Object-Oriented Programming (List 7)

Deadline: 10.05.2011

This exercise is about `std::map`. In the program of task list 5, we will add an explicit name comparator. (Instead of using $<$ on **names**.)

1. Define a comparator class for **name**.

   ```
   struct namecomparator
   {

   };
   ```

   which has a method

   ```
   bool operator( ) ( const name& n1, const name& n2 ) const;
   ```

   In general, I think that different classes should be in different files. But in this case, the class `namecomparator` is closely connected to `name`, so it can be added to file `name.h`.

2. In class telephonebook, replace

   ```
   std::map< name, number > table;
   ```

   by

   ```
   std::map< name, number, namecomparator > table;
   ```

   Verify that the new program compiles and that it really uses `namecomparator`, for example by putting a print function in the compare function.

3. Add a method

   ```
   bool remove( const name& n );
   ```

to `class telephonebook`. It returns **true**, if an actual element was removed, otherwise **false**.

4. Modify method `insert( )` of `telephonebook` in such a way that it returns a boolean. The boolean must be **true** if a new name was inserted, **false** if an existing telephone number was overwritten.

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