```
/****
 *
 * Implementation of rolodex-test.h.
 *
 */

#include "rolodex-test.h"
#include <stdio.h>
#ifndef HP700
#include <stdlib.h>
#endif


RolodexTest::RolodexTest() {
}

RolodexTest::~RolodexTest() {
}

void RolodexTest::Run() {
    Phase1();
    Phase2();
    Phase3();
    Phase4();
    Phase5();
    Phase6();
    Phase7();
}

void RolodexTest::Phase1() {
    DumpPhaseHeader(1);
    TestRolodex();
    TestTildaRolodex();
    DumpPhaseEndSpacing();
}

void RolodexTest::Phase2() {
    int i;                              // Loop var

    DumpPhaseHeader(2);

    /*
     * Construct 5 empty rolodexes for storing phase 2 results.
     */
    for (i = 0; i < 5; i++) {
        rs[i] = new Rolodex();
    }

    /*
     * Call TestAdd.
     */
    TestAdd();

    DumpPhaseEndSpacing();
}

void RolodexTest::Phase3() {
    DumpPhaseHeader(3);
    TestPrint();
    DumpPhaseEndSpacing();
}

void RolodexTest::Phase4() {
    DumpPhaseHeader(4);
    TestDelete();
    TestChange();
    TestFind();
    DumpPhaseEndSpacing();
}

void RolodexTest::Phase5() {
    DumpPhaseHeader(5);
    TestAddDeleteSeries();
    DumpPhaseEndSpacing();
}

void RolodexTest::Phase6() {
    DumpPhaseHeader(6);
    Phase1();
    Phase2();
    Phase3();
    Phase4();
    Phase5();
    DumpPhaseEndSpacing();
}

void RolodexTest::Phase7() {
    DumpPhaseHeader(7);
    StressTest();
    DumpPhaseEndSpacing();
}

void RolodexTest::TestRolodex() {
    DumpUnitTestHeader("Constructor");
    DumpUnitTestEndSpacing();
    rs[0] = new Rolodex();
}

void RolodexTest::TestTildaRolodex() {
    DumpUnitTestHeader("Destructor");
    delete rs[0];
    DumpUnitTestEndSpacing();
}

void RolodexTest::TestAdd() {
    DumpUnitTestHeader("Add");

    TestAdd1();
    Dump(0, "Dump of a 1 element rolodex should follow");
    TestAdd2();
    Dump(1, "Dump of a 2 element rolodex should follow");
    TestAdd10();
```

```
    Dump(2, "Dump of a 10 element rolodex should follow");
    TestAdd100();
    Dump(3, "Dump of a 100 element rolodex should follow");
    TestAdd1000();
    Dump(4, "Dump of a 1000 element rolodex should follow");

    DumpUnitTestEndSpacing();
}

void RolodexTest::TestAddNoDump() {
    TestAdd1();
    TestAdd2();
    TestAdd10();
    TestAdd100();
    TestAdd1000();
}

void RolodexTest::TestAdd1() {
    rs[0]->Add(new Card(
        new String("N1"),               // Name
        111111111,                      // Id
        1,                              // Age
        Male,                           // Sex
        new String("1 One Street")      // Address
    ));
}

void RolodexTest::TestAdd2() {
    rs[1]->Add(new Card(
        new String("N1"),               // Name
        111111111,                      // Id
        1,                              // Age
        Male,                           // Sex
        new String("1 One Street")      // Address
    ));

    rs[1]->Add(new Card(
        new String("N2"),               // Name
        222222222,                      // Id
        2,                              // Age
        Female,                         // Sex
        new String("2 Two Street")      // Address
    ));
}

void RolodexTest::TestAdd10() {
    AddCards(rs[2], 10);
}

void RolodexTest::TestAdd100() {
    AddCards(rs[3], 100);
}

void RolodexTest::TestAdd1000() {
    AddCards(rs[4], 1000);
}


void RolodexTest::AddCards(Rolodex* r, int n) {
    int i;                              // Loop var
    char nbuf[50];                      // Temp for name
