**CSCI 220 -- Project 1 (Integer Sets)**

**Due Tuesday, 09/17/2013**

This project is designed to refresh your C++/Java programming skill.  You may want to review some C++/Java topics (especially arrays and OOP) and research some information about sets before working on this project.  You can implement your project in either C++ or Java.

A set is a collection of distinct objects or values.  One simple way to implement a set of integers is to represent it internally as an array of ones and zeros (or true and false) and we will use this approach for this project.  Array element a[ i ] is 1 if element i is in the set.  Array element a[ j ] is 0 if element j is not in the set.  This approach would work well for a small range of non-negative integer values such as 0 to 127.  However, it is very inefficient for a large range of integer values and an adjustment is needed for negative values or other types of values.

The following common set operations shall be available as requested by the users.  Provide a ***union2*** member function that creates a third set which is the set-theoretic union of the two existing sets without changing the two existing sets (i.e., an element is in third set if that element is in either one or both existing sets; the member function has second set as a parameter and returns the third set).  Provide an ***intersect2*** member function that creates a third set which is the set-theoretic intersection of the two existing sets without changing the two existing sets (i.e., an element is in the third set if that element is in both existing sets).  Provide an ***insertElement*** member function that inserts a new element k into an existing set.  Provide a ***removeElement*** member function that removes element m from an existing set.  Provide a ***print*** member function that prints a set as a list of elements in ascending order separated by commas (i.e., {3, 6, 10}).  Print only those elements that are present in the set and print {} for an empty set.  Provide a boolean ***isEqual*** member function that determines if two sets are equal.  Provide a boolean ***isSubset*** member function that determines if first set is a subset of the second set (all elements in first set are also in second set).  For some operations, make sure that it is legal (for example, inserting or removing a valid element); otherwise, display an error message (can use exception handling).

You need to provide an **IntegerSet** class with basic operations described above.  Each object of class **IntegerSet** can hold a set of integer values from 0 to 127.  Provide three overload constructors:  a default constructor (empty set), a constructor with an integer parameter (set with one integer element), and a constructor with an integer array parameter (set with a list of elements).  Make sure to set up your constructor(s) and public member functions in such a way that your IntegerSet class is very flexible and easier to use.  Now write a test driver to test your **IntegerSet** class (separate from your implementation file).  Instantiate several **IntegerSet** objects and test that all your member functions work properly.   Your test driver would contain all necessary test cases to test the IntegerSet class.

**Extra credit:** You can earn up to 5 additional points if you can outline a better and efficient way to implement a set with much large range of integer values or other types of values.  Include runtime in Big-Oh notation for each basic set operation described earlier.

Please provide documentation and applying good coding style because it is part of the grade.  You must come up with a sufficient number of test cases since the test cases are also part of the grade.  Please submit the following items **in a folder** if flash drive or CD is included (can also submit source code ahead of time via Moodlerooms (MR) and a folder is not needed):

1. Title page with name, class, project number, and relevant information about your program (compiler and system used, file names).
2. Notes about your program (status of your program and lessons learned at the minimum).
3. Runtime in Big-Oh notation for each set operation.
4. Printouts of any input/output.
5. A printout of the source code.
6. A copy of your source code on a flash drive or CD or MR (.h /.cpp or .java file).

Your program will be graded as follow:

* Correctness/Efficiency: 35 points
* Test Cases:  7 points
* Documentation/Coding Style:  8 points

