**Assignment Three**

**Question 1**

If an algorithm has a best case running time that is equal to the worst case running time, it can be said the algorithm is tightly bound. That is, for a function the following is true.

From this, we can say that for our algorithm beautiful(A, n) to have a best case running time the same as the worst case running time, we find the average case running time for order n, i.e

**Algorithm** beautiful(A, n)

**Input** array A of n integers

**Output** the largest element of A

largest A[0]

**for** i 1 **to** n-1 **do**

**if** A[i] > largest **then**

largest A[i]

{increment counter i}

return largest

Algorithm beautiful(A, n) finds the largest element in a given integer array. No matter which index the largest element is found in the array, the for loop in the algorithm checks all the remaining elements to compare with the currently assigned largest element. This algorithm has a running time complexity of

**Question 2**

Solution - Ordering based on time complexity:

**Question 3**

1. : Accessing an array element by index. (eg. int a = arr[3])
2. : Binary Search
3. : Traversing an array
4. : Merge Sort
5. : Selection Sort
6. : An algorithm that uses the nested for loops to draw a pyramid pattern.
7. : Recursive solution to finding then nth Fibonacci number.

**Question 4**

The Master Theorem cannot be applied to the algorithm that finds the nth Fibonacci number because b is 1. The recursion decreases the problem statement by subtracting rather than by dividing into smaller problems.

**Question 5**

After k steps

Assume

Substitution

**Question 6**

…… 3rd case



………. 2nd case



……… 1st case