IntSet Example

Breakdown:

- `public IntSet()` constructor that creates an empty set
- `public IntSet(int[] elts)` constructor that creates an IntSet from an array of integers, elts.
- `public IntSet(IntSet s)` constructor that creates an IntSet that is a copy of s.
- `public void insert(int x)` method that adds x to the IntSet, if it's not already there.
- `public boolean remove(int x)` Method that removes x from IntSet. Returns True if x was removed, false if not
- `public boolean isIn(int x)` method that returns true if the integer x is in the IntSet, false otherwhise.
- `private int indexOf(Integer x)` private method that returns the index of the first occurrence of x in the IntSet. return -1 if not present.
- `public int choose()throws EmptySetException` this method returns a random element from IntSet. If IntSet is empty, throw EmptySetException.
- `public boolean SameValues(IntSet s2)` Method that returns true if IntSet has the same values as S2, regardless of the order.

```
public class IntSet {
    /** This class provides an ADT for sets of int.
    IntSet is mutable, unbounded (=can hold elements of unspecified type)
    Class overview:
    - public IntSet()
    - public IntSet(int[] elts)
    - public IntSet(IntSet s)
    - public void insert (int x)
    - public boolean remove(int x)
    - public boolean isIn(int x)
    - private int indexOf(Integer x)
    - public int choose() throws EmptyIntSetException
    - public boolean sameValues(IntSet s2)
    **/

    /** ABSTRACTION FUNCTION:
    The set is composed of all all the integers that are represented in this.elements
    Elements is not sorted.
    INVARIANT:
    elements != null && elements contains no duplicates && elements containes boxed int (Integer)

    **/
    private Vector<Integer> elements; //private field, Vector of Integer Objects.

    /** EFFECT: initialize this to a new set, empty. **/
    public IntSet(){ // constructor of the class, takes no arguments
        this.elements = new Vector<Integer>();
    }


    /** @param: elts (= the elements to be added to the set)
    EFFECTS: initialize this to a new set, which contains each element of elts. Duplicated elements are not considered
    @throws NullPointerException if elts is null
    **/
    public IntSet(int [] elts){
        if (elts == null){
            throw new NullPointerException("elts should not be a null");
        }
        this.elements = new Vector<Integer>(); //initialize elements field to a new ampty Vector of Integer.
        for (int x:elts){ // iterate through elts array
            Integer y = new Integer(x);
            if (!this.elements.contains(y)){
                this.elements.addElement(y);
            }
        }
    }


    /** Copy constructor.
    @param s: a set to be duplicated
    EFFECT: initialize this to a new set that contains all and only the elements of s.
    @throws NullPointerException if s is null.
    **/
```