HOMEWORK 1

1. Prove the following statement:

$$f_1(n) \in O(g_1(n)) \wedge f_2(n) \in O(g_2(n)) \Rightarrow f_1(n) + f_2(n) \in O\left(\max\{g_1(n), g_2(n)\}\right)$$

2. Consider an improved version of Bubble sort algorithm:

```
ImprovedBubbleSort(a[1 .. n]) {
   flag = true;
   m = 1;
   while (flag) {
      flag = false;
      m++;
      for (j = n; j ≥ m; j--)
         if (a[j - 1] > a[j]) {
            a[j - 1] ⇆ a[j];
            flag = true;
         }
   }
}
```

   Analyze the complexity of this algorithm.

3. Consider the Insertion sort algorithm:

```
InsertionSort(a[1 .. n]) {
   for (i = 2; i ≤ n; i++) {
      v = a[i];
      j = i – 1;
      while (j ≥ 1) && (a[j] > v) {
         a[j + 1] = a[j];
         j--;
      }
      a[j + 1] = v;
   }
}
```

   Analyze the complexity of this algorithm.

4. Given an array of n integers. Design an algorithm with the time complexity $\Theta(n^2)$ to find a subsequence (or a contiguous subarray) with largest sum of numbers in the array. Note that the sum of the numbers of an empty subsequence is 0.