Developing Technical Software
Assignment 2 (25Marks)
Due date is 11th September 2020 23:59hrs

You may be asked to demonstrate/explain your work to the tutor, if you are absent/unavailable or fail to demonstrate properly, zero marks will be awarded.

Please note, this is an individual task and it will be checked for plagiarism. All the involved parties will be penalised if any plagiarism is found.

Please visit https://goo.gl/hQ87zg for more details.

Instructions

1. This assignment contains 3 questions. Q1 is for 10 marks, Q2 is for 7.5 marks, and Q3 is for 7.5 marks. The total assignment is for 25 marks (See the detailed rubric attached at the bottom of this page).
2. Submit only one document (Single word document).
3. Copy and paste your code into **word document (No need to copy and paste question, question numbers would be enough)**.
4. Copy and paste the screen shot of the output window in the same word document.
5. Use only .doc, .docx extensions – no other format will be accepted for marking.
6. Marks will be given for proper **indentation and comments.**
7. **Assignment Demonstration** is mandatory and randomly selected students should demonstrate their work and may be asked to modify the program.

Qn 1. A major online store specialising in selling games has the following business model.

The company sells many kinds of games through their online platform. Customers place orders online to buy game/s. The gaming company will search for the specific game upon customers request in the order of first come first served basis. The gaming company has the following requirements. (10marks)

1. There are many kinds of games available (at least five). The details of the games are stored in a text file called games.txt
2. The customers' orders are organised in a Queue as a first come first served basis
3. The gaming company should be able to retrieve the data of the last sold games.

Your program should consist of a linked list with the following features

a.) Insertion at the head
b.) Insertion at the tail
c.) Deletion from the head
d.) Deletion form the tail
e.) Deleting specific element from the list

The next part should be implementing a Stack based on the above linked list with the following features

a.) Pop
b.) Push
c.) Top

The next part should be implementing a Queue based on the above linked list with the following features

a.) Enqueue
b.) Dequeue
c.) Top

The program flow is as follows:

The details of the games should be read from games.txt and saved in a linked list. The list of games ordered by customers should be saved in a queue. You should take the order from the beginning of the queue and search for it in the linked list then delete it and put this game in a stack to be able to retrieve for the last sold game.

Sample of games.txt file:

The `main()` function handles all interactions with the **user** and other functions:

- Calls a function named `read_file()` which opens a text file `games.txt`(a sample text file is shown above) for reading and storing names of games from the file to a LinkedList in order of name (insertion should happen in alphabetical order).
- It then repeatedly calls the `menu()` function to display user options, get the user selection returned by the `menu()` function, use a `switch (or if ..else if)` statement to process user request by calling appropriate function(s).

Explanation of options in menu function:
(1) Display the current stock (linked-list) – displays the contents of the LinkedList
(2) Add a Game to stock (linked-list) – inserts a new game to LinkedList
(3) Display next order info – displays the next game in the order-list (first node of the queue)
(4) Display all orders – displays all nodes of the queue
(5) Add order to queue – adds new order to the end of the queue
(6) Process the next order – Processes the first order in the queue. It searches for this game in the linked list deletes (if found) it and puts it into a stack, deletes from the queue as well.
(7) Cancel last order – It cancels the last processed order. It inserts the game (top of the stack) back into LinkedList (Game is not added back into queue)
(8) Display info of last order – displays the information of the last processed order (top of the stack).
(9) Update Game file – updates the games.txt with the remaining games in the list (including the games added in option2).
(10) Quit program

Sample runs of the output is attached with the assignment as a txt file in Canvas.

Qn 2. Write a C program that displays a report of 50 student records by implementing bubble sort, selection sort and insertion sort algorithms (7.5 marks)

a. Declare an array of integers that holds student marks using a random function, fill the array (the maximum marks should be out of 100)
b. Ask the user to enter the choice 1.) For Bubble Sort 2.) For Selection Sort and 3.) for Insertion sort
c. Display all the values after the insertion process
d. Create a report of the students who achieved HD (> 80%), D (> 70%), C (> 60%), P (> 50%), and N (> 50%).
e. Display the list of students based on the user choice 1.) For HD (high-distinction) 2.) For D (Distinction), 3.) For C (Credit), 4.) For P (Pass), 5.) For N(Fail)

Qn 3. Write a C program to perform Binary search and linear search using **recursion** (7.5 marks)

a. Create 10 employee records with associated names and salaries, prompt the user to enter the details.
b. Implement Binary Search and Linear Search by prompting the user to enter the choice 1.) For Binary Search 2.) For Linear Search
c. After the implementation, identify the algorithm that performs well by presenting the differences.

