COMP103: Structured Programming

Year: I
Semester: I

Credit: 2

Course Description:
This course introduces the fundamental concepts of procedural programming in C. Topics include data types, control structures, functions, arrays, etc. This course also focuses on the development of problem solving skills using programs.

Contents

1. **Introduction to Computer Systems [2 hours]**
   - Brief history of computation
   - Architecture and Peripherals
2. **Introduction to Software Systems [2 hours]**
   - System Software
   - Application Software
   - Programming Languages
3. **Introduction to Software Life Cycle [2 hours]**
   - Problem solving and software engineering a brief introduction (SDLC)
   - Algorithms and Flowchart
4. **Fundamentals of C [2 hours]**
   - The C Character Set
   - Identifiers and Keywords
   - Data Types
   - Variables, Constants, Declarations Statements
5. **Operators and Expressions [3 hours]**
   - Introduction
   - Arithmetic Operators
   - Unary Operators
   - Relational and Logical Operators
   - Assignment Operators
   - Conditional Operators
   - Operator Precedence
6. **Decision Control Statements [3 hours]**
   - Introduction
   - The if-else Construct
   - The nested if-else Construct
   - The else-if ladder Construct
   - The switch Construct
7. **Loop Control Statements [3 hours]**
   - Introduction
   - The while Construct
   - The do-while Construct
   - The for Construct

8. **Functions [3 hours]**
    - Anatomy of a Function (Defining a function, accessing a function)
    - Function Prototype
    - Recursion (Introduction and some programs)
9. **Program Structure [1 hours]**
    - Storage Classes
    - Automatic, External and Static Variables
10. **Arrays [3 hours]**
    - Introduction
    - Processing an Array
    - Passing Arrays to Functions
    - Multidimensional Array
11. **Structures [3 hours]**
    - Understanding C's Structures
    - Referencing a Structure Member
    - Using Structure with Function calls
    - Arrays of Structures
    - Understanding Unions
12. **Pointers [3 hours]**
    - Introduction
    - Passing Pointers to Functions
    - Pointers and One Dimensional Array
    - Pointers to Structures
    - Dynamic Memory Allocation
    - Operations on Pointers

Total Lecture Hours: 75 (15 Weeks * 5 hours of theory and Practical)

Reference Books:

1. Byron s. Gottfried, Theory and Problems of Programming with C, 2/e, McGraw-Hill.

Exam: Internal: 50 and Final: 50

Assignment #1: Operators, Expression

```
Question 1
What is meant by operator precedence? Illustrate with an
example.
Question 2
What is meant by associativity? Illustrate with an example.
Question 3
What are library functions? Why are they important?
```

Assignment #2: Branching, looping

Question 1

Compare the use of the *if-else* statement with the use of conditional operator.

Question 2

What is the purpose of the *default* keyword?

Question 3

What is the purpose of *do-while* statement? How does it differ from the*while* statement?

Question 4

What is the purpose of the *break* statement?

Assignment #3: functions

Question 1

State at least three advantages of making your program modular using functions.

Question 2

What is Recursion? What advantages is there in its use?

Question 3

What three types of errors do function-prototypes help prevent?

Question 4

What is meant by the scope of a variable within a program?

Assignment #4:Array, Structures, pointers

Question 1

In what way does an array differ from an ordinary variable?

Question 2

Differentiate between pass by value and pass by reference with an example.

Question 3

What is a pointer variable? What are the advantages of using a pointer?

Question 4

What is the difference between array and pointer variable? In what way are they similar?

Question 5

What is a structure? How is it different from union?

COMP103: Structured Programming

Lab Manual
Department of Computer Science & Engineering
2009

COMP 103: Structured Programming
Lab Assignment

Week #1
Introducing Turbo C++ Compiler and its environment / Linux Environment
Week #2
A Sample C program. **Some problems related formatted strings.**
Week #3: Operators and expressions
1. **Write a program to convert centigrade to Fahrenheit. [F = 9/5 * C + 32]** 2. **Write a program that calculates the area of a circle and circumference.** 3. **Write a program that calculates the area of a triangle.** 4. **Write a program that reads the marks in each subject and calculates the percentage.**
Week #4: Conditional Statements
1. **Write a program that reads a number and identifies whether the given number is even or odd.** 2. **Write a program to find the largest number among two numbers** 3. **Write a program to read the mark of a subject and prints the equivalent grade.**
Week #5: Loop
1. **Write a program to read a sentence and counts the total number of character (excluding space) using while loop.** 2. **Write a program to generate Fibonacci number using do while loop.** 3. **Write a program to read number and identifies whether the given number is a prime number or not.**
Try this at HOME
1. **Write a program to identify whether the given number is a perfect number or not. 28 is a perfect number.** 2. **Write a program to calculate the factorial of a given number.**
Week #6: Function
1. **Write a program to identify whether the given number is a perfect number or not using a function. 28 is a perfect number.** 2. **Write a program to evaluate GCD of two given integers. Use function that returns GCD.**
Week #7: Recursion
1. **Write a recursive program to find the factorial of a given number.** 2. **Write a recursive program to find a GCD of two numbers.** 3. **Write a recursive program to find the sum of n natural numbers.**
Week #8: Array

1. **Write a C program to store N numbers in a one dimensional array and calculate its average with the help of the function.**
2. **Write a C program to convert a binary number to decimal with the help of the function. [*`int todecimal(char bits[20], int length)`*]☒ here *bits* is the character array to represent bits of binary numbers and *length* is the number of bits in the binary number.**

Week #9: Array

1. **Write a program to evaluate transpose of n by n matrix with the help of function. [*`int [][20] transpose(int matrix[][20],n`*]☒ here *matrix is* the matrix is theto be transformed and *n* is the dimension of *matrix*. The function should return transpose of the matrix.**
2. **Write a C program for matrix addition with the help of function [*`int [][20] add(int a[][20], int b[][20], int n, int m)`*] ☒ Here *a* and*b* are matrix to be added and *n* and *m* are dimension of *a* and *b*. the function should return *m* by *n* matrix containing the addition data.**
3. **Write a C program to determine determinant of a square matrix with the help of function [*`int determinant(int [][]a, n)`*]☒ here ☐a☐ is the matrix whose determinant is to be found and n is dimension of square matrix.**

Week#10: Sorting

1. **Write a program to arrange the numbers (array) in ascending order using bubble sort.**

Week #11: Structure

1. **Write a program that defines a structure called STUDENT with suitable attributes and reads the data for 5 students. Your program should display the records in ascending order according to the name of the students.**
2. **Consider a plane graph. Write a program that uses function to return a distance between given point and the origin.**

Week #12: Pointers

1. **Write a program that swaps two variables. Use function and pointers.**
2. **Solve the matrix multiplication problem using pointers.**