DEPARTMENT OF COMPUTER SCIENCE

Cochin University of Science and Technology

M.Sc. (Five Year Integrated) in Computer Science (AI & DS)

End Semester Examination, March 2022

21-805-0103: Object Oriented Programming

Time: 3 hours

Maximum Marks: 50

Module I

1. (a) Write a C++ program to compute the value of e^x by using the formula.

$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$

Pass the value of x as command line argument. Use user-defined functions in your program.

(*Hint:* Use **std::stoi(string_var)** to convert a string to **int**.) [6]

(b) Compare and contrast procedure-oriented and object-oriented programming paradigms. [4]

OR

2. (a) Write a C++ program to generate the following *egg* − *timer* shape.

```
#   #   #   #   #   #   #   #   #
#   *   *   *   *   *   *   *   #
#       *   *   *   *   *       #
#           *   *   *           #
#               *               #
#           *   *   *           #
#       *   *   *   *   *       #
#   *   *   *   *   *   *   *   #
#   #   #   #   #   #   #   #   #
```

[7]

(b) Define the term Abstract Data Types (ADT). [3]

Module II

3. (a) Create a class *Person* and two derived classes *Employee* and *Student*, inherited from class *Person*. Now create a class *Manager* which is derived from two base classes *Employee* and *Student*. Write a C++ program to demonstrate the use of virtual base class. Use constructors in your program to initialize the data members of both derived and base classes. [7]

(b) Define the term *containership* in C++. Give an example. [3]

OR

P.T.O.

4. (a) Write a C++ program which creates a multiple inheritance hierarchy of *Teacher* class derived from *Person* and *Employee* classes. Each class must implement a *show*() member function and utilize scope resolution operator to access the member function using *Teacher* object. Use constructors to initialize data members of both derived and based classes. [7]

(b) Discuss the use of *protected* visibility label in the context of class inheritance. [3]

Module III

5. (a) Write a C++ program to overload the '*' operator to multiply two *Matrix* objects. Use *friend* function and constructors in your program. [7]

(b) What are dynamic constructors? How we can dynamically construct a 2-D array member of a class using its constructor? [3]

OR

6. (a) Write a C++ program to demonstrate the conversion of one class type to another using a casting operator function. [7]

(b) With the help of an example program show how constructors with default arguments can be defined. [3]

Module IV

7. (a) Crreate a base class called *Shape*. Use this class to store two double type values that could be used to compute area of figures. Derive two specific classes called *Triangle* and *Rectangle* from the base class *Shape*. Add to the base class, a member function *get_data*() to initialize base class data members and another member function *display_area*() to compute and display area of figures. Make *display_area*() as a virtual function and redefine this function in the derived classes to suit their requirements. Write a C++ program that accept the dimensions of a triangle or a rectangle interactively and display the area. (Input treated as base and height in the case of a triangle and as length of two sides in the case of a rectangle.) [7]

(b) Define the term *pure virtual function*. Give an example. [3]

OR

8. (a) Suppose you have an inheritance hierarchy with a base class *Animal* and two derived classes *Bird* and *Snake*. Based on the characteristics of animals in general, and birds and snakes specifically, can you think of possible virtual and pure virtual functions to place in *Animal* class. Justify your answer. [6]

(b) What are abstract base classes? What condition should be met for a class to be abstract? [4]

Module V

9. (a) Write a C++ program with templates to implement a class named *Calculator* which contains member functions to add, substract, multiply and divide two numbers. [6]

(b) Define *inline* functions. Explain its significance in class member function definition. [4]

OR

10. (a) Write a C++ program to demonstrate the concept of exception handling related to *division-by-zero* exception.

(b) Write a C++ program to swap data using function templates.