Recursion Practice Problems

1. Write a function (or set of functions) that takes in an int parameter and returns an int that is the sum of all the digits added together.  
   (Ex: 1,234 -> 10; 121 -> 4; 9,823,555 -> 37)
2. Write a function (or set of functions) that takes in an array of ints as a parameter and returns an int that is the sum of all the elements multiplied together.  
   (Ex: [1,2,3,4]-> 24; [9,4,2] -> 72; [7,5,8,6,3]-> 5,040)
3. Write a function (or set of functions) that takes in a String as a parameter and returns an String that is comprised of every odd letter (the first letter in the string is considered even).  
   (Ex: “hello”-> “el”; “valiant” -> “ain”; “tarnishing”-> “ansig”)
4. Write a function (or set of functions) that takes in an array of ints as a parameter and modifies that array such that each index is set to the value multiplied by the place in the array.  
   (Ex: [500, 3, 6, 7]-> [0, 3, 12, 21]; [4, 5, 7] -> [0, 5, 14]; [9, 8, 9, 6]-> [0, 8, 18, 18])
5. Draw a recursion diagram for fun2(7). What does this function do?   
     
   /\* Assume that n is greater than or equal to 1 \*/

static void fun2(int n)

{

if(n == 0)

{

return;

}

//Note that this is integer division-the result will be

//rounded down to the nearest int.

fun2(n/2);

System.out.println(n%2);

}

1. This function is a recursive function that prints off \*s. What is the general pattern of \*s that are printed? How many asterisks would starPrint(6) yield?

static void fun1(int n)

{

int i = 0;

if (n > 1)

{

fun1(n - 1);

}

for (i = 0; i < n; i++)

{

System.out.print(" \* ");

}

}

1. Draw a recursion diagram for this function, given the argument “cat”. What is the output of this program for that input?

static void abc(String s)

{

if(s.length() == 0)

{

return;

}

abc(s.substring(1));

abc(s.substring(1));

System.out.print(s.charAt(0));

}