VIT

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

24BLC1198.

Final Assessment Test(FAT) - Apr/May 2025

Programme	B.Tech.	Semester	Winter Semester 2024-25
Course Code	BCSE102L	Faculty Name	Prof. Sivakami R
Course Title	Structured and Object-Oriented Programming	Slot	G2
		Class Nbr	CH2024250502428
Time	3 hours	Max. Marks	100

Instructions To Candidates

- Write only your registration number in the designated box on the question paper. Writing anything elsewhere on the question paper will be considered a violation.
- Only non-programmable calculator without storage is permitted

Course Outcomes

CO1: Understand different programming language constructs and decision-making statements; manipulate data as a group.

CO2: Recognize the application of the modular programming approach; create user-defined data types and idealize the role of pointers.

CO3: Comprehend various elements of object-oriented programming paradigm; propose solutions through inheritance and polymorphism; identify the appropriate data structure for the given problem and devise the solution using generic programming techniques.

Section - I
Answer all Questions (7 × 10 Marks)

01. Ashwin, a meteorologist, is analyzing the weekly temperature data for multiple cities. He categorizes the temperatures into Cold, Moderate, and Hot based on predefined thresholds. Temperature Ranges: Cold: Below 15°C (i.e., temperatures less than 15°C) . Moderate: Between 15°C and 30°C, inclusive (i.e., 15°C <= temperature <= 30°C) Hot: Above 30°C (i.e., temperatures greater than 30°C). Receive the temperatures of each city day-wise and find the category of it. Based on the category compute the sum of temperatures and also keep track of the counts of temperature for each category of temperatures. Then compute the average temperatures for each category and display the averages with its corresponding count of temperatures for those categories. Update the temperature records by replacing each value with the average of its category and display the same. Implement the problem in C.

[10] (CO1/K3)

02. A word is categorized into one of the following 4 categories:
    - **Good:** If all the vowels in the word appear in alphabetical order (a, e, i, o, u).
    - **Worst:** If all the vowels appear in reverse alphabetical order (u, o, i, e, a).
    - **Bad:** If the vowels do not follow either of these orders or if any of the vowel appears more than once
    - **Neutral:** If the word contains no vowels, it is classified as Neutral.

    Develop a C program with the following functions and requirements:

    main()
    - Reads N number of words. (Assume all alphabets are in lowercase).
    - Calls word_category() function

    word_category()

    Accepts each word as its argument, identifies and returns its category to the main function

[10] (CO2/K4)

03. A software company is developing a multi-dimensional search algorithm that works on jagged arrays (arrays where each row can have a different number of elements). Assume the following jagged array assignments -

```
sizes[ROWS] = {3, 4, 2};
int row1[] = {1, 2, 3};
int row2[] = {4, 5, 6, 7};
int row3[] = {8, 9};
```

Scenario:
A researcher wants to implement a recursive function to search for a given element in a jagged array.
The function should:

Recursively traverse each row and column. Return the position (row, column) if the element is found, otherwise return (-1, -1).

Develop the Implementation in C.

[10] (CO2/K3)

04. LIC allows policyholders to select a payment mode for their insurance premium:
    - Bank Transfer → Requires Bank Name & Account Number
    - Credit Card → Requires Card Number & Expiry Date
    - UPI Payment → Requires UPI ID

Since different payment modes have different input requirements, function pointers can be used for flexibility.

Problem Statement:
Implement a C program using function pointers that:

a) Defines separate functions to take input for Bank Transfer, Credit Card, and UPI Payment details. (6 Marks)
b) Uses a function pointer to call the appropriate payment function based on user choice. (2 Marks)
c) Stores and displays the selected payment details. (2 Marks)

[10] (CO2/K3)

05. Design a C++ program to manage jewellery purchases using a class named BuyJewellery. The jewellery store allows customers to purchase gold, silver, and platinum, with fixed rates per gram: gold at ₹8600, silver at ₹105, and platinum at ₹2600. To encourage combo purchases, the store offers attractive discounts. If a customer purchases all three types of jewellery—gold, silver, and platinum—a flat discount of ₹3000 is applied. If the purchase includes only gold and platinum, a ₹2000 discount is given, and for silver and platinum, a ₹1000 discount is offered. No discount is applicable for any other combination.

The class should include three float data members: goldGrams, silverGrams, and platinumGrams to store the quantity of each metal in grams. Implement three types of constructors in the class: a default constructor that initializes all metal quantities to zero, a parameterized constructor to initialize them with user-supplied values, and a copy constructor that creates a duplicate of an existing BuyJewellery object. Additionally, define a member function discount() that calculates the total cost before discount, determines the eligible discount based on the metal combination, and computes the final payable amount after applying the discount.

To demonstrate the functionality, create objects using each of the three constructors and invoke the discount() function for each object to display the cost computations accordingly.

