W/D/TY

Final Assessment Test – April 2025

VIT
Vellore Institute of Technology

Course: BCSE102L - Structured and Object-Oriented Programming
Class NBR(s): 1460/1471/1482/1486/1500/1505/1522/ 1556/1564/1579/1590/1596
Slot: B1
Time: Three Hours
Max. Marks: 100

- KEEPING MOBILE PHONE/ANY ELECTRONIC GADGETS, EVEN IN 'OFF' POSITION IS TREATED AS EXAM MALPRACTICE
- DON'T WRITE ANYTHING ON THE QUESTION PAPER

Answer ALL Questions
(10 X 10 = 100 Marks)

1. Explain the different sections of a C program in detail, highlighting their purpose and significance. Provide a structured breakdown of each section with examples illustrating their role in program execution.

2. a) Develop a C program that takes marks of five subjects as input and calculates the percentage. Using the if-else ladder, assign and display the corresponding grade based on the following criteria: **[5]**
   - 90-100: A
   - 80-89: B
   - 70-79: C
   - 60-69: D
   - <60: Fail

   b) Write a C program to count and display the Amstrong numbers present in between 1 to 1000. **[5]**

3. Assume you are developing a program to manage a list of integer numbers entered by the user. Initially, the user specifies how many numbers they want to enter, and memory is allocated dynamically. Later, they decide to add more numbers, so the program should expand the memory accordingly. Implement a C program that:
   - Allocates memory for n integer numbers using **malloc()**.
   - Accepts numbers from the user and calculates their sum and average.
   - Uses **realloc()** to expand the array when the user wants to add more numbers.
   - Displays the updated list and releases memory using **free()**.

   Explain how dynamic memory allocation helps in efficiently managing memory when handling variable-sized data.

4. Compare and contrast the Structure and Union in C concerning memory allocation, size, member accessibility, modification impact, and use cases. Explain how they differ in handling data storage and efficiency. Provide real-world scenarios where each would be preferred. Additionally, implement a C program that demonstrates both structure and union usage by defining a common data type (e.g., a Person as structure and a Person as a union) containing common attributes like Aadhaar ID, Name, Age, and Annual Income. Show how memory is allocated differently in both cases and explain the impact when modifying values.

5. a) Modern programming paradigms emphasize code reusability, modularity, and data security. Discuss how the Object-Oriented Programming methodologies achieve these goals by organizing data and behavior efficiently. Provide real-world examples of how these principles are applied and compare them with traditional procedural programming approaches. **[5]**

b) Compare and contrast Call-by-Address and Call-by-Reference in C++ programs concerning memory usage, variable modification, and function efficiency. Additionally, write a C++ program for swapping two numbers that demonstrates both methods by implementation. [5]

6. Write a C++ program to build a university management system of various individuals, such as students and faculty members, that need to be tracked with relevant details. Multilevel inheritance should be used to establish a hierarchy to keep the system efficient and well-organized. At the base level, a **Person class** should store general attributes such as **ID** and **age**, which are common to all individuals. Extending from this, a **Student** class should inherit from **Person** and include student-specific attributes like **course and Year_addmission**. Further, an **Exam class**, derived from **Student**, should incorporate **marks for three subjects** and provide a function to calculate total and average marks. Also, create functions within the specific classes to allow the user to input and display details. In main () function create instance for multiple students and get the inputs, compute their total and average scores, and identify the student with the highest average marks.

7. a) Write a C++ program that overloads the increment (++) operator for complex numbers. In the program, a Complex class should be defined with real and imaginary attributes, and pre-increment operators should be implemented (++obj). [7]

b) Compare and contrast virtual and pure virtual functions in C++ concerning their purpose, implementation, and impact on class design. [3]

8. Explain in detail the concept of Generic Programming in C++, discussing its importance in achieving code reusability, type safety, and flexibility. How do *function templates and class* templates contribute to generic programming?
To illustrate your explanation, write a C++ program that:

- Implements a generic function to find and return the maximum of two values. The input for these functions is either integers, floating-point numbers, or characters.
- Defines a generic class for a simple calculator that supports addition, subtraction, multiplication, and division for different data types, such as int, float, and double.

9.a) Assume you are developing an Employee Record Management System for a company where each employee has specific details, including Employee ID (integer), DOB (date format), Date of Joining (DOJ in date format), and Salary (float). To maintain structured data, create a user-defined Date structure to store the day, month, and year for both the DOB and DOJ fields. Then, define an Employee structure that includes all employee attributes along with the Date structure for DOB and DOJ. Also, write a C program to create the above Employee structure and do the following operations in main function.

- The program should first accept details for n employees from the user, storing them in an array of structures.
- After input, the program should display all employee records.
- Additionally, implement a function to find and display the details of the employee with the highest salary.

OR

9.b) Write a C program to perform the following operations on an array of n integers, and each operation should be implemented as a separate user-defined function. The program should prompt the user to enter the number of elements, accept the values, and display the results of all operations.

- Sort the array in ascending order.
- Search for a specific element entered by the user using linear search.
- After sorting, swap the largest and smallest elements in the sorted array.

10.a) i) Analyze and evaluate how a derived class can access private members of its base class when using private visibility mode for inheritance. Justify your reasoning with appropriate examples and explain the implications of such access. [5]

ii) Compare and contrast function overloading and function overriding by analyzing their key differences, similarities, and use cases. [5]

OR

10.b) i) John is developing a C++ program for an online store that manages product details, such as name and price. He wants to initialize product objects in different ways based on the available information. Sometimes, a product may be created without any details, in which case default values ("Unknown" for the name and 0.0 for the price) should be assigned. In other cases, only the product name might be provided, with the price initialized to its default value (0.0). Lastly, both the product name and price may need to be specified. To achieve this flexibility, he decides to use constructor overloading. Write a C++ program with a Product class that implements constructor overloading to handle these three scenarios. Additionally, create a main function to test all three constructors. [7]

ii) *Modify the following program to count the number of objects created in the class.* [3]

```
class parameter
{
  private:
  int len;
 public:
 parameter (int i)
 {
    len=i;
 }
int value()
{
  return len*len;
}
};

int main()
{
      parameter p1(10);
      parameter p2(20);
      cout<<p1.value();
}
```