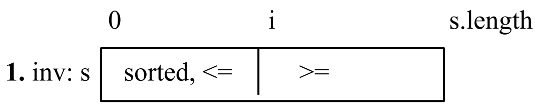
1. 

```
/** Sort s using selection sort, .... */
public static void selectionSort(int[] s) {
    for (int i= 0; i < s.length; i= i+1) {
        int m;
        Set m so that m in i..b.length-1 and s[m] is
           lexically the smallest in b[i..s.length-1];
        Swap s[i] and s[m]
    }
}
```

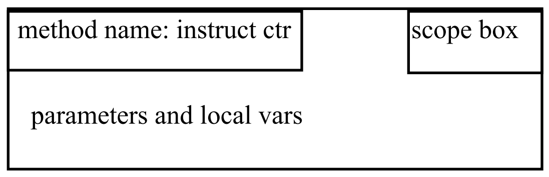
2.
```
public static int sumEntries(String[][] b, int j) {
    try {
        int sum= 0;
        // inv: sum = sum of ints in b[j][0..i-1]
        for (int i= 0; i<  b[j].length; i= i+1) {
           sum= sum + Integer.parseInt(b[j][i]);
        }
        return sum;
    } catch (NumberFormatException nfe) {
         throw new IllegalArgumentException(
                      "an element of b[j] is not an int");
    }
  }
```

3.
```
/** = Number of nulls in b */
  public static int nulls(Object b) {
    if (b == null) return 1;
    if (!(b instanceof Object[])) return 0;
    int n= 0;
    Object[] c= (Object[]) b;
    // inv: n = number of nulls in b[0..k-1]
    for (int k= 0; k < c.length; k= k+1) {
       n= n + nulls(c[k]);
    }
    return n;
  }
```

4. (a) The purpose of a constructor is to initialize the fields of a newly created object so that the class invariant is true.

```
public CourseMtg(String name, String[] b) {
    super();
    this.name= name;
    selectionSort(b);
    instructors= b;
}
```

(b)
```
public boolean add(String n) {
      if (contains(n)) { return false; }
      else { super.add(n); return true; }
    }
```



scope box contains the name of object or file drawer that contains the method being called

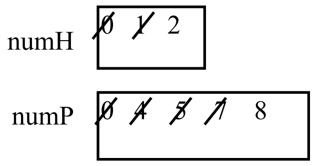
(c)
```
public boolean equals(Object ob) {
    if (!(ob instanceof CourseMtg)) { return false; }
    CourseMtg c= (CourseMtg) ob;
    if (!c.name.equals(name) ||
        c.instructors.length != instructors.length) {
      return false;
    }
    // inv: instructors[0..i-1] = c.instructors[0..i-1]
    for (int i= 0; i < instructors.length; i= i+1) {
      if (!c.instructors[i].equals(instructors[i]))
         { return false; }
    }
    return true;
  }
```

(d)
```
public String toString() {
    return nameAndInstr() + "\n" + super.toString();
  }
```

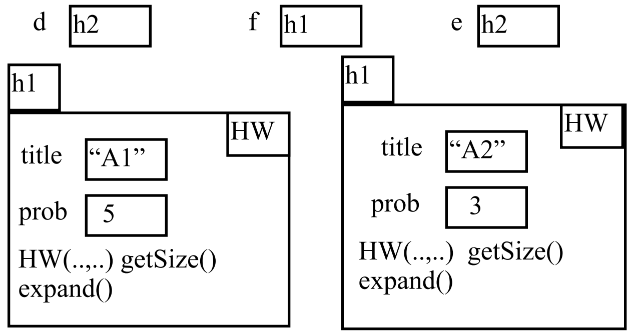
5.
```
public Section(String n, String[] ls, Lecture lec) {
    super(n, ls);
    mainLecture= lec;
    lec.secList.add(this);
  }

  public void add(String n) {
    mainLecture.add(n);
    super.add(n);
  }
```

6. (a) NumH= numH + 1;
numP= numP + n;
numP= numP + 1;


numH
$\cancel{0}$ $\cancel{1}$ 2
numP
$\cancel{0}$ $\cancel{4}$ $\cancel{5}$ $\cancel{7}$ 8

(c) false false



7.(a) Write the method to be called; inform Java that the class contains the method; and register the object as a listener.
7.(b) To the left.
7.(c) Parameter: variable that is declared within the (...) of a method header.
Argument: expression within the (...) of a method call.
Local var: a variable declared within a method body.
Inside-out rule: to find the declaration corresponding to a reference to variable or method, look in the construct in which the reference occurs and within successive enclosing constructs until it is found.