Instructions

Bring this handout to tutorial with the answers filled in.

Popping n elements

Recall the Stack interface:

```java
public interface Stack{
    // Push o onto me.
    // Precondition: I am not full.
    public void push (Object o);

    // Pop off and return my top element.
    // Precondition: I am not empty.
    public Object pop ();

    // Return whether or not I am empty.
    public boolean isEmpty ();

    // Return whether or not I am full.
    public boolean isFull ();

    // Empty me out.
    public void clear ();
}
```

Suppose we are writing some other class that uses stacks and often needs to pop a bunch of items at once. We might write the following method and put it into our class:

```java
// Pop n items from s and discard them.
public static void popn (Stack s, int n){
    for (int i=1, i<=n; i++)
        s.pop();
}
```

1. Write appropriate “preconditions” for this method. If I satisfy your preconditions when I call \texttt{popn()}, it should not crash and it should give me the answer promised in its comments.
Mystery method

Trace the following method on some examples to see what it does. Then answer the questions below.

```java
public static int mystery (int[] list, int i){
    int best = list[i];
    i++;
    while (list[i] != -1) {
        if (list[i] < best)
            best = list[i];
        i++;
    }
    return best;
}
```

2. Write an appropriate comment specifying what this method returns.

3. Write appropriate preconditions for this method. Again, if I satisfy your preconditions when I call the method, it should not crash and it should give me the answer you just promised.

IntSet class

The following class keeps track of a set of integers in an interesting way.

```java
public class IntSet {
    private boolean[] inSet;

    public IntSet (int n){
        inSet = new boolean[n];
        for (int i=0; i<inSet.length; i++){
            inSet[i] = false;
        }
    }
    // Add i to my set.
    public void add (int i){
        inSet[i] = true;
    }
    // Return whether or not i is in my set.
    public boolean member (int i){
        if (inSet[i] == true)
            return true;
        else
            return false;
    }
}
```

4. Write an appropriate representation invariant for this class.

5. Rewrite the body of `member()` in one line. Hint: This has to do with using booleans properly.