Mathematical Analysis : Selection Sort

```
Void SelSort (int [], int n){
    int indx, Min;
    for (int Pos = 0; Pos < n-1; Pos++){
        Min = a[Pos]; indx = Pos;
        for (int i = Pos+1; i < n; i++){
            if(a[i] < Min){
                Min = a[i];
                indx = i;
            }
        }
        a[indx] = a[Pos];
        a[Pos] = Min;
    }
}
```

First loop worst case : $n-1$

Inner Worst case : from 0 to $n-2$

$n-i$ loops

The Sort inner loop runs $n-i$ times for $n-1$ times.

$$T(n) = \sum_{i=1}^{n-1}(n-i) = (n-1)+(n-2)+(n-3)\ldots+3+2+1$$
$$= 1+2+3\ldots+(n-3)+(n-2)+(n-1)$$

$$2\cdot T(n) = (1+(n-1))+(2+(n-2))\ldots+(2+(n-2))+(1+(n-1))$$

$$2\cdot T(n) = n+n+n\ldots = n(n-1)$$

$$T(n) = \frac{n(n-1)}{2} = \frac{n^2-n}{2}$$ largest term

$$O(n^2)$$ for Selection Sort

Mathematical Analysis : Bubble Sort

```
Void bublSrt(int a[], int n){
    bool swap;
    do{                                   Do while Worst Case: n-1
        swap = false;
        for(int i=0; i<n-1; i++){         for worst case: n-i
            if(a[i] > a[i+1]){
                int temp = a[i];
                a[i] = a[i+1];
                a[i+1] = temp;
                swap = true;
            }
        }
    }while(swap);
```

In worst case, do-while loops n-1 times and for each loop the for-loop loops n-i times.

Same as selection Sort

$(n-1) + (n-2) + (n-3) + \dots 1 = T(n) = \text{Operations } O$

$$T(n) = \sum_{i=1}^{n-1} i = 1 + 2 + 3 \dots n-1$$

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

$$2T(n) = n \cdot (n-1)$$

$$T(n) = \frac{n^2 - n}{2} \rightarrow O(n^2)$$