CSL201: DATA STRUCTURES LAB

LAB INSTRUCTIONS

EVALUATION CRITERIA

Mark Distribution

Total Marks : 150

CIE (Continuous Internal Evaluation) Marks : 75

ESE (End Semester Exam) Marks : 75

ESE Duration : 3 hours

Continuous Internal Evaluation Pattern:

Attendance : 15 marks

Continuous Evaluation in Lab : 30 marks

Continuous Assessment Test (Internal Examination) : 15 marks

Viva-voce : 15 marks

Continuous Evaluation (30 Marks)

Continuous evaluation in the lab will be on the basis of experiments completed and also daily performance in the lab.

Mark Division will be :

Lab Cycle - 60 marks + Daily Performance Evaluation - 30 marks (Total 90 marks which will be converted to 30 during internal mark calculation)

Lab Cycle: The list of experiments for this lab consists of 20 programs , to be completed within the duration of this semester. These will be divided into two lab cycles (10 programs per cycle). Each lab cycle will be evaluated out of 30 marks (3 marks for each program) . Total 60 marks for two lab cycles.

Program evaluation for the lab cycle experiments will be done by considering the algorithm, implementation strategies, output formatting and the student's ability to successfully explain the algorithm/program. **Of the 3 marks for each program, 1 mark will be for successfully**

completing the program, 1 mark for neatly completing and presenting the rough record with the algorithm and 1 mark for properly explaining the code/algorithm.

Daily Performance Evaluation: During each lab session, the student will be given a practice program related to the programs that they have completed so far in the lab cycle. Students are expected to solve this problem, formulate an algorithm and to implement the same using C, within the duration of that lab session. **This will be evaluated out of 5 marks (Algorithm - 2 marks + Implementation - 2marks + Output -1 mark)** .The average of all such evaluations will be converted to 30 and considered as the Daily Performance Evaluation .

Internal Examination Pattern

The marks will be distributed as Algorithm 30 marks, Program 20 marks, Output 20 marks and Viva 30 marks. Total 100 marks which will be converted out of 15 while calculating Internal Evaluation marks.

Viva Voce

There will be two general vivas, one for each lab cycle. In addition there will be surpise short vivas in the lab. Average marks of all these, converted to 15 would be considered as the total mark for viva voce while calculating Internal marks.

Lab Record:

All Students attending the Data Structures Lab should have a Rough Record. Use a 180 or 200 page book with neat margins for lab record. Every experiment conducted in the lab should be noted in the record. For every experiment in the record the right hand page should contain Experiment Heading, Experiment Number, Date of Experiment, Aim of Experiment, Data Structure used, algorithm and Result of Experiment. The left hand page should contain sample output obtained for a set of input.

For every program completed in the lab cycle, students should submit a report (in pdf format) and the executable code in the Moodle ClassRoom (assignment window will be provided). Report should contain Experiment Heading, Experiment Number, Date of Experiment, Aim of Experiment, Data Structure used, algorithm , Program , Sample Output and Result of Experiment. A single document should be maintained for the reports of all programs in the lab cycle. This will be printed and used as your fair record during the final external lab exam.

End Semester Examination Pattern

The marks will be distributed as Algorithm 30 marks, Program 20 marks, Output 20 marks and Viva 30 marks. Total 100 marks will be converted out of 75 for End Semester Examination.

Operating System to Use in Lab : Linux
Compiler/Software to Use in Lab : gcc
Programming Language to Use in Lab : Ansi C

LAB RULES AND CONDUCT

1. All programs and reports must be submitted on specified deadlines. Marks will be reduced for late submissions.

2. Since the number of available lab sessions per student is less (due to the Covid Protocol), more than one program may have to be submitted on a particular lab day and the three hour lab session may not be enough to complete it all. So students are required to plan their lab activities accordingly.

3. Students can complete the experiments at home using personal laptops and bring it for evaluation on the specified date during lab hours. Students who are not able to do this due to unavailability of devices can use the lab facility at Department, during free hours or after 4pm to complete the experiments.

4. Regular lab hours will be used to evaluate the programs done by students and also for daily performance evaluation using practice problems (which are not part of the lab cycle) or viva.

5. While writing the algorithm, it shouldn't contain any programming language constructs (if-then-else, for or while loops, etc) . It should be a step by step detailing of the program logic, independent of any programming language.

