Dylan T Carlson

1161653

Lab102 Assignment

09/05/2018

**Bag:**

/\*\*

\* This interface, Bag, contains all the method signatures that are

\* implemented into Scores.

\*

\* @author dylca

\*/

public interface Bag {

/\*\*

\* getCurrentSize returns the value of count.

\* @return

\*/

int getCurrentSize();

/\*\*

\* isEmpty returns true if the list is empty, and false if it isn't.

\* @return

\*/

boolean isEmpty();

/\*\*

\* clear sets all the values of list to zero.

\*/

void clear();

/\*\*

\* add adds a number to the list, unless

\* the next number would be outside the array.

\* In this case, it recreates the array double its size.

\* \* @param num

\*/

void add(int num);

/\*\*

\* getFrequencyOf is passed an integer and

\* returns the number of times that number is present in the list.

\* @param num

\* @return

\*/

int getFrequencyOf(int num);

/\*\*

\* contains is passed and integer, and if that value is found within

\* the list, it will return true, if not, false.

\* @param num

\* @return

\*/

boolean contains(int num);

/\*\*

\* remove(int num) is passed an integer, and removes it from the list.

\* @param num

\*/

void remove(int num);

/\*\*

\* remove() removes a random number from the list.

\*/

void remove();

/\*\*

\* toString prints all the contents of the Scores class.

\* Which includes: the class name, the count, and the list of numbers.

\* @return

\*/

String toString();

/\*\*

\* equals is passed an object and tests if it is equal

\* to the list from the scores class.

\* @param o

\* @return

\*/

boolean equals(Object o);

}

**Scores:**

/\*\*

\* This class, Scores, implements the Bag interface.

\* It implements all the methods that the Bag interface has.

\* It keeps track of the numbers in the Bag with count, and stores them into an int array, list.

\* The Random class is imported to generate random ints.

\*

\* @author dylca

\*/

import java.util.Random;

public class Scores implements Bag {

private int[] list, temp;

private int count = 0;

//Default Constructor

public Scores(){

list = new int[50];

}

//Overload Constructor

public Scores(int arrayLength){

list = new int[arrayLength];

}

/\*\*

\* getCurrentSize returns the value of count.

\* @return

\*/

public int getCurrentSize(){

return count;

}

/\*\*

\* isEmpty returns true if the list is empty, and false if it isn't.

\* @return

\*/

public boolean isEmpty(){

if (count == 0)

return true;

else

return false;

}

/\*\*

\* clear sets all the values of list to zero.

\*/

public void clear(){

for (; count>=0; count-- ){

list[count] = 0;

}

count = 0;

}

/\*\*

\* add adds a number to the list, unless

\* the next number would be outside the array.

\* In this case, it recreates the array double its size.

\* \* @param num

\*/

public void add(int num){

if (count == list.length){

temp = new int[ 2\* list.length ];

//Copying list into temp

for(int k = 0; k<count; k++){

temp[k] = list[k];

}

//Assigning reference temp to list

list = temp;

//Setting temp equal to null

temp = null;

}

list[count] = num;

count++;

}

/\*\*

\* getFrequencyOf is passed an integer and

\* returns the number of times that number is present in the list.

\* @param num

\* @return

\*/

public int getFrequencyOf(int num){

int frequency = 0;

for (int i = 0; i<count; i++){

if(list[i] == num)

frequency++;

}

return frequency;

}

/\*\*

\* contains is passed and integer, and if that value is found within

\* the list, it will return true, if not, false.

\* @param num

\* @return

\*/

public boolean contains(int num){

for (int i = 0; i<count; i++){

if(list[i] == num)

return true;

}

return false;

}

/\*\*

\* remove(int num) is passed an integer, and removes it from the list.

\* @param num

\*/

public void remove(int num){

for (int i = 0; i<count; i++){

if(list[i] == num){

for(; i<count; i++){ //shifts everything to the left

list[i] = list[i+1];

}

list[count] = 0;

break;

}

}

count--;

}

/\*\*

\* remove() removes a random number from the list.

\*/

public void remove(){

Random rand = new Random();

int random = rand.nextInt(count);

//Removing the number at the index

for(; random<(count-1); random++){

list[random] = list[random+1];

}

count--;

}

/\*\*

\* get is passed an integer, and returns the integer at that

\* location, not that index. (The ith position).

\* @param i

\* @return

\*/

public int get(int i){

i--;

return list[i];

}

/\*\*

\* toString prints all the contents of the Scores class.

\* Which includes: the class name, the count, and the list of numbers.

\* @return

\*/

public String toString(){

String numbersInBag = "";

for(int i = 0; i<count; i++){

numbersInBag = numbersInBag + list[i] + " ";

}

return getClass().getName() + ":" + count + ":" + numbersInBag;

}

/\*\*

\* equals is passed an object and tests if it is equal

\* to the list from the scores class.

\* @param o

\* @return

\*/

public boolean equals( Object o ){

Scores s = (Scores) o;

for(int i = 0; i<count; i++){

if(list[i] != s.list[i]){

return false;

}

}

return true;

}

}

**Client:**

/\*\*

\* This class, Client, uses the Scores class to create a

\* list of integers that a number of methods are applied too.

\* The Random class is imported to generate random ints.

\*

\* @author dylca

\*/

import java.util.Random;

public class Client {

public static void main(String[] args){

Scores Main = new Scores(100);

Random rand = new Random();

//This loop fills the list with random numbers from -100 to 100.

for(int i = 0; i<100; i++){

Main.add(rand.nextInt(201) - 100);

}

//Printing the contents of the bag.

System.out.println( Main.toString() );

//Adding 86 to the end of the list.

Main.add(86);

//Printing size of bag.

System.out.println("Current size of the object: " + Main.getCurrentSize());

//Randomly removing a number from the bag.

Main.remove();

int picked = Main.get(75);

System.out.println("\nThe frequency of the number that is in the 75th position (which is: " + picked + ") is: " + Main.getFrequencyOf(picked) );

//The picked number is removed.

Main.remove( picked );

//Should be one less since picked was removed.

System.out.println("The frequency that the number occured is now: " + Main.getFrequencyOf(picked) );

//Prints the frequency of 86.

System.out.println("The frequency of 86 occuring is: " + Main.getFrequencyOf(86) );

//Returns true or false if the array contains 86.

System.out.println( Main.contains(86) );

}

}

**Output:**

run:

Scores:100:30 -5 -19 -45 98 -48 68 -4 52 -68 83 -46 -70 -32 -37 -37 95 30 94 72 -79 72 -93 -34 -53 -69 79 -29 74 72 -3 37 -67 25 -66 -7 -78 -27 100 -86 -47 91 67 94 87 -83 -70 98 -20 -24 87 48 -99 66 -93 -73 -36 -69 85 80 -67 91 55 -46 92 -49 -74 -64 -41 -3 85 -34 -49 43 -72 8 38 -5 66 47 52 69 -81 51 69 59 87 -62 9 -55 1 18 -61 -32 89 48 51 67 -84 63

Current size of the object: 101

The frequency of the number that is in the 75th position (which is: 8) is: 1

The frequency that the number occured is now: 0

The frequency of 86 occuring is: 1

true

BUILD SUCCESSFUL (total time: 0 seconds)