**1) Sort a given set of *n* integer elements using Selection Sort method and compute its time complexity. Run the program for varied values of *n*> 5000 and record the time taken to sort. Plot a graph of the time taken versus *n*on graph sheet. The elements can be read from a file or can be generated using the random number generator. Demonstrate using C how the divide -and-conquer method works along with its time complexity analysis: worst case, average case and best case**

**#include <stdio.h>**

**int main()**

**{**

**int a[100], n, i, j, position, swap;**

**printf("Enter number of elements \n"); scanf("%d", &n);**

**printf("Enter %d Numbers", n); for (i = 0; i < n; i++) scanf("%d", &a[i]);**

**for(i = 0; i < n - 1; i++)**

**{**

**position=i;**

**for(j = i + 1; j < n; j++)**

**{**

**if(a[position] > a[j]) position=j;**

**}**

**if(position != i)**

**{**

**swap=a[i]; a[i]=a[position]; a[position=swap;**

**}**

**}**

**printf("Sorted Array:n");**

**for(i = 0; i < n; i++) printf("%dn", a[i]); return 0;**

**}**

**2)Sort a given set of *n* integer elements using Merge Sort method and compute its time complexity. Run the program for varied values of *n*> 5000, and record the time taken to sort. Plot a graph of the time taken versus *n*on graph sheet. The elements can be read from a file or can be generated using the random number generator. Demonstrate using C how the divide-and-conquer method works along with its time complexity analysis: worst case, average case and best case**

**#include<stdio.h>**

**#include<time.h>**

**#include<stdlib.h>**

**#define max 100000**

**void merge(int c[],int d[],int l,int m,int r)**

**{**

**int i=l, j=m+1,k=l; int q;**

**while((i<=m)&&(j<=r)) if(c[i]<=c[j])**

**d[k++]=c[i++];**

**else d[k++]=c[j++];**

**if(i>m)**

**for(q=j;q<=r;q++) d[k++]=c[q];**

**else for(q=i;q<=m;q++) d[k++]=c[q];**

**}**

**void mergesort(int a[],int left,int right)**

**{**

**int b[right], i; if(left<right)**

**{**

**int mid=(left+right)/2; mergesort(a,left,mid); mergesort(a,mid+1,right); merge(a,b,left,mid,right); for(i=left;i<=right;i++) a[i]=b[i];**

**}**

**}**

**void main()**

**{**

**int n,y[max],right,left; int i,pos;**

**long counter; float seconds;**

**clock\_t start,finish;**

**printf("\n\n\tMerge Sort \n\nEnter the no of elements to be sorted\n"); scanf("%d",&n);**

**printf("\n Enter %d elements", n); for(i=1;i<=n;i++)**

**{**

**scanf("%d",&y[i]);**

**}**

**start=clock(); counter=0; left=1; right=n;**

**while(clock()-start<1000000)**

**{**

**counter++; mergesort(y,left,right);**

**}**

**finish=clock();**

**seconds=(finish-start)/CLOCKS\_PER\_SEC; printf("\n The sorted elements:\n"); for(i=1;i<=n;i++)**

**printf("%d ", y[i]);**

**printf("\nTime complexity = %f", (float)seconds/counter);**

**}**