1. Write a recursive procedure for method `isOdd` that takes an `int` parameter `n` and returns a (boolean value) true if and only if `n` is odd. (You can assume `n` is always a positive integer.) Your procedure cannot use modulus or division operations.
   (Hint: It’s OK if your procedure ends up running in time proportional to `n` itself...)

   ```java
   public boolean isOdd( int n )
   {
   }
   ```
2. Provide code to method `diagonalLine` of class `Point` which takes two Point objects \( p_1 = (x_1, y_1) \), and \( p_2 = (x_2, y_2) \) and returns an array of one thousand instances of a Point object, whose coordinates range from \((x, y) = (x_1, y_1)\) to \((x, y) = (x_2, y_2)\), i.e. the points in the array are evenly spaced on a diagonal line connecting point \( p_1 \) and \( p_2 \). Don’t worry if the coordinates of the last point in the array is only approximately equal to the coordinates of point \( p_2 \). Also, use the `while` construct for a loop, and not the `for` construct.

```java
public class Point {
    private double x, y;

    public double getX() { return x; }
    public double getY() { return y; }
    public void setX( double newX ) { x = newX; }
    public void setY( double newY ) { y = newY; }

    public Point[] diagonalLine( Point p1, Point p2 )
    {
        return null;
    }
}
```
3. What is the output of a call to the method `scopingTest` in the code below? For example, if variable `example` is initialized as an object of type `Scoping`, this method can be invoked by `example.scopingTest()` procedure call. (By the way, it’s not a great idea to re-use variable names for both class variables, formal arguments, and local variables, as is done in the code below.)

```java
public class Scoping {
    String s = "Tolstoy";

    private void stringMethod1() {
        System.out.println(s);
        s = "Naipul";
    }

    private void stringMethod1(String s) {
        System.out.println(s);
        s = "Colette";
    }

    private void stringMethod2() {
        String s = "Mishima";
        System.out.println(s);
        s = "Austen";
    }

    public void scopingTest() {
        String s = "Steinbeck";

        stringMethod1();
        stringMethod1( s );
        stringMethod1();
        stringMethod2();
        stringMethod1();
        System.out.println( s );
        System.out.println( this.s );
    }
}
```
4. What is the output of the function `test()`?

class TestParameterPassing {
    public void test() {
        int[] a = new int[10];
        int[] b = new int[10];

        for (int i = 0; i < a.length; i++) {
            a[i] = i;
            b[i] = -i;
        }

        System.out.println(a[2]);
        change1(a);
        System.out.println(a[2]);
        change2(a[2]);
        System.out.println(a[2]);
        change3(a);
        System.out.println(a[2]);
        a = b;
        System.out.println(a[2]);
    }
}

class TestParameterPassing {
    public void change1(int[] b) {
        for (int i = 0; i < b.length; i++)
            b[i] = b[i] * b[i];
    }

class TestParameterPassing {
    public void change2(int b) {
        b = b * b;
    }

class TestParameterPassing {
    public void change3(int[] b) {
        int[] a = new int[b.length];

        for (int i = 0; i < b.length; i++)
            a[i] = b[i] * b[i];
        b = a;
    }
}