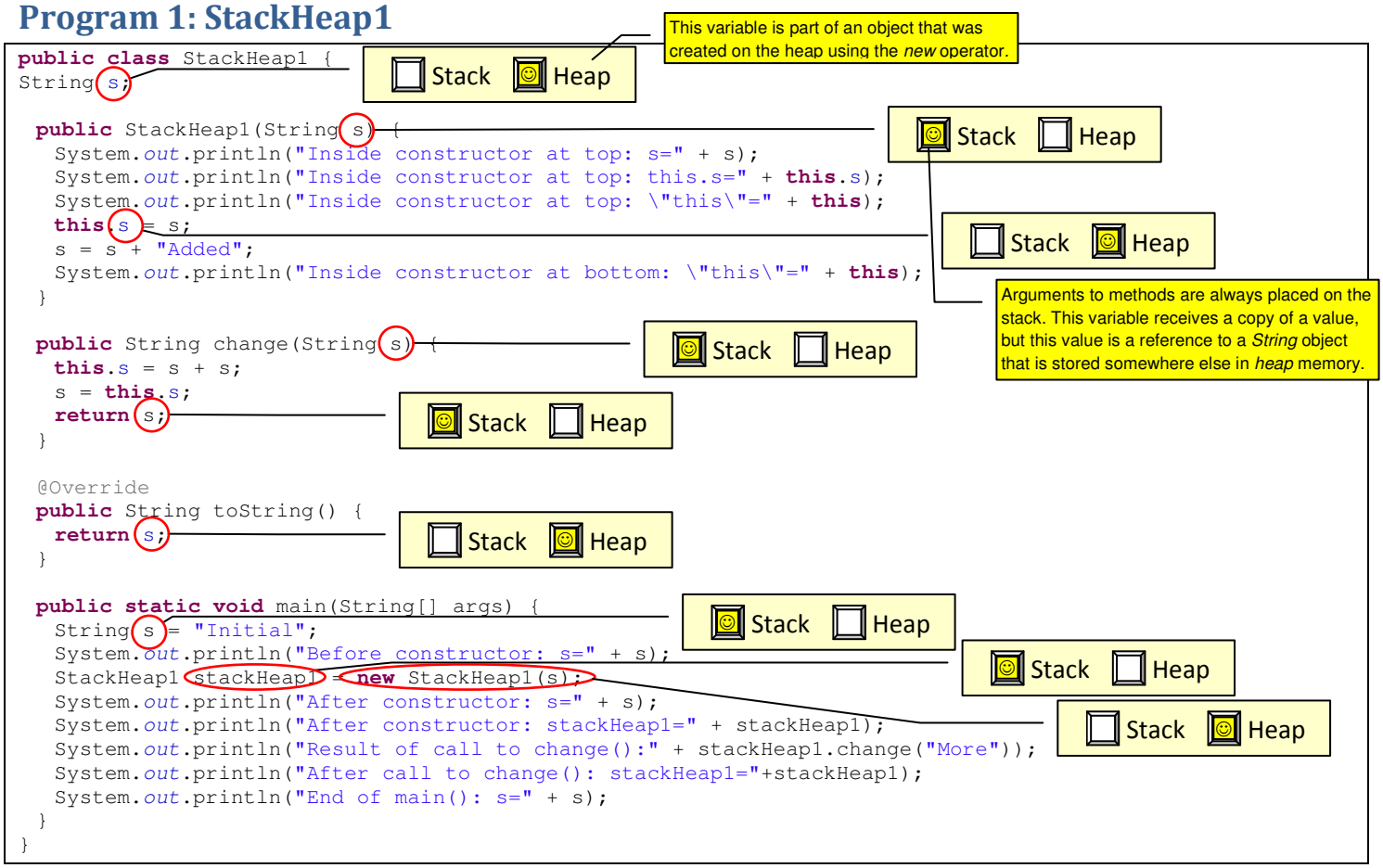
Object-Oriented Programming (in Java)

Exercise: Stack / Heap Variables: Solution

Work through these programs on paper to determine the output that will be generated. Also identify where each variable will be located: *stack* or *heap*. The key in understanding *stack* / *heap* storage relates directly to the issue of persistence, that is, how long does a variable stay in memory. Later, run the programs in *Eclipse* to check your work.

