Source File: `~/2336/04/lab04.(C|CPP|cpp|c++|cc|cxx|cp)`
Input: Under control of `main` function
Output: Under control of `main` function
Value: 3

In this assignment create an `IntegerSet` class that will provide an efficient representation for a set of integers and a collection of methods (member functions) to provide various operations on `IntegerSet`s.

Given some number N, represent the set of integers $0, 1, 2, \ldots, (N-1)$ as an array, where each element of the set is represented by a single bit. Use an array of `unsigned int`s to represent the set. For example, if $N = 128$ and since the `sizeof(unsigned int)` is four bytes, the array would need to contain four elements. Set elements in the range from 0 through 31 would be represented within the first element of the array, set elements 32 through 63 would be represented within the second element of the array, and so on. Within the first element of the array, let the rightmost (least significant) bit represent 0 and the leftmost (most significant) bit represent 31. Within the second element of the array, let the rightmost (least significant) bit represent 32 and the leftmost (most significant) bit represent 63. Etc. This is consistent with the presentation in *The Joy of C*, section 6.4, pp. 152–164.

Additional notes:

- The `print` function prints the elements of a set in traditional set notation. A set containing the elements 3, 7, and 14 should print as `{3,7,14}`. Note the commas and spacing. After the set has been printed, it should be followed by exactly one linefeed. An empty set should be displayed as ∅. Use `static_cast<char>(216)` to approximate this symbol. The empty set symbol should not be enclosed in braces. It should be followed by exactly one linefeed.
- The cardinality of a set is the number of elements in the set.
- The complement of a set A with respect to a given universal set I is the set of elements in I that are not in A; a universal set I (or universe) is the term given to an overall set which includes all the values of concern in a given study.

A header file is shown in Figure 1, a sample `main` function for testing your implementation is shown in Figure 2, and a sample execution sequence is shown in Figure 3. To use the `Makefile` as distributed in class, add a target of `lab04` to `targets2srcfiles`.

```
1  #ifndef LAB04_H
2  #define LAB04_H
3
4  #include <iostream>
5  #include <bits.h>
6
7  using namespace std;
8
9  const uint N = 40;
10
11 class IntegerSet
12 {
13  public:
14   IntegerSet();                            // initializes the set to the empty
15                                            //   set
16   IntegerSet(const IntegerSet& otherSet); // copy constructor
17   ~IntegerSet();                           // destructor
```

Figure 1. `/usr/local/2336/include/lab04.h` (Part 1 of 2)

```
18    bool isMember(uint e) const;           // returns true if e is a member of
19                                           //   the set and false otherwise
20    uint cardinality() const;              // cardinality of a set
21    void insertElement(uint e);            // if e is valid and not a member of
22                                           //   the set, insert e into set
23    void deleteElement(uint e);            // if e is valid and a member of
24                                           //   the set, delete e from set
25    IntegerSet complement() const;         // complement of a Set
26
27    ostream& print(ostream& os) const;
28   private:
29    uint *bitVector;                       // Pointer to dynamically
30                                           //   allocated memory
31    bool isValid(uint e) const;            // 0 <= e < N
32    uint word(uint n) const;               // Determine index within
33                                           //   bitVector where n is located
34    uint bit(uint n) const;                // Determine position within
35                                           //   bitVector[word(n)]
36                                           //   for element n
37    void allocateStorage();                // Calculate # of elements
38                                           //   in bitVector to represent
39                                           //   elements 0..(N-1) & then
40                                           //   allocate storage
41  };
42
43  #endif
```

Figure 1. `/usr/local/2336/include/lab04.h` (Part 2 of 2)

```
1  #include <lab04.h>
2  #include <iomanip>
3  #include <climits>
4
5  using namespace std;
6
7  int main()
8  {
9    uint e, j, n;
10   IntegerSet s;
11
12   while (cin >> n)
13   {
14     for (j = 0; j < n; ++j)
15     {
16       cin >> e;
17       s.insertElement(e);
18     }
19     cout << "s = ";
```

Figure 2. `/usr/local/2336/src/lab04main.C` (Part 1 of 3)

```
20      s.print(cout);
21      cout << "s.cardinality() = " << s.cardinality() << endl << endl;
22
23      // Use the copy constructor to make a copy of s
24      IntegerSet sCopy(s);
25      cout << "sCopy = ";
26      sCopy.print(cout);
27      cout << "sCopy.cardinality() = " << sCopy.cardinality() << endl << endl;
28
29      // Use the copy constructor to initialize t with the complement of s
30      IntegerSet t(s.complement());
31      cout << "t = ";
32      t.print(cout);
33      cout << "t.cardinality() = " << t.cardinality() << endl << endl;
34
35      // clear set s
36      for (e = 0; e < N; ++e)
37      {
38        if (s.isMember(e))
39          s.deleteElement(e);
40      }
41      cout << "s.cardinality() = " << s.cardinality() << endl << endl;
42    }
43
44    return 0;
45  }
46
47  uint IntegerSet::word(uint n) const
48  {
49    return n / (CHAR_BIT * sizeof(uint));
50  }
51
52  uint IntegerSet::bit(uint n) const
53  {
54    return n % (CHAR_BIT * sizeof(uint));
55  }
56
57  void IntegerSet::allocateStorage()
58  {
59    uint numElements = word(N - 1) + 1;
60
61    // dynamically allocate memory for the bitVector
62    try
63    {
64      bitVector = new uint[numElements];
65    }
```

