"I have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I understand that if I am involved in plagiarism or cheating I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of **0** for the involved assignment for my first offense and that I will receive a grade of **"F" for the course** for any additional offense."

Name: - Kushal S Shinde Sign: - Kushal S Shinde

1. Insertion Sort

Instruction count using Method 1.

Code snippet: -

```
while (i < number)
{
  j = i;
      while(j > 0 && array[j] < array[j-1])
      {
              swapping(&array[j],&array[j-1]);
              j--;
      }
      i++;
}
```

As per above code,

The outer loop executes from 1 to Number of input array elements times so N times.

The inner while loop will execute till value of i

$$
\begin{aligned}
T(n) &= T(n-1) + (n-1) \\
&= T(n-2) + (n-2) + (n-1) \\
&= T(n-3) + (n-3) + (n-2) + (n-1) \\
&\;\;\ldots\ldots \\
&= T(1) + 1 + 2 + 3 + \ldots + (n-1)
\end{aligned}
$$

So to calculate instruction count as per Summation formula,

$$
\begin{aligned}
\sum_{i=1}^{n} i &= n\,(n + 1) / 2 \\
&= (n^2 + n) / 2
\end{aligned}
$$

So this proves the insertion sort runs **$\Theta(n^2)$.**

2. Counting Sort

Instruction count using Method 2.

Code snippet: -

```
  while (i < number) {
    int second = array[i];
    second_array[second] = second_array[second] + 1;
    i++;
  }
   while (i < number) {
    int second = array[i];
    int location = second_array[second] - 1;
    result[location] = second;
    second_array[second] = second_array[second] - 1;
    i++;
  }
```

As per above code,

For this while loop will execute from 0 to input array number i.e. N-1 times.

$T(n) = 1$

$T(n) = T(n-1) + 1$

$\quad = T(n-2) + 1 + 1$

$\quad = T(n-3) + 1 + 1 + 1$

$\quad$ ……

$\quad = T(1) + 1+1+1+….+1$

$\quad = 1+n-1$

$\quad = n$

So to calculate instruction count as per Summation formula,

$\sum_{i=1}^{n} 1 = n$

So this proves the counting sort runs **Ɵ(n).**

3. Merge Sort

Instruction count using method 1.

Code snippet: -

```
  While ((n <= higher) && (x <= middle))
  {
    If (array[n] >= array[x]){
      other[j] = array[x];
```

```
            x++;
        }
        else{
            other[j] = array[n];
            n++;
        }
        j++;
    }

    if(x > middle){
        p=n;
        while(p<=higher){
            other[j] = array[p];
            j++;
            p++;
        }
    }
    else{
        p=x;
        while(p<=middle){
            other[j] = array[p];
            j++;
            p++;
        }
    }
```

$T(n) = 2T(n/2) + n$ For $n=2^k$

$= 4[T(n/4)+ (n/2)]+2n$

$= 8[T(n/8)] + 3n$

$= 2^k[T(n/2^k)] + kn$

$= 2^k[T(1)] + n\lg n$ $n=2^k$

$= 2^k(1) + n\lg n$

$= n + n\lg n$

So this proves the merge sort runs **ϴ(nlgn).**