Assignment 4

1. Design a pseudo-code recursive method, findMax(L), that returns the maximum number in the list L.

   Algorithm findMax(arr)

   Hint: you also need a helper function with argument index i
   Algorithm findMaxHelper(arr, i)

   Algorithm findMax(arr)
   Input: array arr of n integers
   Output: maximum element in the array arr

   return findMaxHelper(arr, arr.length-1)

   Algorithm findMaxHelper (arr, i)
   Input: array arr of n integers and index i
   Output: maximum element in the array arr

   ```
   if i = 0 then
        return arr[i]
   else
   number = findMaxHelper(arr, i-1)
   return ((arr[i]>number) ? arr[i]:number);
   ```

Running time: O(n)

2. Write a pseudo code function, *sum*(n), to recursively sum the first n natural numbers but divide the problem in half and make two recursive calls.

```
Algorithm sum(int n) {
        return sumHelper(1,n);
    }

Algorithm sumHelper(int start, int end) {
   if (start > end) {
      return 0;
   } else {
      int mid = (start + end) / 2;
      int leftSum = sumHelper(start, mid - 1);
      int rightSum = sumHelper(mid + 1, end);
      return leftSum + rightSum + mid;
   }
}
```

Runtime Analysis: You are making two recursive calls. Each call is n/2. Time complexity is n/2 + n/2 = O(n)

3. Write a pseudo code function, *isEven*(n) to recursively determine whether a natural number, n, is an even number.

```
Algorithm isEven(n)
        if n = 0 then
                return true
        if n = 1 then
                return false
return isEven(n-2)
```

Runtime Analysis: With each recursive call, n is reduced by 2. It will make n/2 recursive calls. So time complexity is O(n).

4. Write a pseudo code function, power(x, k), that computes x^k. Can you do this in log k time?

Computes the value of x raised to the nth power, for nonnegative integer n.

```
Algorithm power(double x, int n) {
        if (n == 0)
            return 1;
        else {
                partial = power(x, n/2);
                double result = partial ∗ partial;
                if (n % 2 == 1)
                result ∗= x;
        return result;
          }
}
```

Running time is log n