Assignment	Write a Stack class in the package **ku.util**. This class implements the stack datatype.
What to Submit	Commit your project to Bitbucket as project name PA2. Both problem 1 (ArrayIterator) and this problem (Stack) are part of the same project. Share the project with the TAs.

The Java API has a Stack interface, but some of the method names are *inconsistent* with other collections, and it has a search method which doesn't make sense for a Stack. So, we will define our own Stack type.

1. Stack class

1.1 Define a Stack class in the package **ku.util**, with the methods shown below. The Stack class has a type parameter (**T**) so that it can be used to hold any kind of data we want.

The Stack methods are

int capacity( )	the maximum number of elements that this Stack can hold. Return -1 if unknown or infinite.
boolean isEmpty( )	true if stack is empty.
boolean isFull( )	true if stack is full.
T peek( )	return the item on the top of the stack, without removing it. If the stack is empty, return **null**.
T pop( )	return the item on the top of the stack, and remove it from the stack. Throws: `EmptyStackException` if stack is empty.
void push(T obj)	push a new item onto the top of the stack. If the stack is already full, this method does nothing... its the programmer's responsibility to check isFull() before trying to push something onto stack. The parameter (obj) must not be null. Throws: `InvalidArgumentException` if parameter is null.
int size( )	return the number of items in the stack. Returns 0 if the stack is empty.

We want the stack to be able to hold elements of any kind, so define the class with a type parameter (T), like this:

```
public class Stack<T> {

    private T[] items;  // items on the stack
```

1.2 Write a public constructor that specifies the capacity of the stack:

Stack(int capacity)	create a new stack with the given capacity. Capacity must be positive.

1.3 Implement the stack using an array for storage.

To create an array of references using a type parameter (T), use code like this:

```
items = (T[]) new Object[capacity];
```

1.4 Write good Javadoc comments for the class and every method!

The first sentence of each Javadoc comment should be a complete sentence. Try to write sentences that describe what the class or method does. Look at the Javadoc for JDK's Stack class for examples.

1.5 Test your Stack. A JUnit test class named StackTest.java will be posted in the "week4" folder to help testing, but you should also test your own code.

Example using BlueJ Interactive Mode

```
> import ku.util.*;
> Stack<String> stack = new Stack<String>(2); // pretty small
> stack.isEmpty()
true
> stack.size()
0
> stack.push("cake");
> stack.push("ice cream");
> stack.size( )
2
> stack.isFull( )
true
> stack.push("yogurt");    // discarded - stack is already full
> stack.pop( )
"ice cream"
> stack.size( )
1
> stack.peek( )
"cake"
> stack.pop( )
"cake"
> stack.pop( )
java.util.EmptyStackException thrown
> stack.peek( )
null
```