1. Define the abstract class Figure with the virtual destructor and two pure virtual methods getArea and getPerimeter. Add a static method for counting the figures.

   Define the derived classes of concrete figures: Rectangle, Triangle and Circle, all with the proper attributes necessary to compute their areas and perimeters. Override the virtual methods. Implement the necessary constructors, destructors, getters, setters and exceptions.

   Define the class FigureSet with a dynamic array of pointers to figures. Implement public methods of the class for: adding a new figure of any type to the set, removing all the figures from the set, computing the summary area and the summary perimeter of the figures in the set. Overload the indexing operator ([]) for the set to have the direct access to the particular figure in the set (throwing a range check exception if necessary). Add other members which are necessary to implement the class. Write a program which tests all the class capabilities.

2. Define an abstract class of objects of the type Mammal, containing three protected attributes: name (a string), weight and age (positive real numbers), and a pure virtual method utterVoice. Make the destructor of the class virtual.

   Define two classes Cat and Whale inheriting from Mammal, with at least one additional attribute each (i.e. a color, a height or a length), and each overriding the virtual method. The classes should contain constructors enabling to set all the attributes, destructors deallocating the additional memory, and proper getters and setters. Add any necessary exception classes.

   A test program should create an array of pointers to the objects of the class Mammal, fill them in with a number of cats and whales, and check what kind of voice is uttered by each animal in the array.

   Define any other mammal classes inheriting from the existing classes and add some instances of the classes to the array defined in the test program.

3. Extend the class Clock (see the previous lab task) with the private virtual method showing the current state of the clock. Override the method in the derived classes of AlarmClock and RadioAlarmClock to show their current settings. Reimplement the shift operator (<<) of the class Clock to execute only the virtual method for the reference to a Clock object given as a parameter. Test the operator for objects of all the derived classes.