3rd Semester B.Tech Examination, Nov. 2004

OBJECT ORIENTED PROGRAMMING USING C++

Full Marks : 70

Time : 3 hours

Answer question **1** and any **five** from questions **2 to 8**

The figures in the right-hand margin indicate marks

Parts of the same question should be answered together in the same sequence

1. Identify whether the following statements are *True* or *False*. Justify your answer in each case. 2 × 10

   (*a*) A virtual base class is useful in C++ when the different methods in the base and derived classes have the same name.

   (*b*) It is prudent to use the default destructors provided by the C++ language, rather than writing specific destructors for a class.

(*c*) In C++ suppose you derive a class named manager from a class named employee. Then, you can always assign an object of the class employee to an object of the manager class without encountering any compilation error.

(*d*) C++ classes can be considered as abstract data types (ADTs).

(*e*) A class can have many methods with the same name.

(*f*) Deep class hierarchies are signs of an object-oriented design done well.

(*g*) The aggregation relationship among classes is symmetric.

(*h*) The use of the inheritance feature in an object-oriented program results in code reuse.

(*i*) The aggregation relationship among classes can be considered to be a special type of association relationship.

(*j*) An object-oriented program that does not derive new classes through inheritance, cannot exhibit polymorphic behavior through dynamic binding.

2. Write C++ code for the following description of the class/subclasses:

Define an abstract base class 'GeomShape' that has the following:

- data members for the (x,y) co-ordinate position
- a constructor for initilizing GeomShapes
- a virtual method 'MoveShape()'
- a virtual method 'PrintShape()' to output an object

Derive subclasses 'GeomLine', Geom Circle' and 'GeomTriangle, from 'Geom Shape', and implement 'MoveShape()' and 'PrintShape()' methods for each of the sub-classes. You may assume appropriate data-members for each of the subclasses. You should use appropriate access controls in 'GeomShape'. 10

3. (*a*) An abstract class cannot have instances. What then is the use of having abstract classes? Explain your answer using a suitable example.

(*b*) What is the difference between method overloading and method overriding? Explain your answer using a suitable example. 5+5

4. (*a*) Explain why object-oriented programs are more maintainable and reusable compared to function-oriented programs.

(*b*) Give the definition of a virtual base class in C++ syntax. Explain why virtual base classes are required. 5+5

5. (*a*) What do you understand by *exceptions* in a C++ program? With an example explain how exceptions are handled in a C++ program.

(*b*) What do you mean by public, protected, and private attributes of a class? Why is this distinction among attributes necessary? Explain your answer using suitable examples. 5+5

6. Consider a small student's library. The library has a book catalog consisting of 100 issuable items. Each issuable item can either be a book or a periodical. The issue methods of the books and periodicals override the issue method of the issuable class. The member catalog consists of 50 members. Each member is either a student or a teacher. When a book is issued, the member's identity is recorded in the book object. Also, the member object must record the book issued to him. Once a book is returned, these associations are dissolved. Represent your design using Booch's notations. Write C++ code to implement your design for the library system. 10

7. (*a*) Explain the problem of repeated inheritance. With the help of suitable examples explain how this problem can be overcome in C++.

(*b*) Explain the following concepts and explain how they can be implemented in C++ language by using suitable examples:

- Inheritance
- Composition
- Abstract class 5+5

8. (*a*) Write C++ code for the following description of the class "circle": A C++ class named "circle" stores the co-ordinate position of the center of a circle and its radius as three floating point numbers. It supports the following methods:

- **circle**: constructor method which initializes the center and radius of the circle during object creation

- **print**: displays the current center and radius of the circle

- **change**: changes the center point and radius to the specified values.

(*Continued*)

(*b*) Define the following overloaded operators

- = = checks whether two circles are identical

- = circle assignment. 5+5