© by T. Vo   
Last updated on 08/21/2013 18:13:16

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | A simple demostraion of Sets and sets operations by using a Class | http://www.planet-source-code.com/images/IconExperience2011/v_collection/v_collection_png/16x16/plain/mail_forward.png[Email](http://www.planet-source-code.com/vb/scripts/EmailToFriend.asp?lngWId=3&txtCodeId=8835&txtCodeName=A%2520simple%2520demostraion%2520of%2520Sets%2520and%2520sets%2520operations%2520by%2520using%2520a%2520Class&txtCodeDescription=I%2520have%2520demonstrated%2520a%2520nice%2520use%2520of%2520set%2520data%2520structure%2520represented%2520by%2520a%2520Class%2E%0AHere%2520is%2520the%2520assignment%2E%0AYou%2520have%2520to%2520Create%2520a%2520class%2520IntegerSet%2520for%2520which%2520each%2520object%2520can%2520hold%2520integers%2520in%2520the%2520range%25200%2520through%2520100%2E%2520A%2520set%2520is%2520represented%2520internally%2520as%2520an%2520array%2520of%2520ones%2520and%2520zeros%2E%2520Array%2520element%2520a%5Bi%5D%2520is%25201%2520if%2520integer%2520i%2520is%2520in%2520the%2520set%2E%2520Array%2520element%2520a%5Bj%5D%2520is%25200%2520if%2520integer%2520j%2520is%2520not%2520in%2520the%2520set%2E%2520The%2520default%2520constructor%2520initializes%2520a%2520set%2520to%2520the%2520so%2Dcalled%2520%93empty%2520set%2C%94%2520i%2Ee%2E%2C%2520a%2520set%2520whose%2520array%2520representation%2520contains%2520all%2520zeros%2E%2520%0AProvide%2520member%2520functions%2520for%2520the%2520common%2520set%2520operations%2E%2520For%2520example%2C%2520provide%2520a%2520unionOfSets%2520member%2520function%2520that%2520creates%2520a%2520third%2520set%2520that%2520is%2520the%2520set%2Dtheoretic%2520union%2520of%2520two%2520existing%2520sets%2520%28i%2Ee%2E%2C%2520an%2520element%2520of%2520the%2520third%2520set%92s%2520array%2520is%2520set%2520to%25201%2520if%2520that%2520element%2520is%25201%2520in%2520either%2520or%2520both%2520of%2520the%2520existing%2520sets%2C%2520and%2520an%2520element%2520of%2520the%2520third%2520set%92s%2520array%2520is%2520set%2520to%25200%2520if%2520that%2520element%2520is%25200%2520in%2520each%2520of%2520the%2520existing%2520sets%29%2E%0AProvide%2520an%2520intersectionOfSets%2520member%2520function%2520which%2520creates%2520a%2520third%2520set%2520which%2520is%2520the%2520set%2Dtheoretic%2520intersection%2520of%2520two%2520existing) |  |  |  | | --- | --- | | **Submitted on:** | 1/22/2005 3:52:26 PM | | **By:** | [Aba](http://www.planet-source-code.com/vb/authors/ShowBio.asp?lngAuthorId=2295513081&lngWId=3) | | **Level:** | Beginner | | **User Rating:** | Unrated | | **Compatibility:** | C++ (general), Microsoft Visual C++, Borland C++, UNIX C++ | | **Views:** | 6772 | | author picture | |
|  | | |
| I have demonstrated a nice use of set data structure represented by a Class. Here is the assignment. You have to Create a class IntegerSet for which each object can hold integers in the range 0 through 100. A set is represented internally as an array of ones and zeros. Array element a[i] is 1 if integer i is in the set. Array element a[j] is 0 if integer j is not in the set. The default constructor initializes a set to the so-called �empty set,� i.e., a set whose array representation contains all zeros. Provide member functions for the common set operations. For example, provide a unionOfSets member function that creates a third set that is the set-theoretic union of two existing sets (i.e., an element of the third set�s array is set to 1 if that element is 1 in either or both of the existing sets, and an element of the third set�s array is set to 0 if that element is 0 in each of the existing sets). Provide an intersectionOfSets member function which creates a third set which is the set-theoretic intersection of two existing sets (i.e., an element of the third set�s array is set to 0 if that element is 0 in either or both of the existing sets, and an element of the third set�s array is set to 1 if that element is 1 in each of the existing sets). Provide an insertElement member function that inserts a new integer k into a set (by setting a[k] to 1). Provide a deleteElement member function that deletes integer m (by setting a[m] to 0). Provide a printSet member function that prints a set as a list of numbers separated by spaces. Print only those elements that are present in the set (i.e., their position in the array has a value of 1). Print --- for an empty set. Provide an isEqualTo member function that determines whether two sets are equal. Provide an additional constructor that receives an array of integers and the size of that array and uses the array to initialize a set object. Now write a driver program to test your IntegerSet class. Instantiate several IntegerSet objects. Test that all your member functions work properly. | | |
|  | | |
|  |  |  |
