K J SOMAIYA INSTITUTE OF MANAGEMENT STUDIES AND RESEARCH
VIDYA NAGAR, VIDYA VIHAR, MUMBAI – 400 077

Batch : MCA 2019-21
Semester I
Programming and Object Oriented Concepts using C++
End-Term Exam

Max. Marks: 50 Duration: 3 Hrs 09 December,2019

Note:. Question Number 1 is compulsory. Answer any 3 questions out of the remaining 4 questions. DO NOT write full or even any part of a question on the answer sheets except the question number.

[Q.1.] Create a class Array which holds a **dynamic** integer array along with its size. It should have the following methods :

I. A default constructor which creates a dynamic array of size 5 for all objects whose size has not been passed by the client.
II. A one-argument constructor which creates a dynamic array of the size passed as the argument.
III. A copy constructor which does an object initialization with the contents of another Array object.
IV. A set function which populates a calling object with data from the keyboard.
V. An overloaded assignment operator function.
VI. A destructor function which frees the dynamic memory of each object that goes out of scope.
VII. A print function which prints all the array elements of the calling object in formatted output.
VIII. An overloaded ++ function which increments every element of the array of the calling object by 1.
IX. An overloaded = = function which returns a true if two Array objects are identical or false if they're not.
X. An overloaded [ ] subscript operation function which returns the reference of the requested array element or the value stored at that subscript, as the case may be.

A sample client code is given below : **[20 marks]**

```
void main()
{
        void process(Array);
        Array A1, A2(7);
        Al.print();
        A2.print();

        Array A3 = A2;
```

```
        if (A3 == A2)
        {
                A3.print();
                cout<< " is equal to ";
                A2.print();
        }
        else
        {
                A3.print();
                cout<< " is not equal to ";
                A2.print();
        }

        cout<<endl<< "The 5th array element of object A3 = " << A3[5];
        A3[5] = 40;
        cout<<endl<< "The 5th array element of object A3 = " << A3[5];

        process(A2);
}

void process(Array A4)
{
        A4.print();
        A4++;
        cout<< "\n Object A4 after incrementing …\n";
        A4.print();
}
```

[Q.2.a] Create a **Multilevel inheritance hierarchy** for the Point – Circle - Cylinder classes. Class Point will have x and y as float coordinates, class Circle will have an int radius and class Cylinder will have int height as its private data member. class Circle will have an additional method called area() which returns the area of the circle. class Cylinder will have two additional methods, area() which returns the area of the cylinder, and volume() which returns the volume of the cylinder.

Have the necessary constructors and show functions in the entire hierarchy with ***cascading calls for constructors***. Write your own client code demonstrating the entire hierarchy and invoking all the functions. **Hint :**

area of cylinder : (2 * (PI * r^2)) + (2 * PI *r * h)
volume of cylinder : (PI * r^2) * h

(Note : Do not create virtual functions. Plain Multilevel Inheritance)

[Q.2.b] Explain what is abstraction and Encapsulation.. **[10 marks]**

[Q.3.a.]Write a program to demonstrate **container classes** (also known as **Embedded** or **composite** classes). Have three classes: **Date**, **Address** and **Student**. Class **Date** will have day, month and year; class **Address** will have street Name, city and state; and class **Student** will have regno, name, DOB and contact. DOB (Date Of Birth) will be an object of the **Date** class and contact will be an object of the **Address** class. Write a driver program to demonstrate the container class **Student** where a ***Student object is populated via cascading constructors***. Note, that classes **Date** and **Address** will have only the constructors and print() functions.

[Q.3.b] Explain with an example what's a **pure virtual function.** **[10 marks]**

[Q.4.a] Create a class **MyDate** with the basic public interface functions like setDate(), print() etc. Now have three different exception classes (empty body classes) called **dayError**, **monthError** and **yearError** inside the MyDate class. Any year outside the range 2000 and 2019 is erroneous. Have setDate() invoke validate() function which detects any of the three errors and throws the corresponding exception. The client code will have the **try** block and the three **catch** blocks for the three errors **throw**n, which will display suitable error messages. In case the date is valid, the client code will call the print() member function to display the date.

[Q.4.b] Explain function templates with an example. **[10 marks]**

[Q.5.a] Create a class **Time12** consisting of three private integers called hour, minute and second and a char string called ampm of length 3, and another class **Time24** with three private integers called hour, minute and second. class Time12 objects will hold time in "AM" or "PM" format and class Time24 objects will hold time in 24-hour format (which does not require AM or PM). Now define the necessary methods for both classes to do the conversions necessary for the sample driver program given below :

```
void main()
{
        Time24 T1(18,20,30);

        Time12  T2 = T1;                // define the conversion routine in the source class
        cout<< T1.showTime24() << " is the same as  "
        <<T2.showTime12() <<endl;
}
```

[Q.5.b] How do **static** data members differ from non-static data members ? Explain with examples. [**10 marks]**