1. In each box below, briefly describe in conversational English what the following methods do. We are looking for a high-level summary (e.g., your answer could be a comment describing the method's purpose, such as "finds the square root of x").

```
public static int m( int a, int b ) {
        int s = 0;
        for (int i = 0; i < b; ++i) {
                s = s + a;
        }
        return s;
}
```

m() Returns:

```
public static boolean p( int n ) {
        boolean b = true;
        for (int i = 2; i < n; ++i) {
                int r = n % i;
                if ( r == 0 ) {
                        b = false;
                }
        }
        return b;
}
```

p() returns whether

```
public static boolean s(int[] a) {
        int n = a.length;
        boolean b = true;
        for (int i = 1; i < n; ++i) {
                if (a[i-1] > a[i]) {
                        b = false;
                }
        }
        return b;
}
```

s() returns whether a is

2. Write a boolean static method mostlyPositive() that takes a single int[] parameter a and returns whether are are more positive values than non-positive values in array a. An example use would be:

```
int x = stdin.nextInt(); int y = stdin.nextInt();
int z = stdin.nextInt(); int[] list = { x, y, z } ;
boolean b = mostlyPositive( list );
```

3. Assume m is a previously-defined and initialized two-dimensional array (i.e., a matrix). Explain why the following code segment does not correctly report whether all elements in matrix m are nonzero.

```
int rows = m.length;
int columns = m[0].length;
boolean nonzero = false;
for ( int r = 0; r < rows; ++r) {
        for ( int c = 0; c < columns; ++c) {
                int v = m[ r ][ c ];
                if ( v != 0 ) {
                        nonzero = true;
                }
        }
}
System.out.println( "is nonzero: " + nonzero );
```