```cpp
/*-----------------------------------------------------------------
Copyright (c) 2018 Author: Jagadeesh Vasudevamurthy
file: iset64test.cpp

On linux:
g++ iset64.cpp iset64test.cpp
valgrind a.out

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

/*-----------------------------------------------------------------
This file test iset64 object
-----------------------------------------------------------------*/

/*-----------------------------------------------------------------
All includes here
-----------------------------------------------------------------*/
#include "iset64.h"

/*-----------------------------------------------------------------
test a set
-----------------------------------------------------------------*/
void test_basic() {
  iset64 a;
  cout << "a = " << a << endl;
  a = a + 5;
  cout << "set a after adding 5 = " << a << endl;
  a = a + 5;
  cout << "set a after adding 5 = " << a << endl;
  a += 63;
  a += 0;
  cout << "set a after adding 0 and 63 = " << a << endl;
  int x[] = { 1, 3, 6 };
  iset64 b(x, sizeof(x) / sizeof(int));
  cout << "set b = " << b << endl;
  b = b - 3;
  cout << "set b after removing 3 = " << b << endl;
  b = b - 3;
  cout << "set b after removing 3 = " << b << endl;
  b = b - 10;
  cout << "set b after removing 10 = " << b << endl;
  b = b - 6;
  cout << "set b after removing 6 = " << b << endl;
  b = b - 1;
  cout << "set b after removing 1 = " << b << endl;
  b = b + 10;
  b = b + 2;
  cout << "set b after adding {10,2} = " << b << endl;
  {
```

```
50      const int N = 5 ;
51      iset64 *a = new iset64[N] ;
52      iset64 t;
53      t = t + 0 + 1 + 62 ;
54      a[0] = a[1] = a[2] = a[3] = a[4] = t ;
55      for (int i = 0; i < N; ++i) {
56        cout << "a[" << i << "]=" << a[i] << endl ;
57      }
58      delete [] a ;
59    }
60  }
61
62  /*-----------------------------------------------------------------
63  test union
64  -----------------------------------------------------------------*/
65  void test_union() {
66    {
67      cout << "TESTING: iset64 operator+(const iset64& a, const iset64& b)" <<
          endl;
68      iset64 a;
69      a += 1;
70      a += 2;
71      iset64 b;
72      b += 1;
73      b += 2;
74      b += 3;
75      cout << "Set a " << a << endl;
76      cout << "Set b " << b << endl;
77      iset64 c = a + b;
78      cout << "a + b = " << c << endl;
79    }
80    {
81      cout << "TESTING:iset64 operator+(const iset64& a, const int b)" << endl;
82      iset64 a;
83      a += 1;
84      a += 2;
85      cout << a << endl;
86      a = a + 1;
87      cout << "{1,2} + 1 = " << a << endl;
88      a += 1;
89      a += 2;
90      cout << a << endl;
91      a = a + 3;
92      cout << "{1,2} + 3 = " << a << endl;
93    }
94    {
95      cout << "TESTING:iset64 operator+(const int b, const iset64& a)" << endl;
96      iset64 a;
97      a += 1;
```

```
 98      a += 2;
 99      cout << "Set a " << a << endl;
100      a = 1 + a;
101      cout << " 1 + {1,2} = " << a << endl;
102      a += 1;
103      a += 2;
104      cout << "Set a " << a << endl;
105      a = 3 + a;
106      cout << " 3 + {1,2}  = " << a << endl;
107    }
108
109    {
110      cout << "TESTING:iset64& iset64::operator+=(const iset64& a)" << endl;
111      iset64  b;
112      b += 1;
113      b += 2;
114      iset64 a;
115      a += 1;
116      a += 3;
117      cout << "Set b " << b << endl;
118      cout << "Set a " << a << endl;
119      b += a;
120      cout << " {1,2} + {1,3}  = " << b << endl;
121    }
122    {
123      cout << "iset64& iset64::operator+=(const int b)" << endl;
124      iset64 a;
125      a += 1;
126      a += 2;
127      cout << "Set a " << a << endl;
128      a += 3;
129      cout << " {1,2} + 3  = " << a << endl;
130    }
131    {
132      //test chaining
133      iset64 a;
134      a += 1;
135      a += 2;
136      iset64 b;
137      b += 3;
138      b += 4;
139      iset64 c;
140      c += 7;
