Mail log

Submission deadline:	**2019-04-21 23:59:59**
Late submission with malus:	**2019-06-30 23:59:59** (Late submission malus: 100.0000 %)
Evaluation:	**5.0000**
Max. assessment:	**5.0000** (Without bonus points)
Submissions:	2 / 20 Free retries + 20 Penalized retries (-2 % penalty each retry)
Advices:	0 / 2 Advices for free + 2 Advices with a penalty (-10 % penalty each advice)

The task is to develop class `CMailLog` which is capable of processing mailer daemon logs.

When analyzing e-mail traffic, it is common to search for an email (who sent it, when it was delivered, ...). To achieve this, mail log must be parsed first. Class `CMailLog` will be able to parse the log and solve some searches.

The input to the parser is a mail log. The format is:

```
month day year hour:minute:sec relay_name mailID: message
```

for instance:

```
Mar 29 2013 14:55:31.456 relay.fit.cvut.cz KhdfEjkl247D: from=PR-department@fit.cvut.cz
Mar 29 2013 14:50:23.233 relay.fit.cvut.cz ADFger72343D: mail undeliverable
Mar 29 2013 14:58:32.563 relay.fit.cvut.cz KhdfEjkl247D: subject=Meeting this afternoon
Mar 29 2013 15:04:18.345 relay.fit.cvut.cz KhdfEjkl247D: to=CEO@fit.cvut.cz
```

The fields are self-explanatory: month (English notation, 3 letter acronym, first in capital), day of month, year, hour, minute, second (with milliseconds), mail server DNS name, e-mail identifier (letters and digits) and a message. The mail server produces a message every time something important happens (e.g., an email is received from a client, an email is delivered to the mailbox, ...). It is common for one email to generate several log messages, however, the messages share the same mail identifier.

The aim is to match sender, subject, and receiver(s) for each e-mail delivered. Thus, we will only focus on messages starting with `from=`, `subject=`, or `to=`. We will ignore all other messages. If the log does not provide `subject=` for an e-mail, set the email subject to an empty string.

The sample above demonstrates that the log messages are nearly ordered, however there may be some lines that violate the ordering. The out-of-order messages are the messages with the `to=` message. The log first provides `from=`, then `subject=`, and finally one or more `to=`, this relative order is guaranteed for each e-mail. Next, the messages are interleaved for all e-mails being processed. Finally, there may be several `to=` messages for one e-mail. In this case, there will be several records, one for each addressee (possibly with different timestamps).

The `CMailLog` class will parse the above log. It must integrate into existing classes already present in the testing environment (classes `CTimeStamp` and `CMail`). Since the latter two classes are already implemented in the testing environment, they are presented in a conditional compile block. You do not have to implement the two classes to pass the test in Progtest, however, you will have to implement them at least partially to test your implementation ("mocking"). Please keep in mind these two classes must be kept in the conditional compile block, otherwise the program will not compile. On the other hand, you have to fully implement `CMailLog` class and keep this class outside the conditional compile block.

```
class CTimeStamp
{
  public:
    CTimeStamp ( int       year,
                 int       month,
                 int       day,
                 int       hour,
                 int       minute,
                 double    sec );

    int Compare ( const CTimeStamp & x ) const;
    friend ostream & operator << ( ostream      & os,
                                   const CTimeStamp & x );
  private:
    ...
};

class CMail
{
  public:
    CMail  ( const CTimeStamp & timeStamp,
             const string     & from,
             const string     & to,
             const string     & subject );
    int CompareByTime ( const CTimeStamp & x ) const
    int CompareByTime ( const CMail      & x ) const
    const string & From ( void ) const;
    const string & To   ( void ) const;
    const string & Subject ( void ) const;
    const CTimeStamp & TimeStamp ( void ) const;
    friend ostream & operator << ( ostream     & os,
                                   const CMail & x );
  private:
    ...
};
#endif /* __PROGTEST__ */

class CMailLog
{
  public:
    // default constructor
    // destructor
    int            ParseLog           ( istream & in );
    list<CMail>    ListMail           ( const CTimeStamp & from,
                                        const CTimeStamp & to ) const;
    set<string>    ActiveUsers        ( const CTimeStamp & from,
                                        const CTimeStamp & to ) const;
  private:
    ...
};
```

CTimeStamp

This class implements a timestamp. The interface is:

- constructor which fills-in the member variables,
- overloaded output operator which displays the timestamp in ISO format (YYYY-MM-DD HH24:MI:SS.UUU). This operator is useful for testing and debugging only,
- comparison method `Compare`. The result `a . Compare (b)` is either a positive number for `a > b`, 0 for `a == b`, or a negative number for `a < b`.

CMail

This class represents one e-mail record. The interface is:

- constructor which fills-in member variables,
- methods to retrieve the values (getters) - `From`, `Subject`, `To`, and `TimeStamp`,
- methods to compare the timestamps, the result value corresponds to the result returned by `CTimeStamp::Compare`,
- overloaded output operator (testing and debugging).

CMailLog

This class is the mail log processor itself. The interface is:

- default constructor which creates an empty instance,
- destructor to free allocated resources (if any),
- method `ParseLog`, which processes the log. The log is accessible via input stream. The method will read the stream and store the e-mail records (see above). If there is an invalid log entry (an invalid log line), the method skips the line and proceeds with the next log line. The return value is the total number of e-mail records reconstructed from the log (i.e., not the number of log lines processed),
- method `ListMail` returns a list of e-mail records that were delivered in the time interval specified (both