[10] (CO3/K3)

06. You are tasked to develop a C++ program to model a **bank account system**. Define a class BankAccount with private members accountNumber and balance representing the account details. Implement the following functionalities to manage bank accounts:

a) Constructor: Define a constructor to initialize the accountNumber (string) and balance (double) of the bank account. (2 Marks)
b) Operator Overloading: Overload the + operator to combine two BankAccount objects by adding their balances and creating a new BankAccount object with a new account number (concatenate the two account numbers with a hyphen, e.g., "123-456"). The new account's balance should be the sum of the two accounts' balances. (4 Marks)
c) Friend Function for Interest Calculation: Implement a friend function named calculateInterest() that calculates and returns the annual interest earned on the balance of any bankAccount object and the rate. (4 Marks)

Use the formula: interest = balance * rate

Note: Use appropriate access specifiers to protect the account details and ensure the friend functions work correctly.

[10] (CO3/K3)

07. You are tasked with developing a modular and extensible 2D Graphics Editor application aimed at supporting various shape objects. The system should allow for dynamic creation, storage, display, and area computation of different geometric shapes, all managed efficiently using runtime polymorphism and STL vectors for storage. The application must implement dynamic memory allocation to manage the lifetime of shape objects and ensure proper memory cleanup when the application terminates.

To achieve this, begin by defining an abstract base class Shape. This class should declare a pure virtual function float area() that each derived class must implement to compute the specific area of that shape. Additionally, include a virtual function void display() to output shape-specific details.

From this base class, derive at least two concrete shape classes — Circle and Rectangle. The Circle class should accept a radius as input, while the Rectangle class should accept length and breadth. Both classes must override the area() and display() methods, and make use of constructor initializer lists to initialize their properties. The system must support the creation of these shape objects dynamically at runtime.

In the main() function, use a std::vector<Shape*> to store pointers to the base class. Implement this problem in C++.

[10] (CO3/K4)

Section - II
Answer all Questions (2 × 15 Marks)

08. You are required to develop a C program to manage vehicle records at a toll booth using structures and unions. The program should efficiently store, compute, and display toll charges based on the type of vehicle.

VehicleInfo, Heavy_Toll and Vehicle are the structures.

TollCharges is an Union - need to be declared with the following members:
VehicleInfo: reg_no (string), owner_name (string), reg_year (integer)

Heavy_Toll: toll_charge (float), extra_load_charge (float)

Vehicle: info (struct VehicleInfo), vehicle_type (string), charges (union TollCharges)

TollCharges: standard_toll (float), heavy_toll (struct Heavy_Toll)

Assume that the vehicle_type is either car, truck or bus. If the vehicle is either a car or bus, use a standard_toll charge and let it be Rs. 115. If the vehicle is a truck, compute the charge using both toll_charge and extra_load _charge. Read both these values from user.

Develop a C code with appropriate structure definitions and union definitions that dynamically allocate memory to store the details of N number of vehicles (where N is entered by the user).

In addition to main(), implement the following functions:

- **acceptVehicleDetails()**

Accepts details of N vehicles from the user and stores them dynamically.

- **computeTollCharge()**

Computes the total toll charge for each vehicle based on its type.

- **displayVehicleRecords()**

Displays the complete details of all vehicles along with their respective toll charges.

- **searchVehicleByRegNo()**

Allows the user to search for a specific vehicle by its registration number.
If found, display its details and toll charge; otherwise, indicate that the vehicle is not found.

- **searchVehicleByType()**

Allows the user to search for vehicles by vehicle_type.
If found, display its details. otherwise, indicate that the vehicle is not found.

[15] (CO2/K3)

09. You are tasked with developing a C++ program for an Employee Benefit System in a company that calculates various financial benefits for employees, specifically the Employee Provident Fund (EPF) and Gratuity.

The system should consist of the following class hierarchy:

Employee Class (Base Class):
This class stores general employee details. All variables across all classes should be declared with the protected access specifier. The class should contain the following data members- Employee's name, Unique identifier, basic_salary, years_of_service. Define the member functions- inputDetails() – Accepts employee data, displayDetails() – Displays employee data.

ProvidentFund Class (Derived from Employee):
This class is responsible for calculating the Employee Provident Fund (EPF). It includes an additional attribute: employer_contrib – Employer's monthly contribution to the EPF fund. The class should implement the following functions -

- A function to receive the employer's contribution.
- calculateEPF() – Calculates the EPF using the formula:
  EPF = (12% of basic salary) + employer contribution

Gratuity Class (Derived from Employee):

This class handles gratuity calculation based on the employee's years of service. It introduces the attribute: bonus_years – Extra credit years awarded for long service. Include the following functions:

- A function to input bonus_years
- calculateGratuity() – Calculates gratuity using the formula:
  Gratuity = daily_salary × 15 × (years_of_service + bonus_years)
  (Assume 26 working days per month)

EmployeeBenefits Class (Inherits from both ProvidentFund and Gratuity):

This class brings together both EPF and gratuity calculations. Implement the following functions:
inputEmployeeData() – Receives general employee details, employer contribution, and bonus years
calculateEPF() – Uses the inherited method to compute EPF
calculateGratuity() – Uses the inherited method to compute gratuity
displayBenefitsSummary() – Displays the complete benefit details, including EPF and gratuity

Based on the above scenario use appropriate techniques to handle this situation and successfully compute the EPF and Gratuity amount for the Employee.

[15] (CO3/K4)

BL-Bloom's Taxonomy Levels - (K1-Remembering,K2-Understanding,K3-Applying,K4-Analysing,K5-Evaluating,K6-Creating)