141      c += 8;
142      iset64 d = a + b + c + 5;
143      cout << "Set a " << a << endl;
144      cout << "Set b " << b << endl;
145      cout << "Set c " << c << endl;
146      cout << "Set d " << d << endl;
```

```
147   }
148 }
149
150 /*--------------------------------------------------------------
151 test difference
152 --------------------------------------------------------------*/
153 void test_difference() {
154   {
155     cout << "TESTING: iset64 operator-(const iset64& a, const iset64& b)" <<
        endl;
156     iset64 a;
157     a += 1;
158     a += 2;
159     iset64 b;
160     b += 1;
161     b += 2;
162     iset64 c = a - b;
163     cout << "Set a " << a << endl;
164     cout << "Set b " << a << endl;
165     cout << "a - b = " << c << endl;
166   }
167   {
168     cout << "TESTING: iset64 operator-(const iset64& a, const iset64& b)" <<
        endl;
169     iset64 a;
170     a += 1;
171     a += 5;
172     iset64 b;
173     b += 1;
174     b += 2;
175     b += 3;
176     iset64 c = a - b;
177     cout << "Set a " << a << endl;
178     cout << "Set b " << b << endl;
179     cout << "a - b = " << c << endl;
180   }
181
182   {
183     cout << "TESTING: iset64 operator-(const iset64& a, const int b)" << endl;
184     iset64 a;
185     a += 1;
186     a += 2;
187     cout << "Set a " << a << endl;
188     a = a - 3;
189     cout << "a - 3 = " << a << endl;
190   }
191
192   {
193     cout << "TESTING: iset64 operator-(const int b, const iset64& a)" << endl;
```

```
194     iset64 a;
195     a += 1;
196     a += 2;
197     cout << "Set a " << a << endl;
198     a = 3 - a;
199     cout << "3 - a = " << a << endl;
200   }
201
202   {
203     cout << "TESTING: iset64& iset64::operator-=(const iset64& a)" << endl;
204     iset64 a;
205     a += 1;
206     a += 3;
207     iset64 b;
208     b += 1;
209     b += 2;
210     cout << "Set a " << a << endl;
211     cout << "Set b " << b << endl;
212     b -= a;
213     cout << "b -= a = " << b << endl;
214   }
215
216   {
217     cout << "TESTING: iset64& iset64::operator-=(const int b)" << endl;
218     iset64 a;
219     a += 1;
220     a += 2;
221     cout << "Set a " << a << endl;
222     a -= 3;
223     cout << "a -= 3 = " << a << endl;
224   }
225   {
226     //test chaining
227     iset64 a;
228     a += 1;
229     a += 2;
230     iset64 b;
231     b += 2;
232     b += 4;
233     iset64 c;
234     c += 2;
235     c += 8;
236     iset64 d = a - b - c + 5;
237     cout << "Set a " << a << endl;
238     cout << "Set b " << b << endl;
239     cout << "Set c " << c << endl;
240     cout << "Set d " << d << endl;
241   }
242 }
```

```
243
244 /*-------------------------------------------------------------
245 test intersection
246 -------------------------------------------------------------*/
247 void test_intersection() {
248   {
249     cout << "TESTING: iset64 operator*(const iset64& a, const iset64& b)" <<
        endl;
250     iset64 a;
251     a += 1;
252     a += 2;
253     iset64 b;
254     b += 1;
255     b += 2;
256     b += 3;
257     cout << "Set a " << a << endl;
258     cout << "Set b " << b << endl;
259     iset64 c = a * b;
260     cout << "a * b = " << c << endl;
261   }
262   {
263     cout << "TESTING:iset64 operator*(const iset64& a, const int b)" << endl;
264     iset64 a;
265     a += 1;
266     a += 2;
267     cout << "Set a " << a << endl;
268     a = a * 1;
269     cout << "{1,2} * 1 = " << a << endl;
270     a += 1;
271     a += 2;
272     cout << "Set a " << a << endl;
273     a = a * 3;
274     cout << "{1,2} * 3 = " << a << endl;
275   }
276   {
277     cout << "TESTING:iset64 operator*(const int b, const iset64& a)" << endl;
278     iset64 a;
279     a += 1;
280     a += 2;
281     cout << "Set a " << a << endl;
282     a = 1 * a;
283     cout << " 1 * {1,2} = " << a << endl;
284     a += 1;
285     a += 2;
286     cout << "Set a " << a << endl;
287     a = 3 * a;
288     cout << " 3 * {1,2}  = " << a << endl;
