LIBRARY
15 AUG 2011

00082

THE OPEN UNIVERSITY OF SRI LANKA
B.Sc DEGREE PROGRAMME: LEVEL 04
FINAL EXAMINATION: SEMESTER 1-2010/2011
CPU2242: OBJECT ORIENTED PROGRAMMING USING C++ AND JAVA

DURATION: **THREE HOURS** (3 HOURS)

Date: **13th January, 2011** Time: **9.30 am – 12.30 pm**

Answer FOUR Questions ONLY.

Q1.

a) Describe the differences between procedure-oriented programming and object-oriented programming.

b) What are the three main concepts of object-oriented programming? Describe them briefly.

c) List three (03) differences between C++ and Java.

d) Explain the process of converting a Java stand-alone program (source code) into a machine language.

e) What is late binding? Give a simple example to describe late binding.

f) What is an abstract class?

Q2.

a) State whether the following statements are **TRUE** or **FALSE**.

i. x++ is a valid C++ variable name.

ii. Java has unsigned versions of each integer data types.

iii. The following C++ statements,

```
for (int i=1;i<=5;i++){
   for (int j=1;j<=i;j++){
      cout << '*';
   }
}
```

Produce the following output.

```
*
* *
* * *
* * * *
* * * * *
```

iv. Java was invented by Bjarne Stroustrup.

v. C++ constructor may return a value.

b) Explain why a member function cannot be used to overload the multiplication operator (*) in C++ to multiply a scalar (e.g., 4.5, 8) and an object of a class. Assume that the scalar is appeared before the object in the multiplication.

c) Discuss three different ways to use an already tested and debugged class.

d) Write three (03) advantages of using functions in a C++ program.

e) Write a C++ function to return the factorial of a number. The factorial of a number is defined as follows

$$n! = \begin{cases} 1 & if\ n = 0 \\ 1 \times 2 \times \ldots \times n & n > 0 \end{cases}$$

Q3.

a) What are the differences between a Java applet and a Java stand-alone application?

b) What are the different forms of inheritance? Give an example for each.

c) Consider the following Java class

```
class Hello {
   public static void main (String[] args) {
    System.out.println ("Hello: This is my first Java
    program");
   }
}
```

Explain why the keywords **public**, **static** and **void** are included in the header of the main () method.

d) State whether the following names are valid Java identifiers. Briefly explain how you arrived to your conclusion.

i. For
ii. 2dayRainfall
iii. endSales
iv. weighted-average
v. my_salary

e) Create a Java class called **Invoice** that a hardware store might use to represent an invoice for an item sold at the store.

i. The following instance variables and methods should be included in your class.

a. Four pieces of information as instance variables – item number, item description, quantity of the item being purchased, and price per item
b. A constructor that initializes the four instance variables
c. Mutator and accessor methods for each instance variable
d. A method named **getInvoiceAmount** that calculates the invoice amount

ii. Write a test application named **InvoiceTest** that demonstrates class **Invoice**'s capabilities.

Q4.

a) Evaluate each of the following C++ expressions, assuming in each case that **m** has the value 25 and **n** has the value 7.

i. m - 8 - n

ii. m / n * 7

iii. m % 6 + n > 5

iv. m && n > 12 / 6

v. 2 * m + 6 || n – 7

b) Explain the following types of errors that could happen in a computer program.

i. Syntax errors

ii. Logical errors

iii. Run-time errors

c) Write C++ function prototypes for the following descriptions.

i. **toyStory** () takes two float arguments **x** and **y**, changes the value of **y** to **x** – **y**.

ii. **arraySum** () takes a float array and the size of the array as arguments and return the sum of array elements.

iii. **readStudent** () reads the field of a structure variable of structure **Student** that includes the information of a student.

d) What is the difference between function overloading and function overriding?

e) Write a C++ class to represent a bank account.

i. The following data members and member functions should be included in your class.

a. Data members – depositor's name, the account number and the balance

b. Member functions

- Creating an object and initializing it
- Displaying the depositor's name, account number, and balance
- Depositing an amount of money given by an argument
- Withdrawing an amount of money given by an argument

ii. Write a suitable main function to test your class.

Q5.

a) What is "operator overloading"?

b) Write three advantages of using operator overloading instead of a normal member function or a friend function for the same purpose.

c) State whether the following statements are **TRUE** or **FALSE**.

i. All C++ operators can be overloaded for user-defined classes.

ii. Operator overloading is available in Java.

iii. Grammatical rules that apply on operators can be changed for user-defined classes.

iv. A friend function will have no arguments for unary operators and only one argument for binary operators.

v. The C++ compiler interprets the expression **O1 + O2** as **operator + (O1, O2)** in case of a friend function. Here **O1** and **O2** are two objects of a class that overloaded + operator using a friend functions.

d) Write a C++ class named **Money** to represent an amount of Sri Lankan money that consists of rupees and cents. Use integer values to hold rupees and cents. Include the following member functions or friend functions.

i. A default constructor and a parameter constructor

ii. To convert a money in rupees and cents into equivalent cents.

iii. To convert a given cents into the equivalent Money object and return it.

iv. To overload + operator to add two moneys.

v. To overload * operator to multiply a money by an integer (allow to use only integer*money, but not money*integer).

vi. To overload >> operator to read a money from the keyboard. This function should read rupees and cents of the money separately.

vii. To overload << operator to print a money. This function should print rupees and cents of the money separately.

Q6.

a) Explain when protected members are required in a C++ class.

b) Explain when a destructor is required for a C++ class.

c) Fill the following table for visibilities of inherited members of a derived class in C++.

Base class visibility	**Derived class visibility**	
	Public derivation	**Private derivation**
Private		
Protected		
Public		

d) Consider the following class definitions.

```
class Parent1 {
 private:
   int i;
 protected:
   float x;
 public:
   double y;
   int getter_x ();
};
```

```
class Parent2 {
 private:
   int a;
 protected:
   float b;
 public:
   double z;
   int getter_a ();
};
```

```
class Child: private Parent1, public Parent2 {
 private:
   int c;
 public:
   int getter_c ();
   void print ();
};
```

Using the table filled in part Q6. c), find the visibilities of inherited members of the derived class **Child**.

e) Write a C++ class named **Circle** to represent a circle. The following data members and member functions should be included in the class **Circle**.

- A data member to hold the radius of the circle
- A default constructor and a parameter constructor
- Accessor and mutator methods for the radius
- To return the area of the circle

Create a subclass named **Cylinder** of **Circle** to represent a cylinder that contains the following data members and member functions.

- A data member to hold the height of the cylinder
- A default constructor and a parameter constructor
- Accessor and mutator methods for the height
- To return the volume of the cylinder

*** All Rights Reserved ***