Selection sort

```
1. For(i←0 to n-1)//outer loop
2. Index=I
3. For(j←i+1 to n) //inner loop
4. If(arr[j]<arr[index])
5. Index=j
6. End for(inner)
7. If(arr[j]<arr[index] then arr[i] interchange by arr[index])
8. End for(outer loop)
```

Here, 0 to n-1 and 1 to n is time. For 1 time of outer loop inner loop execute for n time. So time complexity is n^2 .

In best case when array is already sorted then it will execute 1 time.

$B(1)=\Omega(n^2)$

In the worst case, it will execute for n time.

$W(n)=O(n^2)$

Code:

```
package selectionsort;
public class Selectionsort {
  void sort(int arr[])
  {
     int n=arr.length;
     for(int i=0;i<n-1;i++)
     {
       int index=i;
       for(int j=i+1;j<n;j++)
          if(arr[j]<arr[index])
             index=j;
       int temp=arr[index];
```

```
            arr[index]=arr[i];
            arr[i]=temp;
          }
      }

      void printArray(int arr[])
      {
          int n=arr.length;
          for(int i=0;i<n;i++)
          {
              System.out.print(arr[i]+"  ");
              System.out.println();
          }
      }

      public static void main(String[] args) {
          int[]arr={9,1,2,3,4,5,7,8,6,0};
          Selectionsort obj=new Selectionsort();
          obj.sort(arr);
          System.out.println("Sorted array");
          obj.printArray(arr);

      }

  }
```