10/26/15 01:43:46 /home/15504319/DSA120/DSAAssignment/connorLib/Yard.java

```java
/***************************************************************************
*    FILE: Yard.java
*    AUTHOR: Connor Beardsmore - 15504319
*    UNIT: DSA120 Assignment S2- 2015
*    PURPOSE: Yard StockRoom for use in the DC ( GENERAL ARRAY )
*    LAST MOD: 19/10/15
*  REQUIRES: NONE
***************************************************************************/
package connorLib;

public class Yard implements IStockRoom
{
     //CLASS FIELDS
     private Carton[] array;
     private int count;

     //CLASS CONSTANTS
     private static final int MAX_VALID_CAP = 10000;
    private static final int MIN_VALID_CAP = 1;
    private static final String YARD = "Y";
//---------------------------------------------------------------------------
    //ALTERNATE Constructor
    //IMPORT: maxCap (int)
    //ASSERTION: Array allocated 'maxCap' elements.Count to default 0

    public Yard(int maxCap)
    {
        //maxCap must be a value between 2 and 10,000
        if ( (maxCap < MIN_VALID_CAP) || ( maxCap > MAX_VALID_CAP ) )
        {
            throw new IllegalArgumentException("Array Capacity Not Valid");
        }
        array = new Carton[maxCap];
        count = 0;
    }
//---------------------------------------------------------------------------
    //ACCESSOR getCount
    //EXPORT: count (int)

    public int getCount()
    {
        return count;
    }
//---------------------------------------------------------------------------
    //ACCESSOR getCapacity
    //EXPORT: array length (int)

    public int getCapacity()
    {
        return array.length;
    }
//---------------------------------------------------------------------------
    //ACCESSOR isEmpty
    //EXPORT: empty (boolean)

    public boolean isEmpty()
    {
        return ( count == 0 );
    }
//---------------------------------------------------------------------------
    //ACCESSOR isFull
    //EXPORT: full (boolean)


    public boolean isFull()
    {
        //Length stored in array itself
        return ( count == array.length );
    }
//---------------------------------------------------------------------------
    //MUTATOR addCarton
    //IMPORT: inCart (Carton), index (int)
    //PURPOSE: Add new Carton to array

    public void addCarton(Carton inCart, int index)
    {
        //Can't add anymore values if array is full
        if ( isFull() )
        {
            throw new IllegalStateException("Array Is Full. Cannot Add");
        }

        //Can't add value if index is full
        if ( array[index] instanceof Carton )
        {
            throw new IllegalStateException("Index is occupied. cannot add");
        }

        array[index] = inCart;
        inCart.setRIndex(index);
        count++;
    }
//---------------------------------------------------------------------------
    //MUTATOR addCarton
    //IMPORT: inCart (Carton)
    //PURPOSE: Add new Carton to array

    public void addCarton(Carton inCart)
    {
        //Add to Array, increment counter
        int i = 0;
        boolean done = false;
        while ( (i < array.length) && (done == false) )
        {
            if ( array[i] == null )
            {
                array[i] = inCart;
                done = true;
            }
            i++;
        }

        inCart.setRIndex(i - 1);
        count++;
    }
//---------------------------------------------------------------------------
    //MUTATOR removeCarton
    //IMPORT: index (int)
    //EXPORT: outCart (Carton)
    //PURPOSE: Remove Carton from the specific index in array

    public Carton removeCarton(int index)
    {
        //Can't remove anymore values if array is empty
        if ( isEmpty() )
        {
            throw new IllegalStateException("Yard Is Empty. Cannot Remove");
        }

        Carton outCart = array[index];
        array[index] = null;
        outCart.setDIndex(-1);
        outCart.setRIndex(-1);
        count--;
        return outCart;
    }
//---------------------------------------------------------------------------
    //MUTATOR removeCarton
    //EXPORT: outCart (Carton)
    //PURPOSE: Remove front Carton from the array

    public Carton removeCarton()
    {
        //Can't remove anymore values if array is empty
        if ( isEmpty() )
        {
            throw new IllegalStateException("Yard Is Empty. Cannot Remove");
        }

        int ii = 0;
        boolean done = false;
        Carton outCart = array[0];

        //Iterate until first non null element in the Yard
        while ( ( ii < array.length ) && ( done == false ) )
        {
            if ( outCart == null )
            {
                ii++;
                outCart = array[ii];
            }
            else
            {
                done = true;
            }
        }

        array[ii] = null;
        outCart.setDIndex(-1);
        outCart.setRIndex(-1);
        count--;
        return outCart;
    }
//---------------------------------------------------------------------------
    //ACCESSOR toString
    //EXPORT: stateString (String)
    //PURPOSE: Prints out room Carton's in  DC Geometry file format

    public String toString()
    {
        String stateString = YARD;
        for (int i = 0; i < array.length; i++)
        {
            //Accounts for empty slots via ":" print outside
            stateString += ":";
            if ( array[i] != null )
            {
                stateString += array[i].getNote();
            }
        }
        return stateString;
    }
//---------------------------------------------------------------------------
    //ACCESSOR contentString
    //EXPORT: stateString (String)
    //PURPOSE: Output All Carton Contents in Queue As a String

    public String contentString()
    {
        String stateString = "";
        for ( int ii = 0; ii < array.length; ii++ )
        {
            if ( array[ii] != null )
            {
                stateString += array[ii].toString() + "\n";
            }
        }
        return stateString;
    }
//---------------------------------------------------------------------------
}
```