Seat No.: ________ Enrolment No.______________

GUJARAT TECHNOLOGICAL UNIVERSITY

MCA- IInd SEMESTER–EXAMINATION –JUNE - 2012

Subject code: 620003 Date: 09/06/2012

Subject Name: Object Oriented Concepts and Programming (OOCP)

Time: 10:30 am – 01:00 pm Total Marks: 70

Instructions:

1. **Attempt all questions.**
2. **Make suitable assumptions wherever necessary.**
3. **Figures to the right indicate full marks.**

Q.1 **(a)** What is object-oriented programming? How is it different from the procedure-oriented programming? Discuss in brief, the main features of object oriented programming. **07**

(b) What is meant by access specifiers? Why are they needed? Which access specifiers are available in C++ and what is the effect of using them? Explain with example. **07**

Q.2 **(a)** How are static members defined in a c++ class? How is a static member different than a normal member of a class? Give an example to use static data member and static member function. **07**

(b) What is a namespace? Why is it required to be used? Explain various ways of using members of a namespace in the program. What are unnamed namespaces? **07**

OR

(b) Differentiate between text and binary stream files. What are the different ways of opening a file in c++? Explain with example use of seekg(), seekp(), tellg() and tellp() functions. **07**

Q.3 **(a)** What do you mean by function template and class template? Explain the need for creating function and class templates by giving example. **07**

(b) (1) Describe the pros and cons of using inline function versus using normal functions. Also mention in which situations the compiler will not allow the function to become inline. **04**

(2)What is member initialization list? Explain all different ways of using it with example. **03**

OR

Q.3 **(a)** What is Exception? What are the benefits of exception handling through C++? Explain how exception handling is implemented in c++ using an example. **07**

(b) (1) What do you understand by copy constructor and dynamic constructor? Explain giving suitable examples **04**

(2)What is conversion function? How is it created? Explain its syntax giving suitable example. **03**

Q.4 **(a)** State True/False It is must to justify your answer **07**

(1) Using the operator overloading concept, we can change the meaning of an operator.

(2) Destructors never take any argument.

(3) Friend functions have access to only public members of a class.

(4) cin and cout are built-in functions.

(5) The main advantage of width() function is that we can use one width specification for more than one items.
(6) Inheritance means the ability to reuse the data values of one object by other objects.
(7) When a function returns a value, the entire function call can be assigned to a variable.

(b) Write a program which copies its standard input, line by line, to its standard output. **07**

OR

Q.4 **(a)** Explain the terms : **07**
(1) friend
(2) this
(3) virtual base class
(4) reference variable
(5) containership
(6) pointer to constant
(7) constant pointer

(b) Given two classes cylinder and volume as below, **07**

```
class cylinder { int r,h; };
class volume { long vol; }
```

calculate the volume, assign it to member vol in class volume and display the value of vol. Make use of friend function.

Q.5 **(a)** (1) Write a C++ program that converts user defined data type into integer data type. **04**

03

(2) Create a user-defined manipulator function to show the positive sign before positive numbers, to show trailing zeros and to set the width of the output as 10. Write a c++ program to use this manipulator.

(b) How is polymorphism achieved at compile time? What is meant by run-time polymorphism? Explain why run-time polymorphism is useful and give example for implementing run-time polymorphism in c++. **07**

OR

Q.5 **(a)** Find error(s), if any or give output for the following

(1) **04**

```
#include <iostream.h>
   #include <conio.h>
   class A
   {    public :
          virtual void fun1()    { cout << "\n In A::fun1";   }
          void fun2()   { cout <<"\n In A::fun2()";  }      };
   class B : public A
   {
    public :
            void fun1() {  cout << "\n In B::fun1";       }
            void fun2() {   cout << "\n In B::fun2";      }
    };
void main()
```

```
{
   A *ptr1,*ptr2;    A objA;    B objB;
   ptr1 = &objA;
   ptr2 = &objB;
   ptr1->fun1();
   ptr1->fun2();
   ptr2->fun1();
   ptr2->fun2();
}
```

(2) **03**

```
 #include <iostream.h>
 class test
{
  private:
        int t1,t2,t3;
  public :
        test(int x,int y,int z)
                { t1 = x; t2 = y; t3 = z;}
        void display()
                { cout << "\n "<<t1 <<t2<< t3; }
};
int main()
{
        test r(2,3,4),s;
        r.display();
        s.display();
         return 0;
}
```

(b) What is RTTI? Why is RTTI needed? Explain the use of typeid and dynamic_cast with example. **07**

****