Program 1: StackHeap1

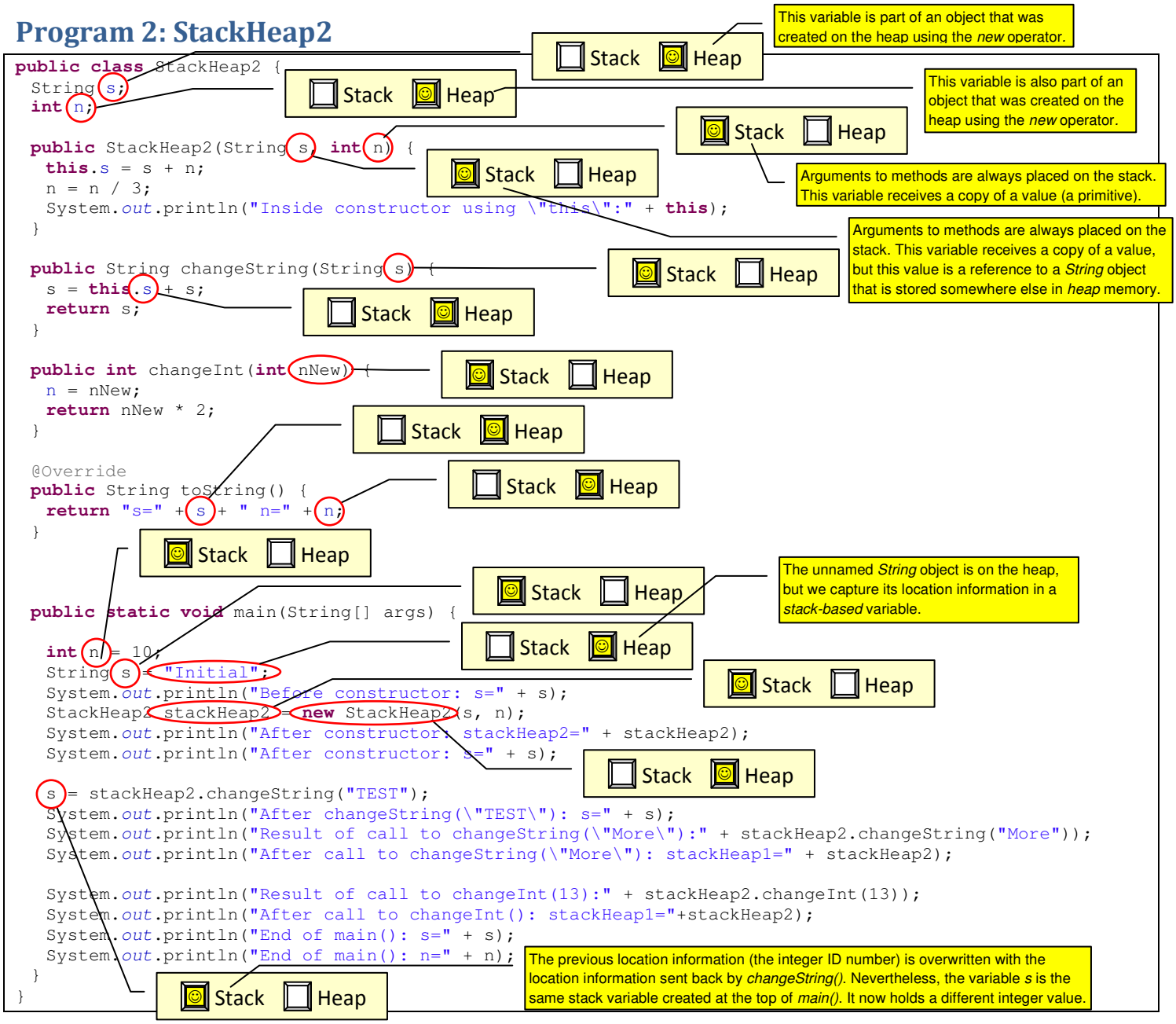



```
public class StackHeap1 {
String s;

  public StackHeap1(String s) {
    System.out.println("Inside constructor at top: s=" + s);
    System.out.println("Inside constructor at top: this.s=" + this.s);
    System.out.println("Inside constructor at top: \"this\"=" + this);
    this.s = s;
    s = s + "Added";
    System.out.println("Inside constructor at bottom: \"this\"=" + this);
  }

  public String change(String s) {
    this.s = s + s;
    s = this.s;
    return s;
  }

  @Override
  public String toString() {
    return s;
  }

  public static void main(String[] args) {
    String s = "Initial";
    System.out.println("Before constructor: s=" + s);
    StackHeap1 stackHeap1 = new StackHeap1(s);
    System.out.println("After constructor: s=" + s);
    System.out.println("After constructor: stackHeap1=" + stackHeap1);
    System.out.println("Result of call to change():" + stackHeap1.change("More"));
    System.out.println("After call to change(): stackHeap1="+stackHeap1);
    System.out.println("End of main(): s=" + s);
  }
}
```

Expected Program Output Based on Paper-Trace of Execution

```
Before constructor: s=Initial
Inside constructor at top: s=Initial
Inside constructor at top: this.s=null
Inside constructor at top: "this"=null
Inside constructor at bottom: "this"=Initial
After constructor: s=Initial
After constructor: stackHeap1=Initial
Result of call to change():MoreMore
After call to change(): stackHeap1=MoreMore
End of main(): s=Initial
```

Program 2: StackHeap2



Expected Program Output Based on Paper-Trace of Execution

```
Before constructor: s=Initial
Inside constructor using "this":s=Initial10 n=0
After constructor: stackHeap2=s=Initial10 n=0
After constructor: s=Initial
After changeString("TEST"): s=Initial10TEST
Result of call to changeString("More"):Initial10More
After call to changeString("More"): stackHeap1=s=Initial10 n=0
Result of call to changeInt(13):26
After call to changeInt(): stackHeap1=s=Initial10 n=13
End of main(): s=Initial10TEST
End of main(): n=10
```