1. Sort a given set of N integer elements using Merge Sort technique and

compute its time taken. Run the program for different values of N and

record the time taken to sort.

#include <stdio.h>

#include <time.h>

int a[20],n;

void simple\_sort(int [],int,int,int);

void merge\_sort(int[],int,int);

int main()

{

int i;

clock\_t start, end;

double time\_taken;

printf("Enter the no. of elements:");

scanf("%d", &n);

printf("Enter the array elements:");

for (i = 0; i <n; i++) {

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

}

start = clock();

merge\_sort(a, 0, n - 1);

end = clock();

time\_taken = (double)(end - start) / CLOCKS\_PER\_SEC;

printf("Sorted array:");

for (i = 0; i <n; i++) {

printf("%d\t", a[i]);

}

printf("\n");

printf("Time taken to sort: %f seconds\n", time\_taken);

return 0;

}

void merge\_sort(int a[],int low, int high){

if(low<high){

int mid=(low+high)/2;

merge\_sort(a,low,mid);

merge\_sort(a,mid+1,high);

simple\_sort(a,low,mid,high);

}

}

void simple\_sort(int a[],int low, int mid, int high){

int i=low,j=mid+1,k=low;

int c[n];

while(i<=mid && j<=high){

if(a[i]<a[j]){

c[k++]=a[i];

i++;

}else{

c[k++]=a[j];

j++;

}

}

while(i<=mid){

c[k++]=a[i];

i++;

}

while(j<=high){

c[k++]=a[j];

j++;

}

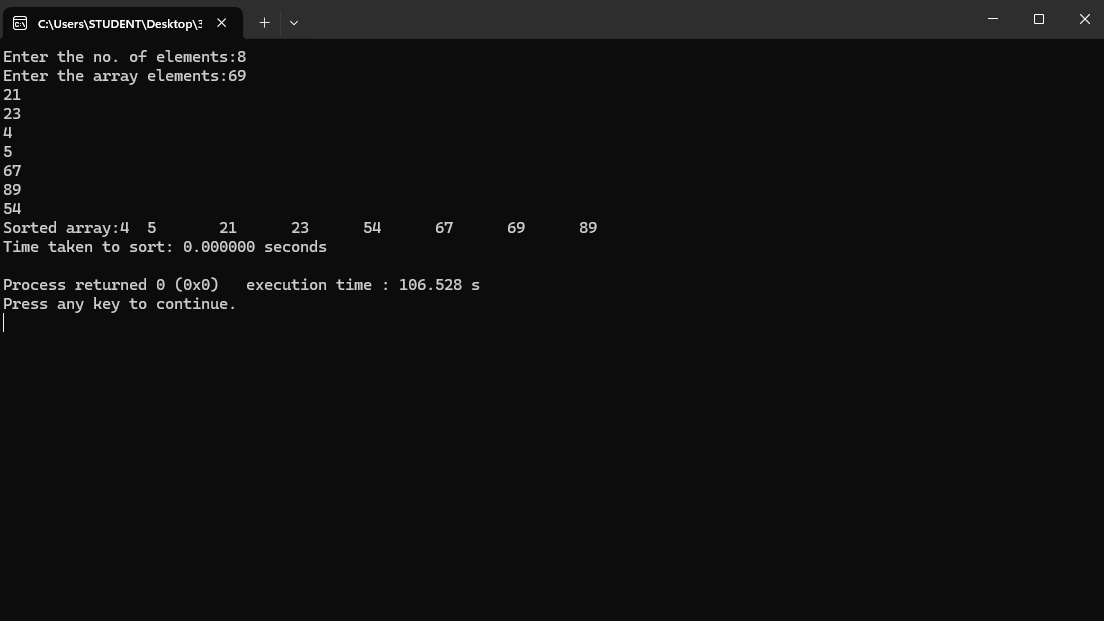
for(i=low;i<=high;i++){

a[i]=c[i];

}

}­­­­

Output:



Graph: