```
/****
 *
 * Class RolodexTest is the companion testing module for class Rolodex.  It
 * defines the following module test plan:
 *
 *    Phase 1: Unit test the constructor and destructor.
 *
 *    Phase 2: Unit test Add, saving constructed results for subsequent tests.
 *
 *    Phase 3: Unit test Print.
 *
 *    Phase 4: Unit test Delete, Change, and Find using data built with Add.
 *
 *    Phase 5: Test a series interleaved Adds and Deletes.
 *
 *    Phase 6: Repeat Phases 1 through 5.
 *
 *    Phase 7: Stress test by constructing and destructing a single rolodex of
 *             100000 cards and 100000 rolodexes of 1 card each.
 *
 * These phases are defined in the functions named Phase1 through Phase7.  The
 * unit tests are defined the functions TestXXX, where XXX is the name of a
 * Rolodex public member function.
 *
 */

#ifndef rolodexttestIncluded
#define rolodexttestIncluded

#include "rolodex.h"


class RolodexTest : public Rolodex {

  public:

    RolodexTest();
    /*
     * Construct this.
     */

    ~RolodexTest();
    /*
     * Destruct this.
     */

    void Run();
    /*
     * Run all the phases of this.
     */


  protected:

    /**
     * PhaseX functions are for each module testing phase.
     */
    void Phase1();
    /*
     * Execute test phase 1 by calling TestRolodex followed by
     * TestTildaRolodex.
     */

    void Phase2();
    /*
     * Execute test phase 2 by constructing 5 rolodexes and calling TestAdd.
     */

    void Phase3();
    /*
     * Execute test phase 3 by calling TestPrint on each of the 5 rolodexes
     * built in Phase 2.
     */

    void Phase4();
    /*
     * Execute test phase 4 by Calling TestDelete, TestChange, and TestFind.
     */

    void Phase5();
    /*
     * Execute test phase 5 by combining the logic of TestAdd and TestDelete so
     * that each added card is immediately deleted.  See the plans for TestAdd
     * and TestDelete below for further details.
     */

    void Phase6();
    /*
     * Execute test phase 6 by deleting all constructed rolodexes and
     * reinvoking Phase1 through TestPase5.
     */

    void Phase7();
    /*
     * Execute test phase 7 by constructing and destructing a single rolodex of
     * 100000 cards and 100000 roldexes of 1 card each
     */


    /**
     * Individual unit testing functions for member functions
     */
    void TestRolodex();
    /*
     * Unit test the constructor by building one rolodex.  Further constructor
     * testing will be carried out in subsequent tests.
     */

    void TestTildaRolodex();
    /*
     * Unit test the destructor by deleting one rolodex.  Further destructor
     * testing will be carried out in subsequent tests.
```

```
 */

void TestAdd();
/*
 * Unit test Add by building rolodexes of sizes 1, 2, 10, 100, and 1000.
 * Each of these is a separate rolodex object.  Validate the postcondition
 * for each of these constructed rolodexes.  Unit test the Add precondition
 * by supplying card inputs that violate each of the preconditions clauses.
 *
 *  Test
 *  Case    Input                   Output          Remarks
 * ========================================================================
 *   1      {"N1", 111111111,       Card added to   Rolodex size 1
 *           1, Male,               rolodex; no
 *           "1 One Street"}        other changes
 *
 *   2      Input to case 1 +       Cards added to  Rolodex size 2
 *          {"N2", 222222222,       rolodex; no
 *           2, Female,             other changes
 *           "2 Two Street"}
 *
 *   3      Input to case 2 +
 *          {"N3", 100000003,       Ibid.           Rolodex size 10
 *           3, Male,
 *           "333 333 333"}
 *             through
 *          {"N10", 100000010,
 *           10, Female
 *           "101010 101010 101010"}
 *
 *   4      Input to case 3 +       Ibid.           Rolodex size 100
 *          {"Ni", 100000000 + i,
 *           i mod 201, Male or Female mod i,
 *           "iii iii iii"}
 *          for i = 4 to 100.
