```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

int status;

int linearsearch(int arr[], int n, int key, int i)
{
    status = -1;
    if(arr[i]==key)
    {
        status=0;
        return i;
    }
    else
    {
        return linearsearch(arr,n,key,(i+1));
    }

}

void sortArray(int arr[], int n)
{
    for(int b=0; b<n; b++)
    {
        for(int e=b+1; e<n; e++)
        {
            if(arr[b]>arr[e])
            {
                arr[b]=arr[b]+arr[e];
                arr[e]=arr[b]-arr[e];
                arr[b]=arr[b]-arr[e];
            }
        }
    }
}
```


swaroop

```
int binarysearch(int arr[], int b, int e, int key)
{
    status = 1;
    if (b<=e)
    {
      int m=b+(e-b)/2;
      if (arr(m)==key)
      {
          status=0;
          return m;
      }
      else if (arr[m]>key)
          return binarysearch(arr,b,m-1,key);
      else
          return binarysearch(arr, m+1,e,key);
    }
}

int main()
{
    srand(time(NULL));
    int i,k;
    printf("Enter length of random array: ");
    scanf("%d", &i);
    int arr[i];
    printf("Enter the element to find ");
    scanf("%d", &k);
    key = k;
    for(i=0; i<i; i++)
    {   arr[i]= (rand()%1000);
    }
```


Swaroop

```
double res1, res2;
clock_t start, end;

start = clock();
int i = linearsearch(arr, l, key, 0);
end = clock();
res1 = ((double)(end-start)) / CLOCKS_PER_SEC;

start = clock();
int bi = binarysearch(arr, 0, l-1, key);
end = clock();
res2 = ((double)(end - start)) / CLOCKS_PER_SEC;

printf("\n - LINEAR SEARCH - \n");
if (status==0)
{
  printf("\n%d found at index : %d\n", key, i);
}
else
  printf("%d not found in the array");

printf("Time taken by linear search : %.1f\n", res1);
```


Swaroop

```
printf("\n--BINARY SEARCH --");
if (status==0)
 {
   printf("\n%d found at index %d\n", key, bi);
 }
 else
   printf("\n%d not found in array\n", key);

 printf("\nTime taken by binary search : %lf\n", usb);
}
```

```
int Mode (int arr[], int nn, int k)
{
    int i = binarySearch (arr, 0, nn-1, k);
    if (i == -1)
        return 0;
    int count = 1;
    int left = i-1;
    while (left >= 0 && arr[left] == k)
        count++, left--;
    int right = i+1;
    while (right < nn && arr[right] == k)
        count++, right++;
    return count;
}
```


Swara