Figure 2. `/usr/local/2336/src/lab04main.C` (Part 2 of 3)

```
66    catch (bad_alloc memoryAllocationException)
67    {
68      cerr << "Unable to allocate memory. Exiting." << endl;
69      cerr << memoryAllocationException.what() << endl;
70      exit(EXIT_FAILURE);
71    }
72  }
```

Figure 2. `/usr/local/2336/src/lab04main.C` (Part 3 of 3)

```
1  newuser@csunix ~> cd 2336
2  newuser@csunix ~/2336> ./getlab.ksh 04
3    * Checking to see if a folder exists for Lab 04. . .No
4    * Creating a folder for Lab 04
5    * Checking to see if Lab 04 has sample input and output files. . .Yes
6    * Copying input and output files for Lab 04
7      from folder /usr/local/2336/data/04 to folder ./04
8    * Checking to see if /usr/local/2336/src/lab04main.C exists. . .Yes
9    * Copying file /usr/local/2336/src/lab04main.C to folder ./04
10   * Checking to see if /usr/local/2336/include/lab04.h exists. . .Yes
11   * Copying file /usr/local/2336/include/lab04.h to folder ./04
12   * Copying file /usr/local/2336/src/Makefile to folder ./04
13   * Adding a target of lab04 to targets2srcfiles
14   * Touching file ./04/lab04.cpp
15   * Edit file ./04/lab04.cpp in Notepad++
16 newuser@csunix ~/2336> cd 04
17 newuser@csunix ~/2336/04> ls
18 01.dat       01.out       Makefile     lab04.cpp    lab04.h      lab04main.C
19 newuser@csunix ~/2336/04> make lab04
20 g++ -g -Wall -std=c++11 -c lab04main.C -I/usr/local/2336/include -I.
21 g++ -g -Wall -std=c++11 -c lab04.cpp -I/usr/local/2336/include -I.
22 g++ -o lab04 lab04main.o lab04.o -L/usr/local/2336/lib -lm -lbits
23 newuser@csunix ~/2336/04> cat 01.dat
24 6
25 1 2 4 8 16 32
26 10
27 3 6 9 12 15 3 6 9 12 15
28 13
29 4 8 12 16 20 24 28 32 36 40 44 48 52
30 48
31  0  1  2  3  4  5  6  7  8  9
32 10 11 12 13 14 15 16 17 18 19
33 20 21 22 23 24 25 26 27 28 29
34 30 31 32 33 34 35 36 37 38 39
35 40 41 42 43 44 45 46 47
36 0
```

Figure 3. Commands to Compile, Link, & Run Lab 04 (Part 1 of 3)

```
newuser@csunix ~/2336/04> cat 01.dat | ./lab04
s = {1,2,4,8,16,32}
s.cardinality() = 6

sCopy = {1,2,4,8,16,32}
sCopy.cardinality() = 6

t = {0,3,5,6,7,9,10,11,12,13,14,15,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,33,34,35,36,37,38,39}
t.cardinality() = 34

s.cardinality() = 0

s = {3,6,9,12,15}
s.cardinality() = 5

sCopy = {3,6,9,12,15}
sCopy.cardinality() = 5

t = {0,1,2,4,5,7,8,10,11,13,14,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39}
t.cardinality() = 35

s.cardinality() = 0

s = {4,8,12,16,20,24,28,32,36}
s.cardinality() = 9

sCopy = {4,8,12,16,20,24,28,32,36}
sCopy.cardinality() = 9

t = {0,1,2,3,5,6,7,9,10,11,13,14,15,17,18,19,21,22,23,25,26,27,29,30,31,33,34,35,37,38,39}
t.cardinality() = 31
```

Figure 3. Commands to Compile, Link, & Run Lab 04 (Part 2 of 3)

```
68
69  s.cardinality() = 0
70
71  s = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39}
72  s.cardinality() = 40
73
74  sCopy = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39}
75  sCopy.cardinality() = 40
76
77  t = ∅
78  t.cardinality() = 0
79
80  s.cardinality() = 0
81
82  s = ∅
83  s.cardinality() = 0
84
85  sCopy = ∅
86  sCopy.cardinality() = 0
87
88  t = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39}
89  t.cardinality() = 40
90
91  s.cardinality() = 0
92
93  newuser@csunix ~/2336/04> cat 01.dat | ./lab04 > my.out
94  newuser@csunix ~/2336/04> diff 01.out my.out
95  newuser@csunix ~/2336/04>
```

Figure 3. Commands to Compile, Link, & Run Lab 04 (Part 3 of 3)