/\*\*

\*

\* @author Zhenhua Yang

\* @date 20190120

\*/

public interface Bag {

public int getCurrentSize();

public boolean isEmpty();

public void add( int num );

public void remove( int num );

public void remove();

public void clear();

public int getFrequencyOf( int num );

public boolean contains(int num);

@Override

public String toString();

@Override

public boolean equals(Object o);

}

import java.util.Random;

/\*\*

\*

\* @author Zhenhua Yang

\* @date 20190120

\* the Scores class implements all the methods in the interface Bag

\*/

public class Scores implements Bag {

// create instance variables

private int[] list; // the array list stores the data of Scores

private static int count; // count stores the number of elements of Scores

// default constructor

public Scores(){

list = new int[50];

count = 0;

}

// overloaded constructor

public Scores( int size ){

list = new int[size];

count = 0;

}

// implement the getCurrentSize(), isEmpty() and clear() methods

@Override

public int getCurrentSize(){

return count;

}

@Override

public boolean isEmpty(){

return count == 0;

}

@Override

public void clear(){

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

list[i] = 0;

}

count = 0;

}

// Implement the add (int num) method

@Override

public void add( int num ){

if(list.length > count){

list[count] = num;

count++;

}else{

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

for( int i = 0; i < list.length; i++ ){

temp[i] = list[i];

}

list = temp;

list[count] = num;

count++;

}

}

// Implement the getFrequencyOf (int num) method

@Override

public int getFrequencyOf( int num ){

if(contains(num)){

int countNum = 0;

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

if( list[i] == num )

countNum++;

}

return countNum;

}else{

throw new IllegalArgumentException("Cannot find the number!");

}

}

// Implement the remove (int num) and remove() methods

@Override

public void remove( int num ){

if(contains(num)){

int index = 0;

for( int i = 0; i < list.length; i++ ){

if( list[i] == num ){

index = i;

break;

}

}

for( int i = index; i < list.length; i++ ){

list[index] = list[index+1];

}

count--;

} else {

throw new IllegalArgumentException("Cannot find the number!");

}

}

@Override

public void remove(){

if(count != 0){

Random rand = new Random();

int r = rand.nextInt(((count - 1) - 0) + 1) + 0;

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

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

}

count--;

} else {

throw new ArrayIndexOutOfBoundsException("the bag is empty and nothing can be removed!");

}

}

// create the get(int i)method that returns the value in a given index position

public int get(int i){

if( i >= count ){

throw new ArrayIndexOutOfBoundsException("The index number you entered is too big!");

}else{

return list[i];

}

}

// Implement the contains(int num) method

@Override

public boolean contains(int num){

int countNum = 0;

for( int i = 0; i < list.length; i++ ){

if( list[i] == num)

countNum++;

}

return !(countNum==0);

}

// Implement the toString() and euqlas() method

@Override

public String toString(){

String str = "";

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

str += " " + list[i];

}

return getClass().getName() + "@{" + str + " }";

}

@Override

public boolean equals(Object o){

if(!(o instanceof Scores))

return false;

else{

Scores s = (Scores)o;

int d = 0;

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

d += (this.list[i] - s.list[i]);

}

return d == 0;

}

}

}

/\*\*

\*

\* @author Zhenhua Yang

\* @date 20190120

\* this class tests the methods in the Scores Object

\*/

import java.util.Random;

public class Client {

public static void main(String[] args) {

// Create an Object of Type Scores with capacity of 100

int size = 100;

Scores s = new Scores(size);

Random rand = new Random();

// populate the list in Scores object with 100 random numbers between

// -100 and +100 inclusive

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

s.add(rand.nextInt(101 - (-101) + 1 ) - 101);

}

// print the contents of the Scores object

System.out.println("Contents of Scores: " + s.toString());

// Call the add( ) method to add the number 86 to the Bag

// and print the current size of the list

s.add(86);

System.out.println("add number 86: " + s.getCurrentSize());

// randomly remove a number from the Bag

s.remove();

// Get the number at the 75th index position

int num = s.get(75);

System.out.println("Get the number at the 75th index position: " + num);

// Print the frequency that the number returned by the previous step occurs in the Bag

System.out.println("The frequency of this number: " + s.getFrequencyOf(num));

s.remove(num);

// print frequency that this number now occurs in the Bag

System.out.println("The frequency of the same number after removing the first one: " + s.getFrequencyOf(num));

// Print the frequency of the number 86

System.out.println("The frequency of this number 86: " + s.getFrequencyOf(86));

// Check if Scores contains number 86

System.out.println("Does if contain the number 86? " + s.contains(86));

System.out.println("The Score now is: \n" + s.toString());

// testing remove() method when the bag is empty.

// s.clear();

// s.remove();

}

}

Contents of Scores: Scores@{ 45 -36 4 -76 -96 39 11 -78 -27 93 86 62 -76 -49 58 66 32 -71 98 -50 55 51 -94 77 36 19 36 58 57 -31 -41 74 -16 -91 -32 -1 10 8 8 -30 42 -91 28 35 -24 62 -28 58 86 -29 -1 -100 51 11 67 22 14 -26 72 -88 48 58 -59 9 60 -90 -77 13 -1 74 -99 73 55 81 -74 -100 30 36 7 -13 67 48 80 -16 -74 -39 61 -64 30 62 4 91 -46 -44 -22 -26 -41 12 -23 -72 }

add number 86: 101

Get the number at the 75th index position: -100

The frequency of this number: 2

The frequency of the same number after removing the first one: 1

The frequency of the number 86: 2

Does it contain the number 86? true

The Score now is:

Scores@{ 45 -36 4 -76 -96 39 11 -78 -27 93 86 62 -76 -49 58 66 32 -71 98 -50 55 51 -94 77 36 19 36 58 57 -31 -41 74 -16 -91 -32 -1 10 8 8 -30 42 -91 28 35 -24 62 -28 58 86 -29 -1 51 51 11 67 22 14 -26 72 -88 48 58 -59 9 60 -90 -77 13 -1 74 -99 73 55 81 -74 -100 30 36 7 -13 67 48 80 -16 -74 -39 61 -64 30 62 91 -46 -44 -22 -26 -41 12 -23 -72 }

BUILD SUCCESSFUL (total time: 0 seconds)