No. of Printed Pages : 3

CS-72

BACHELOR OF COMPUTER APPLICATIONS (BCA) (Pre-Revised)

01386

Term-End Examination

June, 2016

CS-72 : C++ AND OBJECT ORIENTED PROGRAMMING

Time : 2 hours *Maximum Marks : 60*

Note : *Question number 1 is* ***compulsory.*** *Attempt any* ***three*** *questions from the rest. All program codes must be in C++ programming language only.*

1. (a) What is dynamism in the context of OOP paradigm ? What are the different types of dynamisms for object-oriented design ? Explain each with the help of examples. *5*

(b) Differentiate between private and protected class members. Elaborate with suitable examples. *5*

(c) Explain the following : *10*

(i) Library files

(ii) #include directive

(iii) How are objects defined for a class ?

(iv) FOR loop

(v) Constructors and destructors

(d) Design the following functions using C++ : *5*

(i) Computing sum of two numbers

(ii) Swapping two numbers

(e) Explain any five features of JAVA giving suitable examples. 5

2. (a) Compare structured programming and object oriented programming. 5

(b) In the following code, point out errors, if any, in statements marked (i) to (iv). If there are no errors, indicate their output. 5

```
class base
  { int i;
    public:
    void set_i (int num) {i = num;}
    int get_i( ) {return i ;}
  };
class derived : public base
  { int j;
    public
    void set_j (int num) {j = num;}
    int get_j( ) {return j;}
  };
void main( )
  { base *bp;
    derived d;
    bp = &d;
    bp - -> set_i(10);              (i)
    bp - -> set_j(20);              (ii)
    cout << bp - -> get_i( );       (iii)
    cout << bp - -> get_j( );       (iv)
```

3. (a) What is exception handling ? Explain it with an example. *5*

(b) Write a C++ program to accept a number from the user and display its factorial. *5*

4. (a) What is Operator Overloading ? Explain it with the help of an example. *5*

(b) Explain Reusability and Data Hiding. *5*

5. Explain the following : *10*

(a) this pointer

(b) Copy constructor

(c) Redirection operators

(d) Inline functions

(e) Runtime polymorphism