289   }
290
```

```
291   {
292     cout << "TESTING:iset64& iset64::operator*=(const iset64& a)" << endl;
293     iset64 b;
294     b += 1;
295     b += 2;
296     iset64 a;
297     a += 1;
298     a += 3;
299     cout << "Set b " << b << endl;
300     cout << "Set a " << a << endl;
301     b *= a;
302     cout << " {1,2} * {1,3}  = " << b << endl;
303   }
304   {
305     cout << "iset64& iset64::operator*=(const int b)" << endl;
306     iset64 a;
307     a += 1;
308     a += 2;
309     cout << "Set a " << a << endl;
310     a *= 3;
311     cout << " {1,2} * 3  = " << a << endl;
312   }
313   {
314     //test chaining
315     iset64 a;
316     a += 1;
317     a += 2;
318     iset64 b;
319     b += 2;
320     b += 4;
321     iset64 c;
322     c += 2;
323     c += 8;
324     iset64 d = a * b * c + 5;
325     cout << "Set a " << a << endl;
326     cout << "Set b " << b << endl;
327     cout << "Set c " << c << endl;
328     cout << "Set d " << d << endl;
329   }
330 }
331
332
333 /*-----------------------------------------------------------------
334 test equal
335 -----------------------------------------------------------------*/
336 void test_equal_not_equal() {
337   {
338     cout << "TESTING: bool operator==(const iset64& a, const iset64& b)" <<
        endl;
```

```
339     iset64 a;
340     a += 1;
341     a += 2;
342     iset64 b;
343     b += 1;
344     b += 2;
345     cout << "Set a " << a << endl;
346     cout << "Set b " << b << endl;
347     cout << "a == b " << boolalpha << (a == b) << endl;
348     b -= 1;
349     cout << a;
350     cout << b;
351     cout << "a == b " << boolalpha << (a == b) << endl;
352   }
353   {
354     cout << "TESTING: bool operator!=(const iset64& a, const iset64& b)" <<
        endl;
355     iset64 a;
356     a += 1;
357     a += 2;
358     iset64 b;
359     b += 1;
360     b += 2;
361     cout << "Set a " << a << endl;
362     cout << "Set b " << b << endl;
363     cout << "a != b " << boolalpha << (a != b) << endl;
364     b -= 1;
365     cout << "Set a " << a << endl;
366     cout << "Set b " << b << endl;
367     cout << "a != b " << boolalpha << (a != b) << endl;
368   }
369 }
370
371 /*-------------------------------------------------------------------
372 ++ and --
373 -------------------------------------------------------------------*/
374 void test_pre_post_inr_dec() {
375   {
376     int x[] = { 1, 2, 63 };
377     iset64 a(x, sizeof(x) / sizeof(int));
378     cout << "a = " << a << endl;
379     ++a;
380     cout << "++a = " << a << endl;
381     int y[] = { 2, 3, 0 };
382     iset64 b(y, sizeof(y) / sizeof(int));
383     assert(a == b);
384   }
385   {
386     int x[] = { 1, 2, 63 };
```

```
387       iset64 a(x, sizeof(x) / sizeof(int));
388       cout << "a = " << a << endl;
389       iset64 acopy(x, sizeof(x) / sizeof(int));
390       cout << "acopy = " << acopy << endl;
391       iset64 rhs = a++;
392       assert(rhs == acopy);
393       cout << "a++ = " << a << endl;
394       cout << "rhs = " << rhs << endl;
395       int y[] = { 2, 3, 0 };
396       iset64 b(y, sizeof(y) / sizeof(int));
397       assert(a == b);
398     }
399     {
400       int x[] = { 0,2,63 };
401       iset64 a(x, sizeof(x) / sizeof(int));
402       cout << "a = " << a << endl;
403       --a;
404       cout << "--a = " << a << endl;
405       int y[] = { 63, 1, 62 };
406       iset64 b(y, sizeof(y) / sizeof(int));
407       assert(a == b);
408     }
409     {
410       int x[] = { 0, 2, 63 };
411       iset64 a(x, sizeof(x) / sizeof(int));
412       cout << "a = " << a << endl;
413       iset64 acopy(x, sizeof(x) / sizeof(int));
414       cout << "acopy = " << acopy << endl;
415       iset64 rhs = a--;
416       assert(rhs == acopy);
417       cout << "a-- = " << a << endl;
418       cout << "rhs = " << rhs << endl;
419       int y[] = { 63, 1, 62 };
420       iset64 b(y, sizeof(y) / sizeof(int));
421       assert(a == b);