| code: *Can't Copy and Paste this? Click here for a*[*copy-and-paste friendly*](http://www.planet-source-code.com/vb/scripts/ShowCodeAsText.asp?txtCodeId=8835&lngWId=3)*version of this code!*   |  |  |  | | --- | --- | --- | | Terms of Agreement:    By using this code, you agree to the following terms...   1. You may use this code in your own programs (and may compile it into a program and distribute it in compiled format for languages that allow it) freely and with no charge. 2. You MAY NOT redistribute this code (for example to a web site) without written permission from the original author. Failure to do so is a violation of copyright laws. 3. You may link to this code from another website, but ONLY if it is not wrapped in a frame. 4. You will abide by any additional copyright restrictions which the author may have placed in the code or code's description. | | | | //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // Name: A simple demostraion of Sets and sets operations by using a Class  // Description:I have demonstrated a nice use of set data structure represented by a Class.  Here is the assignment.  You have to Create a class IntegerSet for which each object can hold integers in the range 0 through 100. A set is represented internally as an array of ones and zeros. Array element a[i] is 1 if integer i is in the set. Array element a[j] is 0 if integer j is not in the set. The default constructor initializes a set to the so-called �empty set,� i.e., a set whose array representation contains all zeros.  Provide member functions for the common set operations. For example, provide a unionOfSets member function that creates a third set that is the set-theoretic union of two existing sets (i.e., an element of the third set�s array is set to 1 if that element is 1 in either or both of the existing sets, and an element of the third set�s array is set to 0 if that element is 0 in each of the existing sets).  Provide an intersectionOfSets member function which creates a third set which is the set-theoretic intersection of two existing sets (i.e., an element of the third set�s array is set to 0 if that element is 0 in either or both of the existing sets, and an element of the third set�s array is set to 1 if that element is 1 in each of the existing sets).  Provide an insertElement member function that inserts a new integer k into a set (by setting a[k] to 1). Provide a deleteElement member function that deletes integer m (by setting a[m] to 0).  Provide a printSet member function that prints a set as a list of numbers separated by spaces. Print only those elements that are present in the set (i.e., their position in the array has a value of 1). Print --- for an empty set.  Provide an isEqualTo member function that determines whether two sets are equal.  Provide an additional constructor that receives an array of integers and the size of that array and uses the array to initialize a set object.  Now write a driver program to test your IntegerSet class. Instantiate several IntegerSet objects. Test that all your member functions work properly.  // By: Aba  //  //This code is copyrighted and has// limited warranties.Please see [http://www.Planet-Source-Code.com/vb/scripts/ShowCode.asp?txtCodeId=8835&lngWId=3//for](http://www.planet-source-code.com/vb/scripts/ShowCode.asp?txtCodeId=8835&lngWId=3//for) details.//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  #include<iostream.h>  const SIZE=101;//0 to 100 set containing 101 elements  class IntegerSet  {  int a[SIZE];  public:  IntegerSet();  IntegerSet(int \*, int);  IntegerSet unionOfSets(IntegerSet s2);  IntegerSet intersectionOfSets(IntegerSet s2);  int isEqualTo(IntegerSet s2);  void insertElement(int n);  void printSet();  };  IntegerSet::IntegerSet()//default constructor creates empty set  {  int i;  for(i=0;i<SIZE;i++)  a[i]=0;  }  IntegerSet::IntegerSet(int \*ptr, int n)//initialize object using another array of integers  {  int i;  for(i=0;i<SIZE;i++)//Initialize the set first  a[i]=0;  for(i=0;i<n;i++)  a[(ptr[i])]=1;  }  IntegerSet IntegerSet::unionOfSets(IntegerSet s2)  {  IntegerSet s3;  int i;  for(i=0;i<SIZE;i++)  if(a[i]==1 || s2.a[i]==1)  s3.a[i]=1;  return s3;  }  IntegerSet IntegerSet::intersectionOfSets(IntegerSet s2)  {  IntegerSet s3;  int i;  for(i=0;i<SIZE;i++)  if(a[i]==1 && s2.a[i]==1)  s3.a[i]=1;  return s3;  }  int IntegerSet::isEqualTo(IntegerSet s2)  {  int i;  int check=1;//equal  for(i=0;i<SIZE;i++)  if(a[i]!=s2.a[i])  {  check=0;//not equal  break;  }  return check;  }  void IntegerSet::insertElement(int n)  {  if (n<SIZE && n>=0) //check validity  a[n]=1;  }  void IntegerSet::printSet()  {  int i, count=0;  for(i=0;i<SIZE;i++)  if(a[i]==1)  {  count++;  cout<<i<<" ";  }  if(count==0)  cout<<"---";  }  void main()  {  int arr[4]={3,76,34,56};  IntegerSet s1, s2(arr, 4), s3, s4;  s4.insertElement(34);  s4.insertElement(56);  cout<<"\nDefault set=";  s1.printSet();  cout<<"\nUsing array as parameter, set s2= ";  s2.printSet();  cout<<endl;  cout<<"Inserting 50, 100,1,0,34, 56 to s1, set s1=";  s1.insertElement(50);  s1.insertElement(100);  s1.insertElement(1);  s1.insertElement(0);  s1.insertElement(34);  s1.insertElement(56);  s1.printSet();  cout<<endl;  cout<<"Union of s1 and s2, set s3=";  s3=s1.unionOfSets(s2);  s3.printSet();  cout<<endl;  cout<<"Intersection of s1 and s2, set s3=";  s3=s1.intersectionOfSets(s2);  s3.printSet();  cout<<endl;  cout<<"set s4=";  s4.printSet();  cout<<endl;  cout<<"s2==s3=";  if(s2.isEqualTo(s3)==1)  cout<<"True";  else  cout<<"False";  cout<<"\ns3==s4=";  if(s3.isEqualTo(s4)==1)  cout<<"True";  else  cout<<"False";  } |  |  | | | |
|  |  |  |
|  | | |