Course Code	Course name	L	T	P	C
CSEG1041	Programming in C	3	0	2	5
Total Units to be Covered: 7		Total Contact Hours: 105			
Prerequisite(s):		Syllabus version: 1.0			

Course Objectives

1. Introduce students to the basic principles and concepts of programming.
2. Develop students' ability to solve programming problems by applying the fundamental concepts of C programming.
3. Demonstrate students how to design and develop structured programs using modular programming techniques.
4. Enable students to apply their C programming skills to develop small-scale applications.

Course Outcomes

On completion of this course, the students will be able to

CO1. Demonstrate a high level of proficiency in writing correct and efficient C code.

CO2. Acquire the skills to debug and troubleshoot C programs efficiently.

CO3. Expertise in designing structured programs using modular programming techniques.

CO4. Understand file and memory management techniques.

CO5. Acquire the ability to apply their C programming skills to develop practical applications.

CO-PO Mapping

Program Outcomes / Course Outcomes	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO 1	-	-	-	-	3	-	-	-	1	-	-	-	-	-	-
CO 2	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-
CO 3	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-
CO4	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-
CO5	-	-	2	-	2	-	-	-	2	-	-	2	-	-	-

Average	-	-	.4	-	2	-	-	-	.6	-	-	.4	-	-	-

1 – Weakly Mapped (Low) **2 – Moderately Mapped (Medium)**

3 – Strongly Mapped (High) **"_" means there is no correlation**

Syllabus

Syllabus

Unit I: Introduction to Computing **7 Lecture Hours**

Basic computer organization, Evolution of programming languages, Data representation and storage, Basics of programming environment: editors, debuggers, translators, basics of program design and execution, Algorithms, Pseudocode and Flowcharts.

Unit II: C Programming Fundamentals **7 Lecture Hours**

Data types and type conversion, variables (declaration vs definition, local vs global), keywords, header files, structure of a C program. Operators: types of operators (arithmetic, relational, logical, bit-wise, increment/decrement, assignment, sizeof, ternary), operator precedence and associativity. Conditional statements: if, else, switch-case, break, continue, goto, label. Loops: for, while and do-while.

Unit III: Array and Function **9 Lecture Hours**

Array, Multi-dimensional arrays, Strings, Function, Pass and Return by value, Pass and Return by Reference, Recursion, Scope Rules.

Unit IV: Structures and Pointers **8 Lecture Hours**

Structure, typedef, Union, Enum, Bit-Fields, Pointer, Pointer to Arrays, Pointer Arrays, Pointer to Pointers, Address Arithmetic, Pointer to Structures, Pointer to functions, Bit-wise operator.

Unit V: File handling, Memory management **7 Lecture Hours**

Data Organization, File Operations. Dynamic Memory Management: Malloc(), Calloc(), Realloc() and Free (), Garbage Collection.

Unit VI: Preprocessor, Macro, Static and Shared Library **7 Lecture Hours**

Preprocessor & Directives, Macro, Macro vs Functions, C standard library: stdio.h, ctype.h, stdlib.h, assert.h, stdarg.h, time.h etc., Compilation of a C Program, Static Library, Shared Library.

Total lecture Hours 45

References*

Textbooks	1. B. W. Kernighan, and D. M. Ritchie, "The C programming language", 2nd Edition, Prentice Hall, 1988. 2. P. J. Deitel, and H. M. Deitel, "C: How to program", 8th Edition, Pearson Education, 2015.
Reference books	1. B. S. S. Gottfried, "Schaum's Outline of Programming with C", 2nd Edition, McGraw-Hill, 1996. 2. P. V. D. Linden, "Expert C Programming-Deep C Secrets", Pearson Education, 1994.
Web Resources	
Journals	
MOOCs, online courses	

Modes of Evaluation: Quiz/Assignment/ presentation/ extempore/ Written Examination etc.

Examination Scheme

Components	IA	MID SEM	End Sem	Total
Weightage (%)	50	20	30	100

Programming in C Lab

List of Experiments

Experiment 1: Installation, Environment Setup and starting with C language

1. Write a C program to print " Hello World"
2. Write a C Program to print the address in multiple lines (new line).
3. Write a program that prompts the user to enter their name and age.
4. Write a C program to add two numbers, take number from user.

Experiment 2: Operators

1. WAP a C program to calculate the area and perimeter of a rectangle based on its length and width.
2. WAP a C program to Convert temperature from Celsius to Fahrenheit using the formula: F = (C * 9/5) + 32.

Experiment 3.1: Conditional Statements

1. WAP to take check if the triangle is valid or not. If the validity is established, do check if the triangle is isosceles, equilateral, right angle, or scalene. Take sides of the triangle as input from a user.
2. WAP to compute the BMI Index of the person and print the BMI values as per the following ranges. You can use the following formula to compute BMI= weight(kgs)/Height(Mts)*Height(Mts).

	BMI
Starvation	<15
Anorexic	15.1 to 17.5
Underweight	17.6 to 18.5
Ideal	18.6 to 24.9
Overweight	25 to 25.9
Obese	30 to 39.9
Morbidity Obese	40.0 above

3. WAP to check if three points (x1,y1), (x2,y2) and (x3,y3) are collinear or not.
4. According to the gregorian calendar, it was Monday on the date 01/01/01. If Any year is input through the keyboard write a program to find out what is the day on 1st January of this year.
5. WAP using ternary operator, the user should input the length and breadth of a rectangle, one has to find out which rectangle has the highest perimeter. The minimum number of rectangles should be three.

