Stacks & Queues – Implementations

```
public class Stack implements Serializable
{
   public Object push( Object x )
   {
      items.add( x );
      return x;
   }
   public Object pop( )
   {
      if( isEmpty( ) )
         throw new EmptyStackException( );
      return items.remove( items.size( ) - 1 );
   }
   public boolean isEmpty( )
      {  return size( ) == 0;  }

   private ArrayList items;
   // LinkedList????
}
```

```
public class ListQueue implements Queue
{
   public boolean isEmpty( )
   {  return front == null; }
   public void enqueue( Object x )
   {  if( isEmpty( ) )
         back = front = new ListNode( x );
      else              // Regular case
         back = back.next = new ListNode( x );
   }
   public Object dequeue( )
   {  if( isEmpty( ) )
         throw new UnderflowException( "" );
      Object returnValue = front.element;
      front = front.next;
      return returnValue;
   }

   private ListNode front;
   private ListNode back;
}
```

Stacks: Application 1

- Check balanced parentheses
  - (())()(()(()))
  - ((())()))()(()

```
While (expr.nextToken())
{
   if next token is "("
            push "(" on stack;
   else
            if stack is not empty
                     pop "(" from stack;
            else report error;
}
If stack is not empty
            report error;
```

Stacks: Application 2

Evaluate Postfix Expressions

1 2 3 + *

= (1* (2 + 3))

4 1 2 2 3 * ^ + -1 * +

= ?

```
While (expr.nextToken())
{
   if next token is an operand
      push operand on stack;
   else if next token is an operator Op
      {
         pop Val1 from stack;
         pop Val2 from stack;
         compute Val1 Op Val2;
         push result on stack;
      }
   if stack has only one item
      pop value and return as Value of expr;
   else report error;
}
```

Stacks – Applications 3

- Convert Infix Expressions to Postfix

Recursion

- **Example 1:** Fibonacci Numbers
  1, 2, 3, 5, 8, 13, 21, 34, 55, 89, …

```
public static long fib(int n)
{
        if (n <= 1)
                return n;
        else
                return fib(n-1) + fib(n-2);
}
```

- **Example 2:** Towers of Hanoi