10. Runtime Polymorphism, Virtual Destructors & Container classes

CO3: Implement the concept of reusability and data security

1. Write a C++ program to declare a function **show ()** in base and derived class. Display message through the function to know name of the class whose member function is executed. **use late binding concept** using virtual keyword
2. Write a C++ program for static binding (accessing member function using pointers-pointer casting)
3. Write a C++ program for dynamic binding (accessing member function using pointers)
4. Write a C++ program for abstract classes and concrete classes.
5. Write a C++ program to demonstrate virtual destructors.
6. Write a C++ program to demonstrate container class.
7. Write a C++ program to declare **matrix** class which has data member integer array as 3 x 3. Derive class **matrix A** from class matrix and **matrix B** from matrix A. All these classes should have a function show () to display the contents. Read and display the elements of all the three matrices.
8. Consider the following class definition

   ```
   class father
   {
           protected:
                   int age;
           public:
           father(int x)
                   {
                           age=x;
                   }
           virtual void iam()
           {
                   cout<<"I AM THE FATHER, my age is"<<age<<endl;
           }
   };
   ```

   Derive the two classes son and daughter from the above class and for each, define iam() to write out similar bur appropriate messages. You should also define suitable constructors for these classes. Now write a main() that creates objects of the three classes and then calls iam() for them. Declare pointer to father. Successively, assign address of objects of the two derived classes to this pointer and in each case, call iam() through the pointer to demonstrate polymorphism in action.
9. Define a class player data members are name and nationality. Derive the class event. It is data members event type and [individual/ team] name of event. Define another derived class from player named personnel, data members are other personal information. Write a function printing for all three classes base class function is virtual function. By getting the choice we can either display event details or personal details.
10. Create a base class called **shape**. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called **triangle** and **rectangle** from the base shape. Add to the base class, a member function **get_data()** to initialize base class data members and another member function **display_area()** to compute and display the area of figure. Make display_area() as a virtual function and redefine this function in the derived classes to suit their requirements.

Use these three classes design a program that will accept dimensions of a triangle or a rectangle interactively, and display the area.
Remember the two values given as input will be treated as lengths of two sides in the case of rectangles, and base and height in the case of triangles, and used as follows
Area of Triangle=1/2*x*y
Area of rectangle=x*y

11. Extend the above program to display the area of circles. This requires addition of a new derived class 'circle' that computes the area of a circle. Remember, for a circle we need only one value, its radius, but the get_data() function in the base class requires two values to be passes.(Hint: Make the second argument of get_data() function as a default one with zero value).