 *
 *   5      Input to case 4 +       Ibid.           Rolodex size 1000
 *          {"Ni", 100000000 + i,
 *           i mod 201, Male or Female mod i,
 *           "iii iii iii"}
 *          for i = 101 to 1000.
 *
 *   6      {"1234567890\           AddInputErrors  All precond clauses
 *            1234567890\           exception       violated, id = 0,
 *            12345678901"          thrown, with    age < 0.
 *            0, 0,                 all error msgs
 *            InvalidSex,           non-NULL
 *            "1234567890\
 *             1234567890\
 *             1234567890\
 *             12345678901"}
 *
 *   7      {"1234567890\           AddInputErrors  All precond clauses
 *            1234567890\           exception       violated, id >
 *            12345678901",         thrown, with    999999999, age > 200.
 *            0, 1000000000,        all error msgs
 *            InvalidSex            non-NULL
 *            "1234567890\
 *             1234567890\
 *             1234567890\
 *             12345678901"}
 *
 */

void TestAdd1();
void TestAdd2();
void TestAdd10();
void TestAdd100();
void TestAdd1000();
/*
 * Build and test rolodexes of size 1, 2, 10, 100, and 1000.
 */

void AddCards(Rolodex* r, int n);
/*
 * Add n cards to the given rolodex.  This funtion is called as a helper
 * from TestAdd10, TestAdd100, and TestAdd1000.
 */

void TestAddNoDump();
/*
 * Version of TestAdd with no dumping of header messages or results.  Used
 * in other unit test plans to construct testing rolodexes.
 */

void TestPrint();
/*
 * Unit test Print by printing the five rolodexes constructed in TestAdd.
 */

void TestDelete();
/*
 * Unit test Delete by deleting each card in the five rolodexes constructed
 * in TestAdd, performing the deletion in the same order the cards were
 * added.  Reinvoke TestAdd and perform the deletions in the reverse order
 * that the cards were added.  Reinvoke TestAdd and peform the deletions in
 * a random order.
 */

void TestDeleteForward();
void TestDeleteBackward();
void TestDeleteRandom();
/*
 * Delete in forward add order, reverse add order, and random order.
 */

void TestChange();
/*
 * Unit test Change by reinvoking case 5 of TestAdd and changing the name,
 * id, age, sex, and address fields to legal values in the first card of
 * the 100 element rolodex.  Then change cards 2 through 31 by changing all
 * possible combinations of card fields values to legal values.
```

```
225       */
226
227      void TestFind();
228      /*
229       * Unit test Find by reinvoking TestAdd and finding each card in the order
230       * added, reverse order, and random order.  Then construct a rolodex of 100
231       * cards, with 1 card with a unique name, 5 cards of the same name, another
232       * 10 cards of a different name, and the remaining cards of the same name.
233       * Find all cards in this new rolodex in the order added, reverse order,
234       * and random order.
235       */
236
237      /**
238       * Testing utility functions.
239       */
240
241      void DumpPhaseHeader(int phasenum);
242      /*
243       * Output a header message to stdout identifying the test phase.
244       */
245
246      void DumpUnitTestHeader(char* testname);
247      /*
248       * Output a header message to stdout for the unit test of the given
249       * testname.
250       */
251
252      void DumpUnitTestNumber(char* msg, int testnum);
253      /*
254       * Output a header message to stdout for the unit test of the given number.
255       */
256
257      void DumpPhaseEndSpacing();
258      /*
259       * Dump three blank lines following a test phase.
260       */
261
262      void DumpUnitTestEndSpacing();
263      /*
264       * Dump a couple blank lines following a unit test.
265       */
266
267      void Dump(int i, char* message = NULL);
268      /*
269       * Dump the data values of the ith rolodex (i.e, rs[i]) to stdout.  For
270       * test validation purposes, include a print of the number of elements in
271       * the dumped rolodex.  Precede the dump with the given message, if the
272       * message is non-null.
273       */
274
275      void Dump(char* message = NULL);
276      /*
277       * Dump the data values of the all 5 rolodexes to stdout.  Precede the dump
278       * with the given message, if the message is non-null.
279       */
280
281
282      /**
283       * Data to support tests.
284       */
285      Rolodex* rs[100000];        // Storage for testing rolodexes
286
287  };
288
289  #endif
```