6. Programs should be properly intended and commented. Program names and variable/constant names should be sensible and easily recognisable.

7. Output should be properly formatted and easily understandable

8. Mobile phones should not be used in the lab without permission. They should be kept in silent mode.

LIST OF EXPERIMENTS

CYCLE 1 (30 marks)

(Each program carries 3 marks)

1. Write a menu driven C Program to implement a Stack using arrays with the operations:
   a. Pushing elements to the Stack.
   b. Popping elements from the Stack
   c. Display the contents of the Stack after each operation

2. Write a menu driven C Program ,to do the following operations using a stack datastructure:

   a) Convert an infix expression to a postfix expression
   b) Evaluate the postfix expression

3. Write a program to read two polynomials and store them in an array. Calculate the sum of the two polynomials and display the first polynomial, second polynomial and the resultant polynomial.

4. Write a C program to enter two matrices in normal form . Do the following operations, implemented as separate functions:
   a) Convert two matrices to tuple form and display it.
   b) Find the transposes of the two matrices represented in tuple form and display them in tuple form.
   c) Find the sum of the two matrices in tuple form and display the sum in tuple form.

5. Write a menu driven C Program to implement a Queue using arrays with the following operations:
   a. Insert elements to the Queue.
   b. Delete elements from the Queue.
   c. Display the contents of the Queue after each operation.

6. Write a menu driven C Program to implement a circular queue using arrays with the following operations:
   a. Insert an element to the queue.
   b. Delete an element from the queue.
   c. Display the contents of the queue after each operation.

7. Write a menu driven C Program to implement a Double-Ended Queue (DEQUEUE) with the following operations:
   a. Insert elements to the Front of the queue.
   b. Insert elements to the Rear of the queue

c. Delete elements from the Front of the queue.
d. Delete elements from the Rear of the queue.
e. Display the queue after each operation

8. Write a menu driven C Program to implement a Priority Queue using arrays with the following operations:
   a. Insert elements to the Priority Queue.
   b. Delete elements from the Priority Queue.
   c. Display the contents of the Priority Queue after each operation

9. Write a C program to create a structure Employee with fields EmpId, Name and Salary. Name should contain first name, middle name and last name. Store the details of n employees, dynamically allocating memory for the same. Write a function to implement Linear Search to search for a particular employee, given the EmpId.

10. Write a C program to read string data stored in a file. Sort the strings in alphabetical order using Bubble Sort. Implement Binary Search to search for a given string. Implement sort and search routines as separate functions.

.

CYCLE 2 (30 marks)

(Each program carries 3 marks)

1. Write a menu driven C program for performing the following operations on a Linked List:
   a. Display
   b. Insert at Beginning
   c. Insert at End
   d. Insert at a specified Position
   e. Delete from Beginning
   f. Delete from End
   g. Delete from a specified Position

2. Write a C program to read two polynomials and store them using a linked list. Do the following operations on them.
   a. Find the product of two polynomials and store the result using a linked list. Display the resultant polynomial.
   b. Calculate the sum of the two polynomials and display the first polynomial, second polynomial and the resultant polynomial.

3. Write a menu driven C Program to implement a stack using linked list with the following operations:
   a. Push elements to the queue.

   b.Pop elements from the queue.

   c.Display the queue after each operation

4. Write a menu driven C Program to implement a queue using linked list with the following operations:
   a.Insert elements to the queue.
   b.Delete elements from the queue.
   c.Display the queue after each operation

5. Write a menu driven C Program to create a binary tree using linked list with the following operations
   a. Insert a new node
   b. Inorder traversal
   c. Preorder traversal
   d. Postorder traversal
   e. Delete a node

6. Write a C Program to represent any given graph and to do the following operations:
   a. Compute adjacency list and adjacency matrix.

b. Perform a depth first search
c. Perform a breadth first search

7. Write a program to read numerical data stored in a file. Write a menu driven C program to implement the following sorting algorithms for sorting the numbers in ascending order. Implement each algorithm as a separate function.

   a. Insertion Sort

   b. Selection Sort

   c. Heap Sort

   d. Merge Sort

   e. Quick Sort

8. Write a C program to implement a Hash table using the Chaining method. Let the size of the hash table be 10 so that the index varies from 0 to 9.

9. Write a C program to implement a Hash table that uses Linear Probing for collision resolution.

10. Write a C program to implement to simulate a basic memory allocator and garbage collector using doubly linked list