422     }
423   }
424
425   /*----------------------------------------------------------------
426   ~
427   Complement of a set.
428   The complement of A is the set of all element in the universal set U, but not ↵
        in A.
429   a = {0,2,63}
430   x = ~a
431   {1,3,...,62}
432   ----------------------------------------------------------------*/
433   void test_complement() {
434     {
```

```
435     int x[] = { 0, 2, 63 };
436     iset64 a(x, sizeof(x) / sizeof(int));
437     cout << "a = " << a << endl;
438     iset64 nota = (~a);
439     cout << "~a = " << nota << endl;
440     iset64 ans;
441     ans += 1;
442     for (int i = 3; i < 63; ++i) {
443       ans += i;
444     }
445     cout << "ans = " << ans << endl;
446     assert(nota == ans);
447     ans = ~ans;
448     cout << "~ans = " << ans << endl;
449     assert(ans == a);
450   }
451 }
452
453 /*----------------------------------------------------------------
454 a = {0,2,63}
455 if (a) {
456
457 }
458 ----------------------------------------------------------------*/
459 void test_conversion_operator() {
460   int x[] = { 0, 2, 63 };
461   iset64 a(x, sizeof(x) / sizeof(int));
462   cout << "a = " << a << endl;
463   if (a) {
464     cout << "a exists\n";
465   } else {
466     cout << "a does not exists\n";
467   }
468   iset64 b;
469   cout << "b = " << b << endl;
470   if (b) {
471     cout << "b exists\n";
472   } else {
473     cout << "b does not exists\n";
474   }
475 }
476
477 /*----------------------------------------------------------------
478 a = {0,2,63}
479 if (!a) {
480
481 }
482 ----------------------------------------------------------------*/
483 void test_not_operator() {
```

```
484   int x[] = { 0, 2, 63 };
485   iset64 a(x, sizeof(x) / sizeof(int));
486   cout << "a = " << a << endl;
487   if (!a) {
488     cout << "a does not exists\n";
489   } else {
490     cout << "a exists\n";
491   }
492   iset64 b;
493   cout << "b = " << b << endl;
494   if (!b) {
495     cout << "b does not exists\n";
496   } else {
497     cout << "b exists\n";
498   }
499 }
500
501 /*-----------------------------------------------------------------
502 (a+b)' = a'. b'
503 (a.b)' = a' + b'
504 -----------------------------------------------------------------*/
505 void test_demorgan_laws(const int x[], int lx, const int y[], int ly) {
506   {
507     iset64 a(x, lx);
508     cout << "a = " << a << endl;
509
510     iset64 b(y, ly);
511     cout << "b = " << b << endl;
512
513     iset64 aplusb = a + b;
514     cout << "aplusb = " << aplusb << endl;
515
516     iset64 aplusbbar = ~(aplusb);
517     cout << "aplusbbar = " << aplusbbar << endl;
518
519     iset64 abar = ~(a);
520     cout << "abar = " << abar << endl;
521
522     iset64 bbar = ~(b);
523     cout << "bbar = " << bbar << endl;
524
525     iset64 abarplusbbar = abar + bbar;
526     cout << "abarplusbbar = " << abarplusbbar << endl;
527
528     iset64 abardotbbar = abar * bbar;
529     cout << "abardotbbar = " << abardotbbar << endl;
530
531     iset64 adotb = a * b;
532     cout << "adotb  = " << adotb << endl;
```

```
533
534     iset64 adotbbar = ~(adotb);
535     cout << "adotbbar  = " << adotbbar << endl;
536
537     assert(aplusbbar == abardotbbar);
538     cout << "Demorgan law (a+b)' = a'. b' is proved\n";
539     assert(adotbbar == abarplusbbar);
540     cout << "Demorgan law (a.b)' = a' + b' is proved\n";
541   }
542 }
543
544 /*-----------------------------------------------------------------
545 (a+b)' = a'. b'
546 (a.b)' = a' + b'
547 -----------------------------------------------------------------*/
548 void test_demorgan_laws() {
549   {
550     int x[] = { 4, 5, 6 };
551     int y[] = { 5, 6, 8 };
552     test_demorgan_laws(x, (sizeof(x) / sizeof(int)), y, (sizeof(y) / sizeof
        (int)));
553   }
554   {
