Ankit Raj 1906534

Q1. write a program to recursively implement Binary Search using divide and conquer method. Determine the time required to search an element in an array of n integers. Repeat the experiment for different values of n, the number of elements in the list to be searched and plot a graph of the time taken versus n. The n integers can be generated randomly.

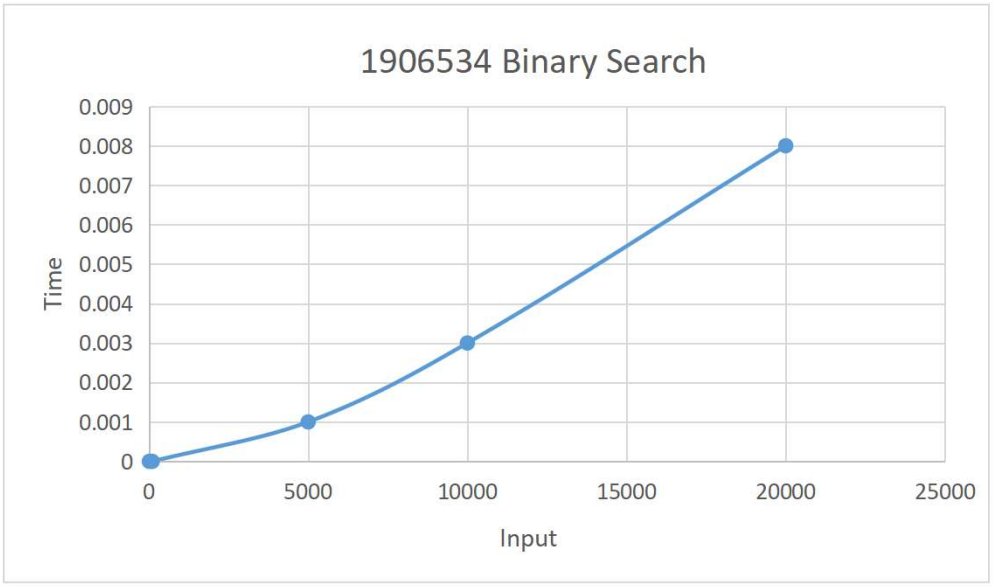
```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
void binarysearch(int arr[], int n  int key {)
int low = 0;
int high = n-1;
int found = 0;
while(low <= high){
int mid = (low+high)/2;
if (key==arr[mid]){
 found = 1;
 printf("%d is present at position  %d\n ", key, mid+1);
 break;
          }
else if (key< arr[mid])
high = mid-1;
else
low = mid+1;
      }
```

```
if (low>high && found ==0)
 printf("\nElemnt does not exist\n");
}

void insertionsort(int arr[], int n){
int j;
for (int i = 1; i < n; i++)
{
 int temp = arr[i];
j = i-1;
 while (arr[j] > temp && j>=0   )
{
 arr[j+1] = arr[j];
 j--;
          }
 arr[j+1] = temp;
 }
}

void display(int arr[], int n){

for (int i = 0; i < n; i++)
printf("%d\t", arr[i]);

}
```

```
int main(){
int n, key;
clock_t start, end;
double totalCPUtime;

printf("Enter the number of elemnts in the array  ");
scanf("%d", &n);
int arr[n];

for (int i =0;i arr[i] = (rand() % 1000);
printf("%d\t", arr[i]);
      }
printf("\nsorted array  .....\n");
(arr, n);
display(arr, n);
printf("\nEnter the number that has to be searched  ");
scanf("%d", &key);

start = clock();
binarysearch(arr,n, key);
end = clock();

totalCPUtime = ((double)(end - start));
printf("\ntotal CPU time in ms: %f", totalCPUtime);
totalCPUtime = ((double)(end - start)/CLOCKS_PER_SEC);
```

```
printf("\ntotal CPU time in s  %f", totalCPUtime);
}
```

INPUT	TIME
10	0
100	0
5000	0.001
10000	0.003
20000	0.008



Q2.write a program to sort a given set of elements using the Merge sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The t elements can be read from a file or can be generated using the random number generator.

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
void merge(int arr[], int low, int mid, int high){
int n1 = mid - low + 1;
int n2 = high - mid;
int L[n1], R[n2];
 for (int i = 0; i < n1; i++)
L[i] = arr[low+i];
 for (int j = 0; j < n2; j++)
R[j] = arr[mid+j+1];

int i= 0;
int j = 0;
int k = low;
while (i if(L[i] <= R[j]){
 arr[k] = L[i];
 i++;
}
else{
```

```
 arr[k] = R[j];
 j++;
}
k++;
}

while (i< n1){
 arr[k] = L[i];
 i++;
 k++;
}

while(j < n2){
  arr[k] = R[j];
  j++;
  k++;
 }
}

void mergesort(int arr[], int low, int high){
if (low < high){
int mid = low + (high-low)/2;
mergesort(arr, low, mid);
mergesort(arr, mid+1, high);
```

```
 merge(arr, low, mid, high);
 }
}

void display(int arr[], int n){
for (int i = 0; i < n; i++)
{
 printf("%d\t", arr[i]);
 }
}

int main(){
int n;
clock_t start, end  ;
double totalCPUtime;
printf("Enter the number of elemnts in the array  ");
scanf("%d", &n);
int arr[n];

for (int i =0;i arr[i] = (rand() % 101);
 printf("%d\t", arr[i]);
}
int low = 0;
int high = n-1;
```

```
printf("\nsorted array ..........\n");
start = clock();
mergesort(arr, low, high);
end = clock();
display(arr, n);
printf("\n\n");

totalCPUtime = ((double)(end - start));
printf("\ntotal CPU time in ms: %f", totalCPUtime);
totalCPUtime = ((double)(end - start)/CLOCKS_PER_SEC);
printf("\ntotal CPU time in s  %f", totalCPUtime);

return 0;
}
```

INPUT	TIME
10	0
100	0.001
5000	0.004
10000	0.007
20000	0.018