UNIVERSITY OF PETROLEUM & ENERGY STUDIES

College of Engineering Studies

Dehradun

COURSE PLAN

Programme : B. Tech. CSE spl. in Cyber Security and Forensics

Course : Programming and Data Structures Lab

Subject Code : CSEG 1102

No. of credits : 1

Semester : I

Session : July 2018 – December 2018

Batch : 2018 - 2022

Prepared by : Ms. Richa Choudhary

Email : r.choudhary@ddn.upes.ac.in

Approved By

Faculty HOD

UPES Campus
"Energy Acres"
P.O. Bidholi, Via Prem Nagar, Dehradun

Tel: +91-135-2770137
Fax: +91 135- 27760904
Website: www.upes.ac.in

COURSE PLAN

A. PRE-REQUISITES

Basic knowledge about computers would be beneficial.

B. COURSE OBJECTIVES

The Course "Programming and Data Structures lab" would help the learners to enhance their programming skills through C programming language. Data structures are the foundation for computer science; therefore, the course introduces the concept of data handling such as queue, stacks and linked list, including a study of the different searching and sorting algorithms.

C. PEDAGOGY

Solution to the problems should be designed (algorithm/flow-chart/pseudocode) and tested. After obtaining a successful design, the solution to the problem is implemented using C language. Students are evaluated based on Performance (via efficient design, implementation) and record keeping, and preparation of students (via viva-voce/quiz).

D. COURSE OUTLINE

Expt. No.	Big Ideas/ Topics	Modality
1	Basics of Problem Solving and Programming using C language (Sequential logic) in Linux environment.	F2F
2	Conditional Branching	F2F/e-lab
3	Working with Iterations	F2F/e-lab
4	Recursion, Functions and Pointers.	F2F/e-lab
5	1D arrays and handling Strings in C language	F2F/e-lab
6	2D arrays and Searching in Arrays	F2F/e-lab
7	Structure, Union and Dynamic Memory Allocation	F2F/e-lab
8	Stack and its Applications	F2F/e-lab
9	Queue and its Applications	F2F/e-lab
10	Linked List and its Applications	F2F/e-lab
11	Sorting and File Handling	F2F/e-lab

E. COURSE COMPLETION PLAN

No. of Labs Sessions planned	No. of Viva planned
11	2

One Session = 120 minutes

F. EVALUATION & GRADING

Students are continuously evaluated through the proceedings of the course based on the following:

1) Performance & Record (50%)
2) Viva Voce or Quiz (50%)

F.1 Performance & Record: WEIGHTAGE - 50%

A minimum of 11 experiments should be handled as F2F in the lab and online through e-labs or virtual labs. This component is obtained through continuous evaluation after performing the experiments. Each experiment requires several activities (Lab & Practice) to be performed for better understanding of C programming and data structures concepts by the students. More details of these lab and practice activities are available in Section G.

Experiment Evaluation (10 marks per experiment)	
Algorithm Design	3 Marks
Coding Syntax	2 Marks
Execution / Bug Finding	2 Marks
Records (submitted **before** the very next turn.)	3 Marks

F.2 Viva Voce or Quiz Examination: WEIGHTAGE - 50%

The preparation of the students would be evaluated based on viva-voce or quiz or e-lab test conducted in periodic schedules.

It is mandatory for the students to attend the above said evaluations. Students who do not attend will lose their marks. Continuous Internal Assessment Record Sheet will be displayed at the end of the semester.

GRADING:

The overall marks obtained at the end of the semester comprising the above two mentioned shall be converted to a grade. Student(s), who have met the qualifying criteria of individual practical subject but not met qualifying criteria of SGPA, will not be allowed to re-appear for improvement. Students, who wish to re-appear in the practical subject, shall be required to pay the prescribed fee per subject as notified by the University. The student with Grade "F" only will be eligible to *repeat continuous evaluation* of that respective practical subject (s) during summer vacation (June-July).

Grade shall be awarded on the performance of the student(s). The Grade will be capped as per the rules mentioned in student Bulletin. There will be no capping of SGPA for the students re-appeared for Practical Subject. All Other rules and regulations such as requirement of passing, etc. will remain same as mentioned in rules & regulations.

G. DETAILED SESSION PLAN

Activities listed under 'Lab activities' should be performed F2F (face-to-face) in the allocated laboratories (either compiled in lab machines or compiled online using e-labs). In addition to the 'Lab activities', the instructor should also select the necessary number of 'Practice activities' to be performed using e-labs other than laboratory hours.

EXPERIMENT – 1

Title: Basics of Problem Solving and Programming in C language using Sequential Logic in a Linux Environment.

Objective: Understanding problem solving technique with algorithm, flowchart and C language standard I/O & fundamental data types.

List of Lab Activities:

1. Study of Linux Commands
    a. Working with Directories: mkdir, rmdir, dir, pwd, cd
    b. Handling Files: vi, gedit, more, cp, mv, rm
2. Write algorithm, draw flow chart, prepare test cases and test them. Write C program (with sequential flow logic), compile and test/execute the code using Linux C compiler with the same test cases.
    a. Print your mailing address in the following form.
        First line: Name
        Second line: Door no, Street
        Third line: city, Pin code
    b. Given 2 numbers. Calculate sum, difference, multiplication and division.

