```
public class IntStack {
        // Overview:  IntStack is a mutable, bounded stack of Integers
       private Vector<Integer> elems;
        private int capacity;
        public  IntStack(int c ) {
             // REQUIRES:
              // EFFECTS:
               capacity = c;
              elems = new Vector<Integer> (c);
      }
      public void push(int e) throws  StackOverFlowException {
              // MODIFIES:
              // EFFECTS:

              if (elems.size() == capacity ) throw new StackOverFlowException("Stack.push");
              elems.add( new Integer(e) );
     }
      public void pop() throws StackUnderFlowException {
              // MODIFIES:
              // EFFECTS:

              if (elems.size() == 0) throw new StackUnderFlowException("Stack.pop");
              elems.remove( elems.size() – 1 );
     }
     public int size( ) {
              // EFFECTS:
              return elems.size();     }
};
```