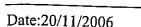
The Open University of Sri Lanka

B.Sc. Degree Programme: Level 04

Final Examination 2006

## CSU 3279- Object Oriented Programming - Paper II

Duration: Two and Half Hours



Time: 9.30 am -12.00 noon

066

Answer FOUR questions only.

Q1.

- (a) Define the terms *Object*, *Class*, *Abstraction*, *Inheritance* and *Polymorphism* in the context of object-oriented programming.
- (b) What do you mean by information hiding? Explain using an example.
- (c) Create a C++ class to represent a person with attributes name, year of birth and height in meters.
  - i) Define methods to set these three attributes.
  - ii) Add a method which will return a person's (approximate) age when the year of birth is passed as a parameter.
  - iii) Add another method which will return the person's height in centimeters.

**Q2.** 

- (a) What do you mean by *function overloading*? Explain very clearly, using an example.
- (b) Define a class to represent a complex number. This class should contain the following members:

Data members

- i) Real part
- ii) Imaginary part

Member functions:

- i) A default constructor to initialize both real and imaginary part to zero.
- ii) A user defined constructor to initialize data members of the class.
- iii) Overload + operator for adding two complex numbers.
- iv) Overload \* operator for multiplying a complex number by a scalar. [E.g. 2(5+6i) = (10+12i)]
- v) To display the data members of the complex number.

Write a simple main () function to test your class.

Q3.

- (a) (i) What is the purpose of the default constructor and the user defined constructor?
  - (ii) Describe the importance of a *destructor*.
- (b) Define a class to represent the time named as *Time*. This class should contain the following members:

Data members

- i) hour
- ii) minute
- iii) second

Member functions:

- i) int Time:: get hour();
- ii) int Time:: get\_min();
- iii) int Time:: get sec();

Write member functions for *Time* that will return the three parts of time as integers. Use the function prototypes given above.

(Note: Write a complete C++ program to represent the above class)

Q4.

- (a) (i) Explain the concept of friend function and friend classes.
  - (ii) A friend function violates the *Data hiding* concept in Object-Oriented Programming. Do you agree with this statement or not? Justify your answer.
- (b) (i) Declare a base class **Thermometer** that holds a temperature in degrees Kelvin. Provide methods that will allow a new temperature to be entered and another that will return the temperature in degrees Kelvin.
  - (ii) Declare two classes derived from the **Thermometer** class in the above part (b-i) called **Celsius** and **Fahrenheit**. Each should use the same method for entering a new temperature as the base class but the function that returns the temperature should provide a Fahrenheit or Celsius value.

(*Hint*: Equation for converting Celsius temperature into Fahrenheit is, F = (32+C) 9/5)

CSU 3279 Object Oriented Programming final-2006

(c)

Q5

(a)

(b)

(;

C

(

What is an "abstract class"? Give a real world example to illustrate it.

- (i) Distinguish Abstraction via encapsulation and explain the difference among private, protected and public data and method types.
- (ii) What is the purpose of overriding a function? Give a suitable example.
- (i) Discuss the role of inheritance in Object Oriented Programming.
- (ii) Briefly explain the role of multiple inheritance.

06.

(iii) When do we use the **protected** visibility specifier to a class member? Give a suitable example.

Briefly discuss the role of polymorphism in Object-Oriented Programming.

What is meant by the term aggregation used in Object-Oriented Programming?

State whether the following statements are TRUE or FALSE.

- (i) Every class can have only the default constructors.
- (ii) An abstract data type defines the attributes and methods of all objects belonging to a particular 'class'.
- (iii) A friend function is called like f(x), while a member function is called like x.f(x).
- (iv) Genericity can not be a powerful tool allowing us to create generic functions, methods and classes.
- (v) Inheritance allows classes to inherit attributes and methods from other classes in a classification hierarchy.
- (vi) Polymorphism means 'having many forms'. In an object-oriented program, methods and operators can have many forms by being 'overloaded' in various ways.
- (vii) Multiple inheritance applies where a class is 'a kind of ' more than one base class.
- (viii) Derived classes inherit overloaded assignment operators.
- (ix) Aggregations have certain properties (transitivity, antisymmetry and propagation) and may be of various types.
- (x) Containers vary in characteristics such as the ability to contain objects of different types, whether they are of fixed size, and what methods of access they allow.

μg

\*\*\*\*