GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Engineering

Level: Degree

Branch: ALL

Course / Subject Code: BE01000121

Course / Subject Name: Programming for Problem Solving

w. e. f. Academic Year:	2024-25
Semester:	1st Year
Category of the Course:	ESC

Prerequisite:	• Basic Computer Skills • Logical Thinking and Problem-Solving Abilities • Understanding of mathematical logic and problem-solving strategies
Rationale:	The "Programming for Problem Solving" course is a foundational skill that significantly enhances an engineer’s problem-solving abilities, career prospects, and capacity for innovation. Integrating programming into the curriculum ensures that engineering programs prepare students to meet the challenges of the modern technological landscape and contribute effectively to their respective fields.

Course Outcome:

After Completion of the Course, Student will able to:

No	Course Outcomes	RBT Level
01	Identify fundamental programming constructs such as variables, data types, operators, expressions, control structures, functions and basic data structures	Remember
02	Explain the principles of programming and software development, including the structure and operation of algorithms, flowcharts, and pseudocode.	Understand
03	Develop simple programs using appropriate data structures and standard libraries.	Apply
04	Apply programming constructs such as loops, conditional statements, and functions to solve basic engineering problems.	Apply
05	Debug and troubleshoot programming errors by systematically testing and refining code.	Analyze

GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Engineering

Level: Degree

Branch: ALL

Course / Subject Code: BE01000121

Course / Subject Name: Programming for Problem Solving

Teaching and Examination Scheme:

Teaching Scheme(in Hours)			Total Credits L+T+(PR/2)	Assessment Pattern and Marks				Total Marks
				Theory		Tutorial / Practical		
L	T	PR	C	ESE (E)	PA / CA (M)	PA/CA (I)	ESE (V)	
02	00	04	04	70	30	20	30	150

Course Content:

Unit No.	Content	No. of Hours	% of Weightage
1.	**Introduction to Programming** • **Basics of Computers**: Overview of computer systems, hardware, andsoftware. • **Introduction to Programming Languages**: Types of programminglanguages, language translators (compilers, interpreters). • **Flowcharts and Algorithms**: Basics of creating flowcharts and writingalgorithms for problem-solving.	03	10
2.	**Basics of C Programming** • **Structure of a Program**: Basic syntax, semantics, and structure of C programs. • **Data Types and Variables:** Primitive data types, variables, and constants. • **Operators:** Arithmetic, relational, logical, bitwise, and assignment operators.	03	10
3.	**Control Structures** • Conditional Statements: if, if-else, nested if-else, switch-case. • Loops: for, while, do-while loops. • Break and Continue Statements: Usage in loop control.	03	10
4.	**Functions and Modular Programming** • Defining Functions: Syntax, return types, and parameter passing. • Library Functions: Standard library functions and header files (for C). • Recursion: Basic concepts and examples.	03	10
5.	**Arrays and Strings** • Arrays: One-dimensional and multi-dimensional arrays, array operations.	03	10

	• Strings: String handling functions, basic string operations.		
6.	**Pointers** • Pointers: Basics of pointers, pointer arithmetic, pointers and arrays, pointers tofunctions (for C).	03	10
7.	**Structures and Unions** • Structures: Defining and using structures, array of structures, pointer tostructures. • Unions: Basics and usage.	03	10
8.	**File Handling** • File Operations: Opening, closing, reading, writing files. • File Pointers: File pointers and basic file operations (for C).	03	10
9.	**Dynamic Memory Allocation** • Memory Allocation: malloc, calloc, realloc, and free functions. Usage and examples.	03	10
10.	**Debugging and Testing** • Debugging Techniques: Common debugging practices, use of debugging tools. • Testing: Writing test cases, unit testing.	03	10
	Total	**30**	**100**

Suggested Specification Table with Marks (Theory):

Distribution of Theory Marks (in %)					
R Level	**U Level**	**A Level**	**N Level**	**E Level**	**C Level**
20	20	40	20	00	00

Where R: Remember; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create (as perRevisedBloom's Taxonomy)

References/Suggested Learning Resources:

(a) Books:

1. C Programming Language by Brian W. Kernighan and Dennis M. Ritchie**,** Latest Edition (for C programming)
2. Let Us C by Yashavant Kanetkar, Latest Edition (for C programming)
3. Problem Solving and Program Design in C by Jeri R. Hanly and Elliot B. Koffman**,** Latest Edition **(**for Cprogramming)

(b) Open-source software and website:

1. OnlineGDB : GDB online Debugger | Compiler - Code, Compile, Run, Debug online C, C++ (onlinegdb.com)
2. Compiler Explorer (Godbolt): Compiler Explorer (godbolt.org)
3. JDoodle: JDoodle - Free online cloud coding platform IDE to practice, teach and learn programming

Suggested Course Practical List:

1. Write a program that performs basic arithmetic operations (addition, subtraction, multiplication, and division)and demonstrates the use of different data types.
2. Create a program that uses if, else, and switch statements to implement a simple menu-driven application. Useloops (for, while, and do-while) to repeat tasks.
3. Develop a program that calculates the factorial of a number using both iterative and recursive functions.
4. Write a program to perform various operations on arrays (e.g., sorting, searching) and strings (e.g.,concatenation, comparison).
5. Implement a program that uses pointers to create and manipulate dynamic arrays, demonstrating the use of malloc, calloc, realloc, and free.
6. Design a student record system using structures that store and display information such as name, roll number,and grades.
7. Write a program to read from and write to files, such as creating a simple text editor that performs basic fileoperations.
8. Implement a singly linked list with operations like insertion, deletion, and traversal.
9. Develop programs to simulate stack operations (push, pop, peek) and queue operations (enqueue, dequeue)using arrays and linked lists.
10. Provide students with a program containing intentional errors and inefficiencies. Have them use debuggingtools (like gdb) to find and fix the errors and optimize the code for better performance.

List of Laboratory/Learning Resources Required:

1. Computer System/Laptop with latest configurations.
2. Internet Connectivity

Suggested Project List:

1. **Library Management System** with the Objectives of
    - Practice basic C syntax and operations,
    - Use control structures effectively,
    - Implement functions and modular programming,
    - Manipulate arrays and strings,
    - Handle file operations for data storage and retrieval

2. **Institute Hall Management System,** Project Objectives:
    - To manage the allocation of halls for different events.
    - To maintain records of bookings, cancellations, and availability of halls.
    - To provide an interactive user interface for managing hall reservations.

Suggested Activities for Students:

- To enhance the programming skills of students in a C Programming and Problem-Solving course, the instructor can assign various activities that progressively build their understanding and ability to applyconcepts.

****