These methods are required to be **fully recursive** in that they contain **no loops at all;**

neither for, while or do-while.

**1.** *void* ***listNumbers(int start, int end)***that outputs the numbers from *start* to *end* to console. Write one version that outputs the numbers in ascending order, and another that outputs them in descending order.

**2.** ***char\* repeat(char\* s, int n)***that creates and returns a new string that is made by concatenating *n* copies of the parameter string *s*. For example, calling this method with the parameters “Hello” and 3 would return the string “HelloHelloHello”. If *n* equals zero, the method should return the empty string.

**3.** ***int min(int[] a, int start, int end)***that returns the smallest element between the indices *start* and *end* in the parameter array a.

**4. *int mul(int a, int b)***that computes the product of two integers *a* and *b*. The only arithmetic

operation that you are allowed to use in this problem is addition +.

**5. *double power(double a, int b)***that calculates the power *ab.* You are allowed to use the

multiplication operator \*.

**6. *double harmonicSum(int n)***that calculates and returns the sum 1 + ½ + 1/3 + ... + 1/*n*.

**7. *int sumOfDigits(int n)***that computes and returns the sum of digits of the positive integer *n*. For example, when called with the parameter 12345, this method would return 15.

**8. *int reverseDigits(int n)***that returns the positive integer that you get when you reverse the digits of parameter *n*. For example, when called with the parameter 12345, this method would return 54321. (Do this with proper integer arithmetic instead of just simply converting to *String*,

reversing that and then using *parseInt* to extract the result.)

**9. *bool binarySearch(int[] a, int start, int end, int x)***that uses the binary search algorithm to check whether the sorted array *a* contains the element *x* anywhere between indices *start* and *end*, inclusive.

**11.** Write a recursive function ***IsPalindrome*** to check whether a string (char\*) is a palindrome or not.

**12.** Write a recursive function ***Sum*** that takes a two-dimensional array and returns sum of its all elements.