```
/*-----------------------------------------------------------------
Copyright (c) 2018 Author: Jagadeesh Vasudevamurthy
file: intmatrix2test.cpp

On linux:
g++ intmatrix2.cpp intmatrix2test.cpp
valgrind a.out

-23174-- REDIR: 0x3e7b87b800 (libc.so.6:free) redirected to 0x4a06acd (free)
==23174==
==23174== HEAP SUMMARY:
==23174==     in use at exit: 0 bytes in 0 blocks
==23174==   total heap usage: 149 allocs, 149 frees, 75,520 bytes allocated
==23174==
==23174== All heap blocks were freed -- no leaks are possible
==23174==
==23174== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 4 from 4)
--23174--
--23174-- used_suppression:      4 U1004-ARM-_dl_relocate_object /tools/baton/ ↵
  valgrind/3.12.0/lib/valgrind/default.supp:1413
==23174==
==23174== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 4 from 4)
[jag@xsjrdevl100 intmatrix2]$

-----------------------------------------------------------------*/

/*-----------------------------------------------------------------
This file test intmatrix2 object
-----------------------------------------------------------------*/

/*-----------------------------------------------------------------
All includes here
-----------------------------------------------------------------*/
#include "intmatrix2.h"

/*-----------------------------------------------------------------
test init and fini
-----------------------------------------------------------------*/
void test_init_fini() {
  intmatrix2 a;
  cout << "Matrix a" << endl;
  cout << a << endl;
  intmatrix2 b(3, 4);
  cout << "Matrix b" << endl;
  cout <<  b << endl;
  intmatrix2 c(2, 0, 7);
  cout << "Matrix c"  << endl;
  cout << c << endl;
  intmatrix2 d(0, 10, 7);
```

```
49    cout << "Matrix d" << endl;
50    cout <<  d << endl;
51    intmatrix2 e(3, 10, 7);
52    cout << "Matrix e"<< endl;
53    cout << e << endl;
54    intmatrix2 f("1 2|3 4|5 6");
55    cout << "Matrix f" << endl;
56    cout << f << endl;
57    intmatrix2 g(" 1 2|3 4 |5 6 ");
58    cout << "Matrix g"<< endl;
59    cout <<  g << endl;
60    assert(f.isEqual(g));
61    assert(g.isEqual(f));
62
63    intmatrix2 h(" 1 2 |3 4 |5 6 8 ");
64    cout << "Matrix h" << endl;
65    cout << h << endl;
66    assert(h.isEmpty());
67    assert(!f.isEqual(h));
68    h = a = f = e = d = c = b = g ;
69    cout << "Matrix g" << endl;
70    cout <<  g << endl;
71    cout << "Matrix h" << endl;
72    cout <<  h << endl;
73    cout << "Matrix e" << endl;
74    cout <<  e << endl;
75  }
76
77  /*----------------------------------------------------------------
78  test add1
79  ----------------------------------------------------------------*/
80  void test_add1(const char* as, const char* bs, const char* anss) {
81    intmatrix2 a(as);
82    cout << "Matrix a" << endl;
83    cout <<  a << endl;
84    intmatrix2 b(bs);
85    cout << "Matrix b" << endl;
86    cout <<  b << endl;
87    intmatrix2 s = a.add(b);
88    cout << "Matrix s" << endl;
89    cout <<  s << endl;
90    intmatrix2 ans(anss);
91    cout << "Matrix expected ans" << endl;
92    cout <<  ans << endl;
93    assert(s.isEqual(ans));
94    assert(ans.isEqual(s));
95  }
96
97  /*----------------------------------------------------------------
```

```
 98  test add
 99  ----------------------------------------------------------------------*/
100  void test_add() {
101    test_add1("7 9 11|13 15 17 ", " 6 8 10| 12 14 16 ", "13 17 21 | 25 29 33");
102    test_add1("1 2 3|4 5  6 ", "1 2  ", "");
103  }
104
105  /*----------------------------------------------------------------
106  test mult1
107  ----------------------------------------------------------------------*/
108  void test_mult1(const char* as, const char* bs, const char* anss) {
109    intmatrix2 a(as);
110    cout << "Matrix a" << endl;
111    cout <<  a << endl;
112    intmatrix2 b(bs);
113    cout << "Matrix b" << endl;
114    cout <<  b << endl;
115    intmatrix2 s = a.mult(b);
116    cout << "Matrix s" << endl;
117    cout <<  s << endl;
118    intmatrix2 ans(anss);
119    cout << "matrix expected ans" << endl;
120    cout <<  ans << endl;
121    assert(s.isEqual(ans));
122    assert(ans.isEqual(s));
123    cout << "-----------------------------------\n";
124  }
125
126  /*----------------------------------------------------------------
127  test mult
128  ----------------------------------------------------------------------*/
129  void test_mult() {
130    test_mult1("1 2 3", " 2 1 3 | 3 3 2| 4 1 2  ", "20 10 13");
131    test_mult1("3 4 2", "13 9 7 15|8 7 4 6| 6 4 0 3 ", "83 63 37 75");
132    test_mult1("3", "5 2 11|9 4 14", "15 6 33|27 12 42");
133    test_mult1("5 2 11|9 4 14", "3 ", "15 6 33|27 12 42");
134    const char* a = "3 9 0 2 2 9 5 2|0 2 2 1 9 6 6 8|7 5 6 1 4 9 8 9|3 3 2 9 2 1 ↵
         7 4|1 9 0 1 2 9 5 2|4 2 0 3 7 3 9 1|5 9 0 6 6 7 8 2|9 3 4 6 8 4 9 1";
135    const char* b = "6 1 6 0 8 3 0 0|6 8 9 0 6 6 7 2|4 8 2 0 5 4 6 7|2 4 4 2 2 6 ↵
         9 8|4 8 2 2 4 6 4 1|1 5 5 6 4 7 5 5|7 4 6 5 0 6 5 3|2 3 7 0 1 3 8 5";
136    const char* s = "132 170 200  87 128 186 175 106|122 186 166  86  92 182 195 ↵
         123|197 235 267 104 179 243 253 178|128 140 164  63  86 162 194 140|118 164 ↵
          184  85 110 174 166  98|138 142 144  83  91 162 137  82|187 222 244 106   ↵
         160 244 232 141|201 212 210  97 171 230 204 142";
137    test_mult1(a, b, s);
138    test_mult1("7 3|2 5 | 6 8| 9 0", "8 14 0 3 1|7 11 5 91 3|8 4 19 5 57", "");
