**CIS 351 – Activity wk-11a**

When a method makes a call to itself this is referred to as ***recursion***. To define a recursive method in Java, you should write an if-statement that checks for the base case. When the operation is *not* the base case, you call to the method you are writing (known **recursive call**).

Consider the following factorial definition:

|  |
| --- |
| public static int factorial(int n) {  System.out.println("n is " + n);  if (n == 0)  return 1;  else {  System.out.println("need factorial of "  + (n-1));  int answer = factorial(n-1);  System.out.println("factorial of " +(n-1) +  + " is " + answer);  return answer \* n;  }  }  public static void main(String[] args) {  System.out.println(factorial(3));  } |

1. Circle the recursive call in the above code.

factorial(n-1)

1. How many distinct calls are made to the factorial method to calculate the factorial of 3? Identify the value of the parameter n for each of these separate calls.

4 calls, factorial (3) = 6, factorial (2) = 2, factorial (1) = 1, factorial (0) = 1

1. Consider the output for the main method below. For each distinct factorial method call, identify all the output lines it printed. In other words, which lines were printed by factorial(3), which lines were printed by factorial(2), and so on.

n is 3

need factorial of 2

n is 2

need factorial of 1

n is 1

need factorial of 0

n is 0

factorial of 0 is 1

factorial of 1 is 1

factorial of 2 is 2

6

1. Consider two different ways to show how to calculate 
2. Write out all numbers that explicitly need to be summed

 = 1+2+3+4

1. Write an expression showing how this sum can be calculated in terms of a “simpler” sum.

 = n = n+1

1. What is the “value” of the base case of a summation expression?

0

1. Create a static recursiveSummationmethod that takes a single int parameter *n* and returns the summation (as an int) using recursion. Your method should have an if-else statement and NO loops. Your method can return any number you wish for negative numbers, but it should not crash. Your method should *not* contain all the System.out.println commands that were included as part of the original factorial method.

Public static recursiveSummation(int n) {

Int sum = 0;

if (n ) {

System.out.println(n);

}

Else{

Sum += n;

}

recursiveSummation(n-1);

}

Below is a slightly different version of the factorial method along with its stack diagram for a call from main using factorial (3).





1. Why are there no values for recurse and result in the stack diagram for the last call to factorial?

There are no values because there is only the base case and cannot do the other cases when n equals another number.