THE OPEN UNIVERSITY OF SRI LANKA
B.Sc DEGREE PROGRAMME: LEVEL 04
FINAL EXAMINATION: SEMESTER 1-2015/2016
CPU2242: OBJECT ORIENTED PROGRAMMING USING C++ AND JAVA

DURATION: **THREE HOURS** (3 HOURS)
Date: **05th July, 2016** Time: **1.00 pm – 4.00 pm**

Answer FOUR Questions ONLY.

Q1.

a) What is *object-oriented programming*? How is it different from *procedure-oriented programming*?

b) Explain four (04) benefits of object-oriented programming.

c) Briefly explain the following basic concepts of object-oriented programming.

i). Class
ii). Object
iii). Encapsulation

d) Which of the following statements are **true** with respect to C++?

i). Function overloading is done at compile time.
ii). Protected members are accessible to the members of derived class.
iii). A derived class inherits constructors and destructors.
iv). A friend function can be called like a normal function.
v). Nested class is a derived class.

Q2.

a) What is a *constructor*?

b) Differentiate between *constructor* and *destructor* in C++.

c) Define *private*, *protected* and *public* access specifiers.

a) Consider the following C++ program. Rewrite the program including default constructor, parameterized constructor, copy constructor and destructor.

```
#include <iostream>
using namespace std;

class B
{
      int a;
      int b;
```

```
    public:
        void get_ab();
        int get_a();
        void show_a();
};

void B :: get_ab(){
    a = 5;
    b = 10;
}

int B :: get_a(){
    return a;
}

void B :: show_a(){
    cout << "a = " << a << "\n" ;
}

int main()
{
    B b;
    b.get_ab();
    b.show_a();
    return 0;
}
```

Q3.

a)

i). What is *exception handling* in Java?

ii). Write two (02) advantages of using them in a program.

b) What do you meant by *abstract class*? Give an example.

c) State whether the following statements are TRUE or FALSE with respect to Java.

i). A private member of a class cannot be accessed by the methods of the same class.

ii). Abstract methods should be implemented in the derived class.

iii). A class which is implementing an interface must implement all the methods of the interface.

iv). A finally block is executed before the catch block but after the try block.

v). Java is a fully object oriented language with strong support for proper software engineering techniques.

d) Define a class named **circle** in Java which includes the following data members and methods.

- Two data members to store the radius and the colour of the circle.

- Default constructor and parameterized constructor.
- `getRadius()` method to access the radius of the circle.
- `getColor()` method to access the colour of the circle.
- `getArea()` method to calculate the area of the circle and return it.

  (*Hint:* Area of the circle = πr^2)

Q4.

a) What are *friend functions* in C++? Explain with an example.

b) Mention three (03) operators which cannot be overloaded.

c) Explain unary and binary operator overloading with an example for each.

d) Consider the following class named **complex** in C++ to represent a complex number which includes real and imaginary part in floating point value. Include the following member functions or friend functions to the class.

```
class complex
{
    private:
        float real;      // Real Part
        float imag;      // Imaginary Part
};
```

i). A default constructor and a parameterized constructor.

ii). To overload + operator to add two complex numbers.

(Example: `(3 + 2i) + (1 + 7i) = 4 + 9i`)

iii). To overload * operator to multiply a complex number by an integer.

(Example: `2 * (3 + i) = 6 + 2i`)

iv). To overload = = operator to check whether two complex numbers are equivalent.

(Example: Two complex numbers `(x + yi)` and `(a + bi)` are equivalent if, the real parts are equal `(x = a)` and the imaginary parts are equal `(y = b)`)

v). To overload << operator to print a complex number.

vi). Write a suitable main function to test your class.

Q5.

a) What is *inheritance*?

b) Explain three (03) characteristics of inheritance.

c) State two (02) different forms of inheritance supported by C++. Explain them with an example.

d) Briefly describe the following concepts of inheritance.

i). Base class
ii). Sub class

e) State whether the following statements are TRUE or FALSE with respect to C++.

i). When a base class is privately inherited, a private member of base class becomes private member of the derived class.

ii). When a base class is privately inherited, public members of the base class become private members of the derived class.

iii). When a base class is publicly inherited protected members of base class become protected members of the derived class.

iv). We can specify which data and function members of the base class will be inherited by derived class.

v). When an object of a derived class is defined, the derived class constructor should be called before the base class constructor.

f) What is the output of the following C++ program?

```
#include <iostream>
using namespace std;

class Polygon {
   protected:
    int width, height;
   public:
     void set_values(int a, int b){
      width = a; height = b; }
};

class Output {
public:
    void print (int i);
};

void Output::print (int i) {
  cout << i <<'\n';
}

class Rectangle: public Polygon, public Output {
public:
    int area ()
      { return width*height; }
};
```

```
class Triangle: public Polygon, public Output {
public:
    int area ()
      { return width * height/2; }
};

int main () {
  Rectangle rect;
  Triangle trgl;
  rect.set_values (4, 5);
  trgl.set_values (4, 5);
  rect.print (rect.area());
  trgl.print (trgl.area());
  return 0;
}
```

Q6.

a) Differentiate between *function overloading* and *overriding.*

b) What is *polymorphism*?

c)

i). Define *virtual function.*

ii). Why do we use virtual functions?

d) Consider the following C++ program to answer the questions given below.

```
#include <iostream>
using namespace std;

class B {
    public:
       void display()
         { cout<<"Content of base class.\n"; }
};

class D1 : public B {
    public:
       void display()
         { cout<<"Content of first derived class.\n"; }
};

class D2 : public B {
    public:
       void display()
         { cout<<"Content of second derived class.\n"; }
};

int main() {
    B *b;
    D1 d1;
    D2 d2;

    b = &d1;
```

```
    b->display();

    b = &d2;
    b->display();
    return 0;
}
```

i). What is the output of the above C++ program?

ii). The user wants to display the output as follows. Rewrite the appropriate class to achieve the target of the user.

```
Content of first derived class.
Content of second derived class.
```