Experiment 3.2: Loops

1. WAP to enter numbers till the user wants. At the end, it should display the count of positive, negative, and Zeroes entered.
2. WAP to print the multiplication table of the number entered by the user. It should be in the correct formatting. Num * 1 = Num
3. WAP to generate the following set of output.

a.

1

2 3

4 5 6

b.

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

4. The population of a town is 100000. The population has increased steadily at the rate of 10% per year for the last 10 years. Write a program to determine the population at the end of each year in the last decade.
5. Ramanujan Number is the smallest number that can be expressed as the sum of two cubes in two different ways. WAP to print all such numbers up to a reasonable limit.

   Example of Ramanujan number: 1729

   12^3 + 1^3 and 10^3 + 9^3. for a number L=20(that is limit)

Experiment 4: Variable and Scope of Variable

1. Declare a global variable outside all functions and use it inside various functions to understand its accessibility.
2. Declare a local variable inside a function and try to access it outside the function. Compare this with accessing the global variable from within the function.
3. Declare variables within different code blocks (enclosed by curly braces) and test their accessibility within and outside those blocks.
4. Declare a static local variable inside a function. Observe how its value persists across function calls.

Experiment 5: Array

1. WAP to read a list of integers and store it in a single dimensional array. Write a C program to print the second largest integer in a list of integers.
2. WAP to read a list of integers and store it in a single dimensional array. Write a C program to count and display positive, negative, odd, and even numbers in an array.
3. WAP to read a list of integers and store it in a single dimensional array. Write a C program to find the frequency of a particular number in a list of integers.
4. WAP that reads two matrices A (m x n) and B (p x q) and computes the product A and B. Read matrix A and matrix B in row major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the

matrices for multiplication. Report appropriate message in case of incompatibility.

Experiment 6: Functions

1. Develop a recursive and non-recursive function FACT(num) to find the factorial of a number, n!, defined by FACT(n) = 1, if n = 0. Otherwise, FACT(n) = n * FACT(n-1). Using this function, write a C program to compute the binomial coefficient. Tabulate the results for different values of n and r with suitable messages.
2. Develop a recursive function GCD (num1, num2) that accepts two integer arguments. Write a C program that invokes this function to find the greatest common divisor of two given integers.
3. Develop a recursive function FIBO (num) that accepts an integer argument. Write a C program that invokes this function to generate the Fibonacci sequence up to num.
4. Develop a C function ISPRIME (num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given ranges.
5. Develop a function REVERSE (str) that accepts a string argument. Write a C program that invokes this function to find the reverse of a given string.

Experiment 7: Structures and Union

1. Write a C program that uses functions to perform the following operations:
    a. Reading a complex number.
    b. Writing a complex number.
    c. Addition and subtraction of two complex numbers

Note: represent complex number using a structure.

2. Write a C program to compute the monthly pay of 100 employees using each employee_s name, basic pay. The DA is computed as 52% of the basic pay. Gross-salary (basic pay + DA). Print the employees name and gross salary.
3. Create a Book structure containing book_id, title, author name and price. Write a C program to pass a structure as a function argument and print the book details.

4. Create a union containing 6 strings: name, home_address, hostel_address, city, state and zip. Write a C program to display your present address.

Experiment 8: Pointers

1. Declare different types of pointers (int, float, char) and initialize them with the addresses of variables. Print the values of both the pointers and the variables they point to.
2. Perform pointer arithmetic (increment and decrement) on pointers of different data types. Observe how the memory addresses change and the effects on data access.
3. Write a function that accepts pointers as parameters. Pass variables by reference using pointers and modify their values within the function.

Experiment 9: File Handling in C

1. Write a program to create a new file and write text into it.
2. Open an existing file and read its content character by character, and then close the file.
3. Open a file, read its content line by line, and display each line on the console.

Experiment 10: Dynamic Memory Allocation

1. Write a program to create a simple linked list in C using pointer and structure.
2. Write a program to insert item in middle of the linked list.

Experiment 11: Bitwise Operator

1. Write a program to apply bitwise OR, AND and NOT operators on bit level.
2. Write a program to apply left shift and right shift operator.

Experiment 12: Preprocessor and Directives in C

1. Write a program to define some constant variable in preprocessor.
2. Write a program to define a function in directives.

Experiment 13: Macros in C

1. Write a program to define multiple macro to perform arithmetic functions.

Experiment 14: Static Library in C

1. Write a program to create a static library for performing arithmetic functions.
2. Write a program to use static library in other program.

Experiment 15: Shared Library in C

1. Write a program to create a shared library for performing arithmetic functions.
2. Write a program to use shared library in other program.

Total Lab hours 60

References*

Textbooks	1. B. W. Kernighan, and D. M. Ritchie, "The C programming language", 2nd Edition, Prentice Hall, 1988. 2. P. J. Deitel, and H. M. Deitel, "C: How to program", 8th Edition, Pearson Education, 2015.
Reference books	1. B. S. S. Gottfried, "Schaum's Outline of Programming with C", 2nd Edition, McGraw-Hill, 1996. 2. P. V. D. Linden, "Expert C Programming-Deep C Secrets", Pearson Education, 1994.
Web Resources	
Journals	
MOOCs, online courses	

Modes of Evaluation: Quiz/Assignment/ presentation/ extempore/ Written Examination etc.

Examination Scheme: Continuous Assessment

Components	Quiz & Viva	Performance & Lab Report
Weightage (%)	50	50