Names: _____

More fun with recursion. Develop these recursive methods from CodingBat.com. On your own, go to CodingBat and try them.

Work with a partner!

1. We have a number of bunnies and each bunny has two big floppy ears. We want to compute the total number of ears across all the bunnies recursively (without loops or multiplication).

bunnyEars(0) → 0
bunnyEars(1) → 2
bunnyEars(2) → 4

First – What is your base case? _____

Next – What is the refinement? _____

```
public int bunnyEars(int bunnies)
{









}
```

2. Given **base** and **n** that are both 1 or more, compute recursively (no loops) the value of base to the n power, so powerN(3, 2) is 9 (3 squared).

powerN(3, 1) → 3
powerN(3, 2) → 9
powerN(3, 3) → 27

First – What is your base case? _____

Next – What is the refinement? _____

```
public int powerN(int base, int n)
{









}
```

3. We have triangle made of blocks. The topmost row has 1 block, the next row down has 2 blocks, the next row has 3 blocks, and so on. Compute recursively (no loops or multiplication) the total number of blocks in such a triangle with the given number of rows.

triangle(0) → 0
triangle(1) → 1
triangle(2) → 3

First – What is your base case? _____

Next – What is the refinement? _____

```
public int triangle(int rows)
{









}
```

4. Given a string, compute recursively a new string where all the lowercase 'x' chars have been moved to the end of the string.

endX("xxre") → "rexx"
endX("xxhixx") → "hixxxx"
endX("xhixhix") → "hihixxx"

First – What is your base case? _____

Next – What is the refinement? _____

```
public String endX(String str)
{









}
```

5. Given an array of ints, compute recursively if the array contains a 6. We'll use the convention of considering only the part of the array that begins at the given index. In this way, a recursive call can pass index+1 to move down the array. The initial call will pass in index as 0.

array6({1, 6, 4}, 0) → true
array6({1, 4}, 0) → false
array6({6}, 0) → true

First – What is your base case? _____

Next – What is the refinement? _____

```
public boolean array6(int[] nums, int index)
{




}
```

6. Given an array of ints, compute recursively the number of times that the value 11 appears in the array. We'll use the convention of considering only the part of the array that begins at the given index. In this way, a recursive call can pass index+1 to move down the array. The initial call will pass in index as 0.

array11({1, 2, 11}, 0) → 1
array11({11, 11}, 0) → 2
array11({1, 2, 3, 4}, 0) → 0

First – What is your base case? _____

Next – What is the refinement? _____

```
public int array11(int[] nums, int index)
{




}
```