    char abuf[50];                      // Temp for age
    Sex s;                              // Temp for sex
    char adbuf[50];                     // Temp for address

    for (i = 1; i <= n; i++) {
        sprintf(nbuf, "N%d", i);
        sprintf(adbuf, "%d %d %d", i, i, i);

        r->Add(new Card(
            new String(nbuf),           // Name
            100000000 + i,              // Id
            i % 201,                    // Age
            i % 2 ? Male : Female,      // Sex
            new String(adbuf)           // Address
        ));
    }
}

void RolodexTest::TestPrint() {
    int i;                              // Loop var

    DumpUnitTestHeader("Print");

    for (i = 0; i < 5; i++) {
        rs[i]->Print();
    }

    DumpUnitTestEndSpacing();
}

void RolodexTest::TestDelete() {
    DumpUnitTestHeader("Delete");

    TestDeleteForward();
    Dump("Dump of 5 empty rololdexes should follow");

    TestAddNoDump();
    TestDeleteBackward();
    Dump("Dump of 5 empty rololdexes should follow");

    TestAddNoDump();
    TestDeleteRandom();
    Dump("Dump of 5 empty rololdexes should follow");

    DumpUnitTestEndSpacing();
}

void RolodexTest::TestDeleteForward() {
    int i;                              // Loop var

    try {
        rs[0]->Delete(111111111);
```

```
225         rs[1]->Delete(111111111);
226         rs[1]->Delete(222222222);
227
228         for (i = 100000001; i <= 100001000; i++) {
229             if (i <= 100000010) {
230                 rs[2]->Delete(i);
231             }
232
233             if (i <= 100000100) {
234             rs[3]->Delete(i);
235             }
236
237             rs[4]->Delete(i);
238         }
239     }
240     catch (DeleteInputError* die) {
241         printf("RolodexTest::TestDeleteForward aborted ");
242         printf("on thrown exception from Rolodex::Delete\n");
243     }
244 }
245
246 void RolodexTest::TestDeleteBackward() {
247     int i;                              // Loop var
248
249     for (i = 100001000; i >= 100000001; i--) {
250         rs[4]->Delete(i);
251
252         if (i <= 100000100) {
253         rs[3]->Delete(i);
254         }
255
256         if (i <= 100000010) {
257             rs[2]->Delete(i);
258         }
259     }
260
261     rs[1]->Delete(222222222);
262     rs[1]->Delete(111111111);
263     rs[0]->Delete(111111111);
264 }
265
266 \&. . .
267
268 void RolodexTest::DumpPhaseHeader(int phasenum) {
269     printf("**** Rolodex Testing Phase %d ****\n", phasenum);
270 }
271
272 void RolodexTest::DumpUnitTestHeader(char* testname) {
273     printf("** Rolodex Unit Test %s **\n", testname);
274 }
275
276 void RolodexTest::DumpPhaseEndSpacing() {
277     printf("\n\n\n");
278 }
279
280 void RolodexTest::DumpUnitTestEndSpacing() {
281     printf("\n\n");
282 }
283
284 void RolodexTest::Dump(int i, char* message) {
285     int l;                          // Temp
286
287     if (message) {
288         printf("* %s *\n", message);
289     }
290
291     printf("Contents of test rolodex number %d (contains %d %s):\n",
292         i, l = ((RolodexTest*)rs[i])->cl->Len(), l != 1 ? "cards" : "card");
293     rs[i]->Dump();
294     printf("\n");
295 }
296
297 void RolodexTest::Dump(char* message) {
298     int i;                                  // Loop var
299
300     if (message) {
301         printf("* %s *\n", message);
302     }
303
304     for (i = 0; i < 5; i++) {
305         Dump(i);
306     }
307 }
```