List of Practice Activities:

Write algorithm and C program (with sequential flow logic), compile and test/execute the code using Linux C compiler with suitable test cases.

1. Accept an amount, time period and rate of interest and calculate the simple interest.
2. Obtain value of two numbers in variables. Swap the values of variables, with a third variable and later without using third variable.
3. Convert temperature from Celsius to Fahrenheit and Kelvin.
4. Print the given days in years-month-days format. E.g. 373 days = 1 year, 1 month, 1 day

EXPERIMENT – 2

Title: Conditional Branching in C Language

Objective: To understand conditional logic of execution; applying conditional branching structures in C (if, if-else, if-else-if ladder, nested-if, switch-case).

List of Lab Activities:

Write algorithm, draw flow chart, prepare test cases and test them. Write C program, compile and test/execute the code using Linux C compiler with the same test cases.

1. Find the biggest of 3 numbers.
2. Check whether a given year is leap year or not.

List of Practice Activities:

Write algorithm and C program, compile and test/execute the code in Linux C compiler with suitable test cases.

1. Find whether a given number is even or odd.
2. Find if the given number is positive, negative or zero.
3. Check if the given number is divisible by 2 and 3 or not.
4. Check whether a given character is an alphabet, digit or special character.
5. Check if a given character is a vowel or consonant using Switch-Case statement.
6. Find the roots of a quadratic equation.

EXPERIMENT – 3

Title: Iterative Constructs in C language

Objective: To understand the concepts of Looping with Iterations; applying while, do-while and for constructs.

List of Lab Activities:

Write algorithm, draw flow chart, prepare test cases and test them. Write C program, compile and test/execute the code using Linux C compiler with the same test cases.

1. Multiple two given numbers without using the arithmetic binary multiplication operator using for loop.
2. Find the sum of digits of a number using while loop.
3. Reverse the digits of the number using do-while loop.
4. Print the given pattern using nested for loop.

```
11111
22222
33333
44444
55555
```

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Check if the given number is a Palindrome or not.
2. Print the numbers which are divisible by 3 and 5 from the first 'n' natural numbers.
3. Generate all the prime numbers between 1 and n, where n is a value supplied by the user.
4. Generate all the Armstrong numbers between 1 and n, where n is a value supplied by the user? [Hint: Armstrong number is a number, whose sum of cube of every digit results in the same number, ex. $153=1^3+5^3+3^3=153$]
5. Generate the first 'n' terms of a Fibonacci sequence. [Hint: A Fibonacci sequence is defined as follows: the first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence.]
6. Print the given patterns as per user choice.

a.

```
1
22
333
4444
55555
```

b.

```
    *
   ***
  *****
 *******
*********
```

c.

```
1
121
12321
1234321
123454321
1234321
12321
121
1
```

7. Given value of n, find the sum of the series 1+ 1/2 + 1/3 + 1/4 + 1/5 + …+ 1/n.

EXPERIMENT – 4

Title: Working with Recursive Looping, Functions and Pointers.

Objective: To understand the concept of Functions and Looping with Recursion.

List of Lab Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Write a function which gets a number 'n' as argument, computes and displays the factorial of 'n' using recursion.
2. Function main() gets two numbers from the user and calls two functions defined as follows:
   a. The function "void swap1(int *, int*)" swaps the two numbers using bitwise operator and displays them.
   b. The function "int swap2(int, int)" swaps two numbers using arithmetic binary operator plus (+) and minus (-).

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Find the maximum and minimum between two given numbers using functions.
2. Check if a given number is prime number, Armstrong number and perfect number using functions.
3. Write a function to reverse any number using recursion.

EXPERIMENT – 5

Title: 1-dimensional arrays & Strings in C Language

Objective: To understand the concept of Arrays, Array notation and representation, manipulating array elements, using multidimensional arrays, arrays of unknown or varying size.

List of Lab Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Find sum of all array elements using recursion.
2. Create an array 'a1' with 'n' elements. Insert an element in i^{th} position of 'a1' and also delete an element from j^{th} position of 'a1'.
3. Convert uppercase string to lowercase using for loop.

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Find out the largest and smallest number in a given array.
2. Create an array 'a1' with 'n' elements.
   a. Copy all elements of 'a1' into another array 'a2' using pointers. Display the contents of both the arrays using pointers.
   b. Merge the contents of 'a1' and 'a2' into a new array 'a3'.
3. Find the total number of alphabets, digits or special characters in a string.
4. Find whether the entered string is palindrome or not.
5. Count the number of words in a string.

EXPERIMENT – 6

Title: Two Dimensional Arrays and Searching

Objective: To understand the concept of 2D Arrays and searching techniques.

List of Lab Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Find the sum of rows and columns of matrix of given order (row x column).
2. Find the product of two matrices using pointers.
3. Store ‘n’ numbers (integers or real) in an array. Conduct a linear search for a given number and report success or failure in the form of a suitable message.

