BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI (RAJ.)

Second Semester 2019-20 CS F111 Computer Programming

Take-home Programming Assignment

Given below are **SIX** assignment questions which have to be solved in teams. EACH TEAM IS REQUIRED TO SOLVE ANY ONE QUESTION. The recommended team size is of 3 students. In case you are unable to get a team-mate, smaller teams will be also allowed. However, ALL team members must belong to the same practical section of the CS F111 course (and this needs to be strictly followed).

IMPORTANT NOTE: Students in even-numbered practical section have to choose an even-numbered question (2, 4, 6), while students in odd-numbered section are required to choose an odd-numbered question (1, 3, 5). There will be no exceptions to this rule!

EVALUATION:

1. You have to provide a C code for the assignment questions. Do not provide a code in any other language (C++ and C# are also not acceptable). Some of the questions require a user-interface. You should create a minimal interface in C itself, and there is no need to use additional frameworks for creating a GUI.
2. You will be evaluated based on the correctness of your code. Your code should also be modular in nature, well-indented and well-commented.
3. Wherever a user-interface is required, you should provide a decent and usable user-interface.
4. If your code doesn't run at all, you should not expect much marks. For a take-home assignment, we expect a running code.
5. The lab instructor may also decide to conduct a viva and question each of the team members regarding their code and their individual contribution to the assignment.
6. All code submissions will be scanned using a professional plagiarism checker software. Plagiarizing your code from the web or from your friends will lead to award of *zero marks* and disciplinary action may be initiated.

SUBMISSION FORMAT: You will submit your solution in a single .c file and its executable (which should be named appropriately, NOT named as a.out) along with a ReadMe.txt file. The ReadMe.txt should clearly list all the team members (Full name and ID no.) at the beginning of the file. It should also contain any instructions required to run your code.

More details on how to share your code with the Instructors will be provided later.

The assignment questions are provided below. Wherever possible, we have provided the expected modules in the code. However, providing all details is not possible. **You are free to make suitable assumptions.**

QUESTION 1

Develop the KBC game using C. The program should have the following features:

1. First of all, primary information, such as the total number of questions, prize money with respect to each question, available life lines and intermediate question levels (described in point no. 4 below) should be displayed on the screen, asking the user to press any key to start the game.
2. When notified by the user, the game should start by displaying one question at a time along with its four options on the screen, prompting the user to select the correct option.
3. When the user selects his/her option, the program should show a message whether the selected option is correct or wrong.
4. If the selected option is correct, the corresponding prize money should be displayed. Otherwise, the prize money counter should be reduced to zero or to the intermediate level (e.g., after question number 3, 8 and 12, as set by you), and displayed on the screen.
5. The user can opt for two life lines, such as “50-50”, upon selection of which, two wrong options should be removed from the screen, or “flip the question” that should replace the entire question with a new one on the screen. These life lines can be availed by the user only once (even at the same time as well). Therefore, the program should take care of this provision.
6. The maximum number of questions a user can play should not exceed 15.

QUESTION 2

Library management system:

Make a simple console-based application for library management that can perform the following:

1. Book management: Search book title by author, Issue no, Title, all Books by the same author, etc.
2. Book transaction: Add the book to check out if available (Print book not available if already issued to a user, with an option for an alert if the book is again in library), delete reservation, return of book
3. Vendor management: Order books to vendors, place request to other libraries
4. Inventory management: Count the number of all books of a particular title
5. User management: Add members to the library, delete members, notification to return books after 15 days of issue
6. A welcome page with user-id and password based login, which also provides a list of newly added books to the library.

QUESTION 3

Multi-user Calendar:

Make a simple console-based multi-user application for a calendar, starting from year 2000, which has the following features:

1. Display month-wise calendar
2. Create Events in the calendar (such as meetings, anniversaries, etc.). Events can be one-time events or monthly events (occurring at the same date every month).
3. Add a functionality to mark an event as personal or public. Personal event is visible only to the user who created it, while a public event is visible to all users.
4. Display all the fixed-date holidays (such as 14th January, 26th January, 15th August, etc.) and Sundays falling in a particular month
5. Given year, month and date, display the day of the week.
6. You are required to implement a multi-user calendar. To view the events created by a user, he has to login. Each user gets to see only their personal events and all public events. You should authenticate users using a user-id password.

QUESTION 4

Develop a console-based text editor using C. Your program should have the following features:

1. Open a new/empty file, write content to it and save it with the desired name
2. Open an existing file, edit its contents and save it
3. Open an existing file and save it with a different name. The original file will continue to exist (similar to “save as” feature)
4. Search for and highlight all occurrences of a pattern/word in a given file (similar to Ctrl+F functionality)
5. Compare two given files and tell whether they are exactly the same or not
6. Delete a file with the given name

QUESTION 5

Grocery Ordering portal:

Design an Online Grocery Ordering portal which allows users to place orders for every day essentials from their home, thus maintaining social distancing which is a requisite for COVID-19. It should have the following modules.

1. Login/Registration for Customer: This module comprises all necessary information of the customer for e.g. name, contact number, home address, email-address.
2. Item Selection: This module contains the information about the items in the Supermart. It has two sub-modules: Perishable and Non- Perishable items. Every perishable item should have an expiry date that should be displayed with the selected item. Together with this, each item should have its expected delivery date which should not exceed the expiry date of item. All the items whose expiry date has reached, those should be automatically removed from the module. This module is also associated with the items that are out of stock and their expected availability. *To give you an overview, the module should work in this manner:* As the customer chooses an item, its cost, availability (whether it is out of stock or not) , expiry date (if the item is perishable) and the expected delivery date should be visible.
3. Payment: The payment can only be made online through debit card. As the customer makes the payment, the card and other details of the person should be verified then only the payment should succeed. (For your reference: Just store a table with details of the customer).
4. Waiting List: This module contains the list of customers who want to shop. Maximum five people are allowed to shop at a time, and the rest of the people who want to shop should be instructed to try after sometime. The priority should be based on first come first serve. (Hint: use <time.h> header file to keep track of the time at which a user logs in)
5. Home delivery: The objective of this module is to deliver the items purchased by the customer at their doorstep. This module should display the expected delivery date of the items purchased by the customer. Keep in mind, the expected date for home delivery should be the at least minimum expiry date of items selected for purchase by the customer. The service should be based on first come first serve.
6. Overall Status: This module is only accessible to Admin. It should state the current status of portal which includes the current addressing customer, number of people in the waiting queue, details of the customers that have been served, information about the items, i.e. quantity, whether an item is out of stock or not, etc.

QUESTION 6

Railway Reservation system for Migrant Labourers:

Features expected in the system:

1. Structure-based implementation of trains (train no., source, destination, seats left) and tickets (PNR, passenger name, train no). The date has been ignored considering the trains operate for a single day. PNR is an auto-increment value starting from 1.
2. Details of all trains – from Delhi to Major cities – should be stored in a text file to be loaded and displayed while accessing the system.
3. Ticket data should be stored in a separate text file and available for reading anytime.
4. Menu-driven enquiry, booking, and cancellation facilities.
5. Enquiry menu will ask for source and destination input, and if the source is any station other than Delhi, show no trains available. Otherwise, display the number of seats left in the train.
6. Booking menu should ask for passenger name, source and destination. Assume booking can be done only for 1 person per PNR. Upon booking, the seats left in that train should be decremented. You should also check whether seats are available before booking the ticket.
7. Cancellation menu should subtract 50% cancellation charges and then display the refund amount. Delete that entry from the ticket data file. The cancelled tickets should be freed up and should be added to the available quota of that particular train.