Marking Criteria in the next page.

Criteria			
Question 1: reading from file	**1 Pts** **Perfect**	**0.5 Pts** **Almost Perfect**	**0 Pts** **Partial or No Attempt**
Question 1: A linked list of games with all the five required features	**2 Pts** **All functions are working and screenshot of the output is included**	**1 Pts** **Some functions are working with respective output**	**0.25 Pts** **Partial attempt with no output**
Question 1: The list of games ordered by customers are implemented in Queue with all the three functions	**2.5 Pts** **All functions are working and screenshot of the output is included**	**1 Pts** **Some functions are working with respective output**	**0.25 Pts** **Partial attempt with no output**
Question 1: Stack implementation (take the order from the top of the queue and search for it, then delete it and insert the game) with all the three functions	**2.5 Pts** **All functions are working and screenshot of the output is included**	**1 Pts** **Some functions are working with respective output**	**0 Pts** **Partial attempt with no output**
Question 1: Update file	**1 Pts** **Perfect**	**0.5 Pts** **Almost Perfect**	**0 Pts** **Illogical or no attempt**
Question 1: Overall structure, Program flow, indentation and appropriate comments etc	**1 Pts** **Perfect**	**0.5 Pts** **Almost Perfect**	**0 Pts** **Not satisfactory**
Question 2: Declare an array of integers that holds student marks using a random function, and fill the array (the maximum marks should be out of 100)	**1 Pts** **Perfect**	**0.5 Pts** **Partial Attempt**	**0 Pts** **No attempt/wrong.**
Question 2: Implementation of Bubble Sort	**1 Pts** **Perfect**	**0.5 Pts** **Partial Attempt**	**0 Pts** **No attempt/totally wrong.**
Question 2: Implementation of Selection Sort and Insertion sort	**2.5 Pts** **Perfect**	**1 Pts** **Partial Attempt**	**0 Pts** **No attempt/totally wrong.**
Question 2: Display values after the sorting	**1 Pts** **Perfect**	**0.5 Pts** **Partial Attempt**	**0 Pts** **No attempt/totally wrong.**

Question 2: Detailed report who achieved HD (> 80%), D (> 70%), C (> 60%), P (> 50%), and N (< 50%).	**1 Pts** **Perfect**	**0.5 Pts** **Partial Attempt**	**0 Pts** **No attempt/totally wrong.**
Question 2: Display the list of students based on the user choice 1.) For HD (high-distinction) 2.) For D (Distinction), 3.) For C (Credit), 4.) For P (Pass), 5.) For N(Fail)	**1 Pts** **Perfect**	**0.5 Pts** **Partial Attempt**	**0 Pts** **No attempt/totally wrong.**
Question 3: Create 10 employee records with associated names and salaries and prompt the user to enter the details	**1 Pts** **Perfect**	**0.25 Pts** **Almost Perfect**	**0 Pts** **No attempt/totally wrong.**
Question 3: Implement Binary Search (recursive)	**2.5 Pts** **Perfect**	**1.5 Pts** **Almost Perfect**	**0 Pts** **No attempt/totally wrong.**
Question 3: Implement Linear Search (recursive)	**2 Pts** **Perfect**	**1 Pts** **Almost Perfect**	**0 Pts** **No attempt/totally wrong.**
Question 3: At least two screenshots for each search by searching for any employees salary	**1 Pts** **Full marks**	**0 Pts** **Almost Perfect**	**0 Pts** **No attempt/totally wrong.**
Question 3: At least three solid differences with a proper explanation for each of the search	**1 Pts** **Full marks**	**0 Pts** **Almost Perfect**	**0 Pts** **No attempt/totally wrong.**
Total points: 25			

Good Luck!