# Object-Oriented Programming (Exercise List 10)

Deadline: 07.06.2011

1. Consider the following three classes:

```
class square
{
   double centerx;
   double centery;
   double side_length;

   double area( ) const;
   double circumference( ) const;
   void print( std::ostream& const;
};

class triangle
{
   double x1, y1;
   double x2, y2;
   double x3, y3;

   double area( ) const;
   double circumference( ) const;
   void print( std::ostream& ) const;
};

class circle
{
   double centerx;
   double centery;
   double radius;

   double area( ) const;
   double circumference( ) const;
   void print( std::ostream& ) const;
}
```

Write suitable constructors for each of the classes, and complete the `area( ) const`, `circumference( ) const` and `print( ) const` methods.

2. We want to be able to put a mixture of squares, triangles, and circles in an `std::list`.

   In order to do this, define a class

   ```
   struct surface
   {
      surf* ref;

   };
   ```

   Make `square`, `triangle` and `circle` inherit from `surf`. Give `surf` a virtual `clone( )` method, and implement the concrete `clone()` methods for the subclasses.

3. Write a copy constructor, assignment operator, and a destructor, following the pattern on the slides.

4. Add a function

   ```
   std::ostream& operator << ( std::ostream& stream, const surface& s );
   ```

   according to the pattern on the slides.

5. Fill an `std::list< surface >` with a couple of surfaces, and make sure that the following code works:

   ```
   double total_surface = 0.0;
   double total_circumference = 0.0;
   for( std::list< surface > :: const_iterator
           p = list. begin( );
           p != list. end( );
           ++ p )
   {
      total_surface += p -> surface( );
      total_circumference += p -> circumference( );
      std::cout << *p << "\n";
   }

   std::cout << "total surface is " << total_surface << "\n";
   std::cout << "total circumference is " << total_circumference << "\n";
   ```

(See the rules for showing code on the course homepage. Default value of a task (when shown on time) is 3 points.)