555     int x[] = { 1,2,4,5 };
556     int y[] = { 2,3,5,6 };
557     test_demorgan_laws(x, (sizeof(x) / sizeof(int)), y, (sizeof(y) / sizeof
        (int)));
558   }
559
560 }
561
562 /*-----------------------------------------------------------------
563 Munadir test bed
564 -----------------------------------------------------------------*/
565
566 void munadir_test() {
567     iset64 a;
568     int x[] = { 1,2,4,5 };
569     iset64 b(x, 4);
570     cout << b;
571 }
572
573
574 /*-----------------------------------------------------------------
575 test bed
576 -----------------------------------------------------------------*/
577 void testbed() {
578   test_basic();
579   test_union();
```

```
580   test_difference();
581   test_intersection();
582   test_equal_not_equal();
583   test_pre_post_inr_dec();
584   test_complement();
585   test_conversion_operator();
586   test_not_operator();
587   test_demorgan_laws();
588   munadir_test();
589 }
590
591 /*--------------------------------------------------------------
592 main
593 --------------------------------------------------------------*/
594 int main() {
595 #ifdef _WIN32
596   _CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF | _CRTDBG_LEAK_CHECK_DF);
597 #endif
598   iset64::set_display(false) ;
599   testbed();
600   cout << "Must attach output of the program to get a grade\n" ;
601   cout << "Must attach a doc that explains the data structure that was used to ↵
        solve to get a grade\n" ;
602   cin.get();
603   return 0;
604 }
605
606 //EOF
607
608 #if 0
609 /*
610 a = {}
611
612 set a after adding 5 = {5}; size of set is: 1
613
614 set a after adding 5 = {5}; size of set is: 1
615
616 set a after adding 0 and 63 = {5, 63, 0}; size of set is: 3
617
618 set b = {1, 3, 6}; size of set is: 3
619
620 set b after removing 3 = {1, 6}; size of set is: 2
621
622 set b after removing 3 = {1, 6}; size of set is: 2
623
624 set b after removing 10 = {1, 6}; size of set is: 2
625
626 set b after removing 6 = {1}; size of set is: 1
627
```

```
628 set b after removing 1 = {}
629
630 set b after adding {10,2} = {10, 2}; size of set is: 2
631
632 a[0]={0, 1, 62}; size of set is: 3
633
634 a[1]={0, 1, 62}; size of set is: 3
635
636 a[2]={0, 1, 62}; size of set is: 3
637
638 a[3]={0, 1, 62}; size of set is: 3
639
640 a[4]={0, 1, 62}; size of set is: 3
641
642 TESTING: iset64 operator+(const iset64& a, const iset64& b)
643 Set a {1, 2}; size of set is: 2
644
645 Set b {1, 2, 3}; size of set is: 3
646
647 a + b = {1, 2, 3}; size of set is: 3
648
649 TESTING:iset64 operator+(const iset64& a, const int b)
650 {1, 2}; size of set is: 2
651
652 {1,2} + 1 = {1, 2}; size of set is: 2
653
654 {1, 2}; size of set is: 2
655
656 {1,2} + 3 = {1, 2, 3}; size of set is: 3
657
658 TESTING:iset64 operator+(const int b, const iset64& a)
659 Set a {1, 2}; size of set is: 2
660
661 1 + {1,2} = {1, 2}; size of set is: 2
662
663 Set a {1, 2}; size of set is: 2
664
665 3 + {1,2}  = {1, 2, 3}; size of set is: 3
666
667 TESTING:iset64& iset64::operator+=(const iset64& a)
668 Set b {1, 2}; size of set is: 2
669
670 Set a {1, 3}; size of set is: 2
671
672 {1,2} + {1,3}  = {1, 2, 3}; size of set is: 3
673
674 iset64& iset64::operator+=(const int b)
675 Set a {1, 2}; size of set is: 2
676
```

```
677 {1,2} + 3  = {1, 2, 3}; size of set is: 3
678
679 Set a {1, 2}; size of set is: 2
680
681 Set b {3, 4}; size of set is: 2
682
683 Set c {7, 8}; size of set is: 2
684
685 Set d {1, 2, 3, 4, 7, 8, 5}; size of set is: 7
686
687 TESTING: iset64 operator-(const iset64& a, const iset64& b)
688 Set a {1, 2}; size of set is: 2
689
690 Set b {1, 2}; size of set is: 2
691
692 a - b = {}
693
694 TESTING: iset64 operator-(const iset64& a, const iset64& b)
695 Set a {1, 5}; size of set is: 2
696
697 Set b {1, 2, 3}; size of set is: 3
698
699 a - b = {5}; size of set is: 1
