```java
public class SortedStack<Integer> implements CarlStack<Integer> {

    private Stack<Integer> mainStack = new Stack();
    private Stack<Integer> tempStack = new Stack();

    @Override
    public void push(String item) {









    }

    @Override
    public String peek() throws EmptyStackException {









    }

    @Override
    public String pop() throws EmptyStackException {









    }

    @Override
    public boolean isEmpty() {









    }
}
```

```java
public class SuperArrayStack<T> {

    // Stores items
    private T[] contents;
    private T[] mysteryArray;

    // Tracks how many items are in my stack
    private int count;
    private int mysteryInt;

    public SuperArrayStack(int size) {
        contents =  (T[]) new Object[size];
        count = 0;
        mysteryArray =  (T[]) new Object[size * 2];
        mysteryInt = 0;
    }

    public void push(T item) {
        if(count == contents.length) {




        }

        contents[count] = item;
        count++;









    }


    public T pop() throws EmptyStackException{
        if( count == 0) {
            throw new EmptyStackException("You can't pop from an empty stack");
        }







        T dataToReturn = contents[count - 1];
        count = count - 1;
        return dataToReturn;

    }

    public T peek() throws EmptyStackException {
        if(count == 0) {
            throw new EmptyStackException("You can't peek at an empty stack");
        }

        return contents[count - 1];
    }
```