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Find the transpose of a matrix.
2. Find the Inverse of a matrix.
3. Find if the given matrix of order (m x n) is a Sparse matrix or not. [Assume that a matrix can become a sparse matrix if more than half the total number of its elements have the value zero]
4. Store ‘n’ numbers (integers or real) in an array in ascending or descending order. Conduct a binary search for a given number and report success or failure in the form of a suitable message.

EXPERIMENT – 7

Title: Structure, Union and Dynamic Memory Allocation.

Objective: To understand the concept of structure and union, experiment on dynamic memory allocation.

List of Lab Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Design a structure ‘product’ to store the details of the product purchased like product name, price per unit, number of quantities purchased, and amount spent. Get the name, price per unit, and number of quantities of the product purchased. Calculate the amount spent on the product and then display all the details of the procured product using structure pointers.
2. Design a union ‘product’ to store the details of the product purchased like product name, price per unit, number of quantities purchased, and amount spent. Get the name, price per unit, and number of quantities of the product purchased. Calculate the amount spent on the product and then display all the details of the procured product.
3. Design a structure ‘subject’ to store the details of the subject like subject name and subject code. Using structure pointer allocate memory for the structure dynamically so as to obtain details of ‘n’ subjects using for loop.

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Design a structure ‘student_record’ to store student details like name, SAP ID, enrollment number, date of registration and data of birth. The element date of joining is defined using another

structure ‘date’ to store date details like day, month, and year. Get data of ‘n’ students and then print the entered values [Hint: Use concept of Nested structures and Array of Structures].

2. Add two complex numbers by passing structure to a function as argument.

EXPERIMENT – 8

Title: **Stack Data Structure and its Applications**

Objective: Understand the operations and applications of Stack along with the implementation using arrays.

List of Lab Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Using array and functions implement Stack and its operations like insert, delete, and display.
2. Reverse a string using stack.

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Using array and functions implement two Stacks and its operations (insert, delete, display).
2. Convert infix to postfix expression using stack and array.
3. Evaluate postfix expression using stack and array.

EXPERIMENT – 9

Title: **Queue Data Structure and its Applications**

Objective: Understand the operations and applications of Queue along with the implementation using arrays.

List of Lab Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Using array and functions implement Queue data structure and its operations like insert, delete, and display.
2. Check whether the string is palindrome or not using array and Queue.

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Using array and functions implement Circular Queue data structure and its operations like insert, delete, and display.
2. Using array and functions implement Priority Queue and its operations (insert, delete, display).
3. Using array and functions implement Double Ended Queue (Input Restricted Deque and Output Restricted Deque) and its operations (insert, delete, display).
4. Using array and functions implement a Stack using Queues.

EXPERIMENT – 10

Title: Link List Data Structure and its Applications

Objective: To experiment the concept of pointers, structure and dynamic memory allocation to realize linked list and its application.

List of Lab Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Implement single Linked List data structure and its operations like insert and delete in the beginning/end and nth position of the list, and display the items stored in the linked list.
2. Using single linked list and functions implement Stack and its operations like insert, delete, and display.

List of Practice Activities:

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Add two polynomials using Linked List.
2. Using single linked list and functions implement Queue and its operations like insert, delete, and display.
3. Implement Circular Linked List and its operations.
4. Implement Doubly Linked List and its operations.

EXPERIMENT – 11

Title: Sorting and File Handling in C

Objective: To understand the concept of Files, Operations on Files, Creation of data bases using files and various I/O facilities in C.

List of Lab Activities

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Merge the contents of two files to another.
2. Create a file with a list of 'n' integers. Read the numbers from the file into an array and sort them using Insertion Sort algorithm.

List of Lab Activities

Write algorithm and C program, compile, execute and test the code using Linux C compiler with suitable test cases.

1. Find a word in a text file and replace it with another word.
2. Input numbers from file using fscanf() and write even, odd and prime numbers to different files.
3. Store 'n' numbers (integers or real) in an array. Sort the elements using Selection Sort algorithm.

H. SUGGESTED READING

Text Books

1. Yashwant Kanetkar, "Let Us C".
2. Schaum Series, "Data Structure".
3. Ellis Horowitz and Sartaz Sahni, "Data Structure using C".
4. P.K. Sinha and P.K. Sinha, "Computer Fundamentals".

Reference Books

1. E Balaguruswamy, "Programming in ANSI C".

Note: Also refer to the Web-links/Resources in Blackboard and NPTEL videos.

I. GUIDELINES

Cell Phones and other Electronic Communication Devices: Cell phones and other electronic communication devices (such as Blackberries/Laptops) MUST be turned off during the lab session.

E-Mail and online learning tool: Each student in the class should have UPES e-mail id and a password to access the Blackboard regularly. The best way to arrange meetings with faculty or is by email and prior appointment. Various research papers/reference material will be mailed/uploaded on online learning platform time to time.

Attendance: Students are required to have **minimum attendance of 75%** in the subject.

Passing criterion: Student has to secure minimum 40% marks of the "highest marks scored by the student for the subject" in the total marks in order to pass in that paper.