700
701 TESTING: iset64 operator-(const iset64& a, const int b)
702 Set a {1, 2}; size of set is: 2
703
704 a - 3 = {1, 2}; size of set is: 2
705
706 TESTING: iset64 operator-(const int b, const iset64& a)
707 Set a {1, 2}; size of set is: 2
708
709 3 - a = {1, 2}; size of set is: 2
710
711 TESTING: iset64& iset64::operator-=(const iset64& a)
712 Set a {1, 3}; size of set is: 2
713
714 Set b {1, 2}; size of set is: 2
715
716 b -= a = {2}; size of set is: 1
717
718 TESTING: iset64& iset64::operator-=(const int b)
719 Set a {1, 2}; size of set is: 2
720
721 a -= 3 = {1, 2}; size of set is: 2
722
723 Set a {1, 2}; size of set is: 2
724
725 Set b {2, 4}; size of set is: 2
```

```
726
727 Set c {2, 8}; size of set is: 2
728
729 Set d {1, 5}; size of set is: 2
730
731 TESTING: iset64 operator*(const iset64& a, const iset64& b)
732 Set a {1, 2}; size of set is: 2
733
734 Set b {1, 2, 3}; size of set is: 3
735
736 a * b = {1, 2}; size of set is: 2
737
738 TESTING:iset64 operator*(const iset64& a, const int b)
739 Set a {1, 2}; size of set is: 2
740
741 {1,2} * 1 = {1}; size of set is: 1
742
743 Set a {1, 2}; size of set is: 2
744
745 {1,2} * 3 = {}
746
747 TESTING:iset64 operator*(const int b, const iset64& a)
748 Set a {1, 2}; size of set is: 2
749
750 1 * {1,2} = {1}; size of set is: 1
751
752 Set a {1, 2}; size of set is: 2
753
754 3 * {1,2}  = {}
755
756 TESTING:iset64& iset64::operator*=(const iset64& a)
757 Set b {1, 2}; size of set is: 2
758
759 Set a {1, 3}; size of set is: 2
760
761 {1,2} * {1,3}  = {1}; size of set is: 1
762
763 iset64& iset64::operator*=(const int b)
764 Set a {1, 2}; size of set is: 2
765
766 {1,2} * 3  = {}
767
768 Set a {1, 2}; size of set is: 2
769
770 Set b {2, 4}; size of set is: 2
771
772 Set c {2, 8}; size of set is: 2
773
774 Set d {2, 5}; size of set is: 2
```

```
775
776 TESTING: bool operator==(const iset64& a, const iset64& b)
777 Set a {1, 2}; size of set is: 2
778
779 Set b {1, 2}; size of set is: 2
780
781 a == b true
782 {1, 2}; size of set is: 2
783 {2}; size of set is: 1
784 a == b true
785 TESTING: bool operator!=(const iset64& a, const iset64& b)
786 Set a {1, 2}; size of set is: 2
787
788 Set b {1, 2}; size of set is: 2
789
790 a != b false
791 Set a {1, 2}; size of set is: 2
792
793 Set b {2}; size of set is: 1
794
795 a != b false
796 a = {1, 2, 63}; size of set is: 3
797
798 ++a = {2, 3, 0}; size of set is: 3
799
800 a = {1, 2, 63}; size of set is: 3
801
802 acopy = {1, 2, 63}; size of set is: 3
803
804 a++ = {2, 3, 0}; size of set is: 3
805
806 rhs = {1, 2, 63}; size of set is: 3
807
808 a = {0, 2, 63}; size of set is: 3
809
810 --a = {63, 1, 62}; size of set is: 3
811
812 a = {0, 2, 63}; size of set is: 3
813
814 acopy = {0, 2, 63}; size of set is: 3
815
816 a-- = {63, 1, 62}; size of set is: 3
817
818 rhs = {0, 2, 63}; size of set is: 3
819
820 a = {0, 2, 63}; size of set is: 3
821
822 ~a = {61, 1, 62, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,  ↵
      20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,  ↵
```

```
      39, 40
823   , 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
      60}; size of set is: 61
824
825   ans = {1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
      22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
      41, 4
826   2, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61,
      62}; size of set is: 61
827
828   ~ans = {0, 63, 2}; size of set is: 3
829
830   a = {0, 2, 63}; size of set is: 3
831
832   a exists
833   b = {}
834
835   b does not exists
836   a = {0, 2, 63}; size of set is: 3
837
838   a exists
839   b = {}
840
841   b does not exists
842   a = {4, 5, 6}; size of set is: 3
843
844   b = {5, 6, 8}; size of set is: 3
845
846   aplusb = {4, 5, 6, 8}; size of set is: 4
847
848   aplusbbar = {0, 1, 2, 3, 63, 62, 61, 7, 60, 9, 10, 11, 12, 13, 14, 15, 16, 17,
