Stacks and Queues

- What are Stacks?
- Definition (Intro to ADTs)
- Implementation
- Palindrome testing using Stacks.

What is a Stack?

Last in First Out - LIFO

Used throughout computer science, Stacks help implement function calls allowing recursion.

Definition

A Stack of integers has the following methods (or equivalent version):

- `Stack()` create an empty stack
- `isEmpty()` test if a stack is empty
- `int stackTop()` look at the top element of the stack.
- `int pop()` return the top element of the stack and remove it.
- `push(int)` add an element to the top of the stack.

Implementation

```
class Stack  {
    class EmptyStack extends Throwable {};

    private Node1 top_of_stack;

    public Stack() {
        top_of_stack = new Node1();
    }
    public boolean isEmpty() {
        return(top_of_stack.isEmpty());
    }
```

Poping items

```
public int pop() throws Stack.EmptyStack {
    if (top_of_stack.isEmpty()) {
        throw (new Stack.EmptyStack());
    } else {
        int return_data = top_of_stack.first();
        top_of_stack = top_of_stack.rest();
        return(return_data);
    }
}
```


Test Code

```
class StackTest {
    public static void main(String argv[] ) {
        Stack s = new Stack();
        s.push(1);  s.push(45); s.push(32);
        try {
            System.out.println(s.stackTop());
            System.out.println(s.pop());
            System.out.println(s.pop());
            System.out.println(s.pop());
            System.out.println(s.pop());
        } catch(Stack.EmptyStack e) {
            System.out.println("Oops, an empty stack.");
        }    } }
```


Checking the Top of a stack

```
public int stackTop() throws Stack.EmptyStack {
    if (top_of_stack.isEmpty()) {
        throw (new Stack.EmptyStack());
    } else {
        return(top_of_stack.first());
    }
}
```


Pushing - Using concat

```
    public void push(int data) {
        top_of_stack = top_of_stack.concat(data);
    }
}
```

Palindrome - Testing

A Palindrome is a string of letters which reads the same backwards as well as forwards.

We'll Write a simple Palindrome tester using a stack.

If we push every thing on the stack, the pop it off it will be in the reverse order.

We really need a stack of strings, but we'll use integers to represent characters.

Palindrome Tester

```
public static boolean palindrome(String str)
throws Stack.EmptyStack {
  Stack s = new Stack();
  for(int i=0; i<str.length() ; i++) {
      char c = str.charAt(i);
      s.push((int) c);
  }
```



```
        for(int i=0; i<str.length(); i++) {
            int str_c = str.charAt(i);
            int stack_c = s.pop();
            if(str_c != stack_c) return(false);

        }
        return(true);
    }
```

How would you modify it to ignore spaces and difference in case?