CSE 211 Data Structures

Lab Sheet

In this lab sheet, you are given 4 different programming exercises that you are supposed to implement in the C++ language during the first 4 weeks of the term. For each programming exercise, you are expected to design the algorithm(s) in the pseudocode language first and then start coding. While your work will not be assessed officially, you are highly encouraged to put your maximum effort on the exercises so as to get good marks from the exams and perform the homework and projects easily.

During the lab sessions, you are expected to follow the following rules:

1. You may not walk around the class room
2. You may not talk to your friends
3. You may ask questions to the lab assistants only
4. You are free to use any online/offline software programming editors that accepts C++
5. You may use the lecture slides uploaded in COADSYS

1. A college offers a course that prepares students for the state licensing exam for real estate brokers. Last year, ten of the students who completed this course took the exam. The college wants to know how well its students did on the exam. You have been asked to write a C++ program to summarize the results. You have been given a list of these 10 students. Next to each name is written a 1 if the student passed the exam or a 2 if the student failed.

   Your program should analyze the results of the exam as follows:

   1. Input each test result (i.e., a 1 or a 2). Display the prompting message "Enter result" each time the program requests another test result.

   2. Count the number of test results of each type.

   3. Display a summary of the test results indicating the number of students who passed and the number who failed.

   4. If more than eight students passed the exam, print the message "Raise tuition."

2. Write a C++ program to check whether any given String is palindrome or not.

   For this task, do the following:

   1. Write a function that determines whether any given string is palindrome or not.
   2. Call your function in the *main* function so as to test whether it produces the correct results or not for any string provided by the user.

3. A small airline has just hired you as a software specialist to program their automated reservations system.
   On your first day, you have been asked to write a C++ program to assign seats on each flight of their only plane. The capacity of the plane is 10 seats. Your program should constantly display the following menu of alternatives:
   Please type 1 for first class
   Please type 2 for economy class
   If the person types 1, then your program should assign a seat in the first class section (seats 1-5). If the person types 2, then your program should assign a seat in the economy section (seats 6-10). After the seat assignment the program should print the boarding pass that includes the class type and seat number on the screen. If the person types a number other than 1 or 2 print "Not a valid class type" on the screen and return to the menu.
   Use a one-dimensional array to represent the seating chart of the plane. Initialize all the elements of the array to 0 to indicate that all seats are empty. As each seat is assigned, set the corresponding element of the array to 1, to indicate that the seat is no longer available.
   Your program should, of course, never assign a seat that has already been assigned.
   When the first class section is full, check if there are empty seats in the economy class and ask the person if it's acceptable to be placed in the economy section. If the answer is yes make the appropriate seat assignment and print the boarding pass on the screen (and vice versa). If the answer is no, then print the message "Next flight leaves in 3 hours.

4. You are expected to write a C++ program for the pancake game. Your program prompts the user to enter the number of pancakes eaten by 7 people.

   Mert 3

   Onur 4

   Ali 5

   ...

   Next, your program will perform the following operations over these data.

   • to identify the person who ate the most and the person who ate the least

   • to compare any two people based on the number of pancakes they ate

   • to find out the name of the person who ate a specific number of pancakes

   • to sort the people in ascending and descending order based on the number of cakes they ate