      18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
      37, 3
849   8, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57,
      58, 59}; size of set is: 60
850
851   abar = {0, 1, 2, 3, 63, 62, 61, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
      19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
      38, 39,
852   40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
       60}; size of set is: 61
853
854   bbar = {0, 1, 2, 3, 4, 63, 62, 7, 61, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
      19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
      38, 39,
855   40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
       60}; size of set is: 61
856
857   abarplusbbar = {0, 1, 2, 3, 63, 62, 61, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
```

```
      17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
      36, 37,
858   38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57,
       58, 59, 60, 4}; size of set is: 62
859
860   abardotbbar = {0, 1, 2, 3, 63, 62, 7, 61, 9, 10, 11, 12, 13, 14, 15, 16, 17,
      18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
      37, 38,
861   39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
       59, 60}; size of set is: 60
862
863   adotb  = {5, 6}; size of set is: 2
864
865   adotbbar  = {0, 1, 2, 3, 4, 63, 62, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
      18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
      37, 38,
866   39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
       59, 60, 61}; size of set is: 62
867
868   Demorgan law (a+b)' = a'. b' is proved
869   Demorgan law (a.b)' = a' + b' is proved
870   a = {1, 2, 4, 5}; size of set is: 4
871
872   b = {2, 3, 5, 6}; size of set is: 4
873
874   aplusb = {1, 2, 4, 5, 3, 6}; size of set is: 6
875
876   aplusbbar = {0, 63, 62, 59, 61, 60, 58, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
      17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
      36, 37,
877   38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56,
      57}; size of set is: 58
878
879   abar = {0, 63, 62, 3, 61, 60, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
      19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
      38, 39
880   , 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
      59}; size of set is: 60
881
882   bbar = {0, 1, 63, 62, 4, 61, 60, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
      19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
      38, 39
883   , 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
      59}; size of set is: 60
884
885   abarplusbbar = {0, 63, 62, 3, 61, 60, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
      17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
      36, 37
886   , 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56,
```

```
      57, 58, 59, 1, 4}; size of set is: 62
887
888 abardotbbar = {0, 63, 62, 61, 60, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
      19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
      38, 3
889 9, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
      59}; size of set is: 58
890
891 adotb  = {2, 5}; size of set is: 2
892
893 adotbbar  = {0, 1, 63, 3, 4, 62, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
      18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
      37, 38,
894 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
       59, 60, 61}; size of set is: 62
895
896 Demorgan law (a+b)' = a'. b' is proved
897 Demorgan law (a.b)' = a' + b' is proved
898 {1, 2, 4, 5}; size of set is: 4
899 Must attach output of the program to get a grade
900 Must attach a doc that explains the data structure that was used to solve to
      get a grade
901
902 */
903 #endif
904
905
```