add your own utility-style functions in addition to each of the functions required. Note also that the program should be contained in one file only (i.e. no header files or multiple source files). Document your code by including comments throughout, for example, document functions with header comments. (Comments contained just before the function definition).Your system should include a user manual containing instructions on how to run the ITAMS as well as pseudo code or flowcharts for each function.

[70 marks]

Function 1: display_menu
When the application is started, use this function to print a list of options to the user using a numbered list. When the user selects an option, the system performs the required action, and returns to the menu waiting for the user to select another option. Include an option to exit the system.

Your main menu system should resemble the following:

1. Add a Truck
2. Add a Delivery
3. View all Truck information
4. Sort Truck information by driver name and type
5. View Delivery information by date and time
6. Sort Delivery information location zip code
7. Sort Delivery information by date and time due
8. Set Delivery as completed
9. View Delivery information by Truck
10. Save and Exit

s
Function 2: load_data
This function is automatically called when the program is started. It reads truck and delivery information from files. If the files do not exist, notify the user, create the files and display the main menu. Once a file is open, data is read into an **array** of **structs** by dynamically creating sufficient memory for each entry read from the file.

Define a **struct** to represent the information for a truck, containing the following fields:

- **Number (int)**
- **Type (int or enum)**
- **Driver Name (char[])**

Define a **struct** to represent the information for a delivery, containing the following fields:

- **Truck number (int)**
- **Delivery Number (int)**
- **Location Zip Code (char([])**
- **Delivery date/time (appropriate type)**
- **Delivery Completed Flag (Boolean or int)**

Function 3: add_new_truck
When the user selects the "add new truck" menu item, prompt them for the data for the new truck. Append the new truck to the array of current trucks and notify the user that the new record has been added successfully.

Function 4: add_new_delivery
When the user selects the "add new delivery" menu item, prompt them for the new delivery information. Append the new delivery to the array of current deliveries and notify the user that the new record has been added successfully.

Function 5: view_trucks
When the user selects the "view trucks" option, this function prints a list of all truck information on record. Make sure that the truck information is in a report with headers, not simply a list of raw information.

Function 6: sort_trucks
Give the user the option to sort trucks by name and/or type. Store the sorted information and notify the user of the name of the file. Make a copy of the original array of trucks to perform the sorting function. Make sure that the truck information is in a report with headers, not simply a list of raw information.

Function 7: view_delivery_information_time_due
When the user selects the "View delivery information by date and time due" menu option, this function prompts the user for a delivery number. It then displays the delivery status, date/time due *or completed* (in a human-readable format), the truck number and the location zip code. Make sure that the information is in a report with headers, not simply a list of raw information.

Function 8: sort_delivery_information
When the user selects the menu items to "sort delivery information "(either by location zip code or by date time due), this function will sort accordingly and display all deliveries. Store the sorted information and notify the user of the name of the file. Make a copy of the original array of deliveries to perform the sorting function. Make sure that the delivery information is in a report with headers, not simply a list of raw information.

Function 9: set_delivery_completed
When the user selects the "Set delivery as completed" from menu options, prompt them for the delivery number. Then find the corresponding delivery, set the completion flag and update the due date and time variable to the current date and time. Store the updated information in the respective data structures.

Function 10: view_delivery_information_by_truck
When the user selects the "view delivery information by truck" menu option, this function displays each truck and the delivery number of the deliveries that they have in the database. Make sure that the information is in a report with headers, not simply a list of raw information.

Function 11: save_data
This function is called whenever data needs to be saved, for instance when the user chooses to exit the system. Open all data files and write out the information contained in the respective arrays.

Part B - Modification

What other set of requirements could be implemented as part of ITAMS to improve its functionality toward a real-time app? Choose one requirement (e.g. Print a list of recently completed deliveries or print the most urgent unfinished delivery.), provide a definition, flowchart or pseudo code, and a suitable implementation to be included as part of the overall system. Amend the truck and/or delivery structures accordingly, to handle any extra information.

[30 marks]