139  }
140
141
```

```
142 /*-------------------------------------------------------------------
143 test bed
144 -------------------------------------------------------------------*/
145 void testbed() {
146
147     //intmatrix2::makeShowOn() ;
148   test_init_fini();
149   test_add();
150   test_mult();
151 }
152
153 /*-------------------------------------------------------------------
154 main
155 -------------------------------------------------------------------*/
156 int main() {
157   testbed();
158   cin.get();
159   return 0;
160 }
161
162 //EOF
163
164
165 #if 0
166 /*
167 Matrix a
168 Empty Matrix
169
170 Matrix b
171 0       0       0       0
172 0       0       0       0
173 0       0       0       0
174
175 Matrix c
176 Empty Matrix
177
178 Matrix d
179 Empty Matrix
180
181 Matrix e
182 7       7       7       7       7       7       7       7       7       7
183 7       7       7       7       7       7       7       7       7       7
184 7       7       7       7       7       7       7       7       7       7
185
186 Matrix f
187 1       2
188 3       4
189 5       6
190
```

```
191 Matrix g
192 1       2
193 3       4
194 5       6
195
196 Matrix h
197 Empty Matrix
198
199 Matrix g
200 1       2
201 3       4
202 5       6
203
204 Matrix h
205 1       2
206 3       4
207 5       6
208
209 Matrix e
210 1       2
211 3       4
212 5       6
213
214 Matrix a
215 7       9       11
216 13      15      17
217
218 Matrix b
219 6       8       10
220 12      14      16
221
222 Matrix s
223 13      17      21
224 25      29      33
225
226 Matrix expected ans
227 13      17      21
228 25      29      33
229
230 Matrix a
231 1       2       3
232 4       5       6
233
234 Matrix b
235 1       2
236
237 Can't do addition as matrix dimensions don't match
238 Matrix s
239 Empty Matrix
```

```
240
241 Matrix expected ans
242 Empty Matrix
243
244 Matrix a
245 1       2       3
246
247 Matrix b
248 2       1       3
249 3       3       2
250 4       1       2
251
252 Matrix s
253 20      10      13
254
255 matrix expected ans
256 20      10      13
257
258 ----------------------------------
259 Matrix a
260 3       4       2
261
262 Matrix b
263 13      9       7       15
264 8       7       4       6
265 6       4       0       3
266
267 Matrix s
268 83      63      37      75
269
270 matrix expected ans
271 83      63      37      75
272
273 ----------------------------------
274 Matrix a
275 3
276
277 Matrix b
278 5       2       11
279 9       4       14
280
281 Matrix s
282 15      6       33
283 27      12      42
284
285 matrix expected ans
286 15      6       33
287 27      12      42
288
```

```
289 ---------------------------------
290 Matrix a
291 5       2       11
292 9       4       14
293
294 Matrix b
295 3
296
297 Matrix s
298 15      6       33
299 27      12      42
300
301 matrix expected ans
302 15      6       33
303 27      12      42
304
305 ---------------------------------
306 Matrix a
307 3       9       0       2       2       9       5       2
308 0       2       2       1       9       6       6       8
309 7       5       6       1       4       9       8       9
310 3       3       2       9       2       1       7       4
311 1       9       0       1       2       9       5       2
312 4       2       0       3       7       3       9       1
313 5       9       0       6       6       7       8       2
314 9       3       4       6       8       4       9       1
315
316 Matrix b
317 6       1       6       0       8       3       0       0
318 6       8       9       0       6       6       7       2
319 4       8       2       0       5       4       6       7
320 2       4       4       2       2       6       9       8
321 4       8       2       2       4       6       4       1
322 1       5       5       6       4       7       5       5
323 7       4       6       5       0       6       5       3
324 2       3       7       0       1       3       8       5
325
326 Matrix s
327 132     170     200     87      128     186     175     106
328 122     186     166     86      92      182     195     123
329 197     235     267     104     179     243     253     178
330 128     140     164     63      86      162     194     140
331 118     164     184     85      110     174     166     98
332 138     142     144     83      91      162     137     82
333 187     222     244     106     160     244     232     141
334 201     212     210     97      171     230     204     142
335
336 matrix expected ans
337 132     170     200     87      128     186     175     106
```

```
122     186     166     86      92      182     195     123
197     235     267     104     179     243     253     178
128     140     164     63      86      162     194     140
118     164     184     85      110     174     166     98
138     142     144     83      91      162     137     82
187     222     244     106     160     244     232     141
201     212     210     97      171     230     204     142

---------------------------------
Matrix a
7       3
2       5
6       8
9       0

Matrix b
8       14      0       3       1
7       11      5       91      3
8       4       19      5       57

Can't do multiplication as matrix dimensions aren't correct
Matrix s
Empty Matrix

matrix expected ans
Empty Matrix

---------------------------------
*/
#endif

```