

PROGRAM1 : RECURSIVE Linear Search and Binary Search programs

[1:41 pm, 01/06/2021] Muskan Gutpa: #include <stdio.h>

#include <stdlib.h>

#include <time.h>

clock_t start, end;

double cpu_time;

int linear_search(int arr[], int high, int low, int key)

{

if (low<high)

return -1;

if (arr[high] == key)

return high;

if (arr[low] == key)

return low;

return linear_search(arr,high+1,low-1,key);

}

int binary_search(int arr[],int high, int low, int key)

{

if (low>=high)

{

int mid = (high+low)/2;

if (arr[mid]==key)

{

return mid;

}

if (arr[mid]>key)

{

return binary_search(arr,high,mid-1,key);

}

return binary_search(arr, mid + 1, low, key);

}

```

    return -1;
}
int main()
{

    int k,pos,c,d,i,n,temp,choice,key,j,flag=1,arr[10000];
    srand(time(0));
    while (flag==1)
    {
        printf("1:Linear_Search\n2:Binary_Search\n3:Exit\n");
        printf("Enter your choice\n");
        scanf("%d", &choice);
        switch(choice)
        {
            case 1:
                printf("Enter the number of elements:\n");
                scanf("%d", &n);
                for (k = 1; k <= n; k++)
                {
                    arr[k]=rand()%100;
                    printf("%d ",arr[k]);
                }
                printf("\nEnter the Element to be Searched : \n");
                scanf("%d", &key);
                start = clock();
                pos = linear_search(arr, 0, n-1, key);
                for (c = 1; c <= 5000; c++) for (d = 1; d <= 5000; d++) { }
                end = clock();
                cpu_time = (double)(end - start) / CLOCKS_PER_SEC;
                if(pos == -1)
                {

```

```

printf("Element is not present in the Array\n");
}
else
{
printf("Element is present at the Position %d\n", pos);
}
printf("Execution time for linear_search = %f ms\n", cpu_time*1000);
break;
case 2:
printf("Enter the number of elements:");
scanf("%d", &n);
for (int k=1; k<=n; k++)
{
arr[k]=rand()%100;
}
for (i=1; i <=n;i++)
{
for (j = i + 1; j <= n; ++j)
{
if (arr[i] >arr[j])
{
temp =arr[i];
arr[i] = arr[j];
arr[j] = temp;
}
}
}
for (int k =1; k <=n; k++)
{
printf("%d ",arr[k]);
}

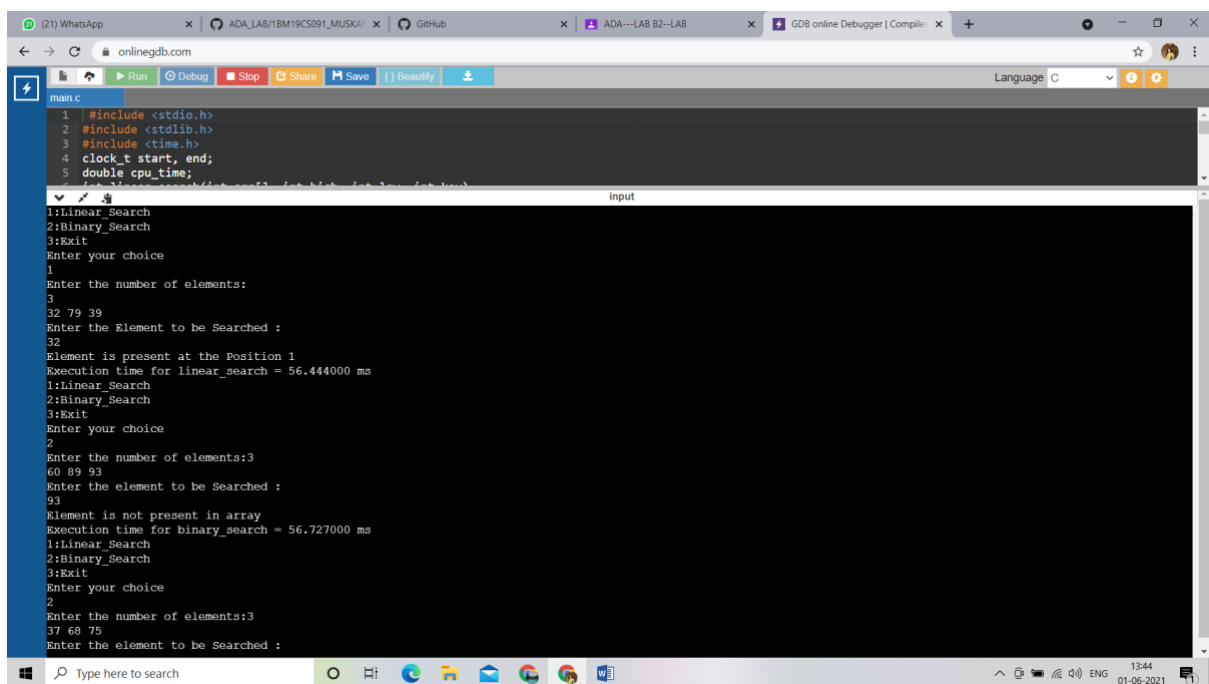
```

```

printf("\nEnter the element to be Searched :\n");
scanf("%d", &key);
start = clock();
for (c = 1; c <= 5000; c++) for (d = 1; d <= 5000; d++) { }
pos = binary_search(arr, 0, n - 1, key);
end = clock();
cpu_time = (double)(end - start) / CLOCKS_PER_SEC;
if(pos == -1)
{
    printf("Element is not present in array\n");
}
else
{
    printf("Element is present at the Position %d\n", pos);
}
printf("Execution time for binary_search = %f ms\n", cpu_time*1000);
break;
default:flag=0;
}
}
return 0;
}

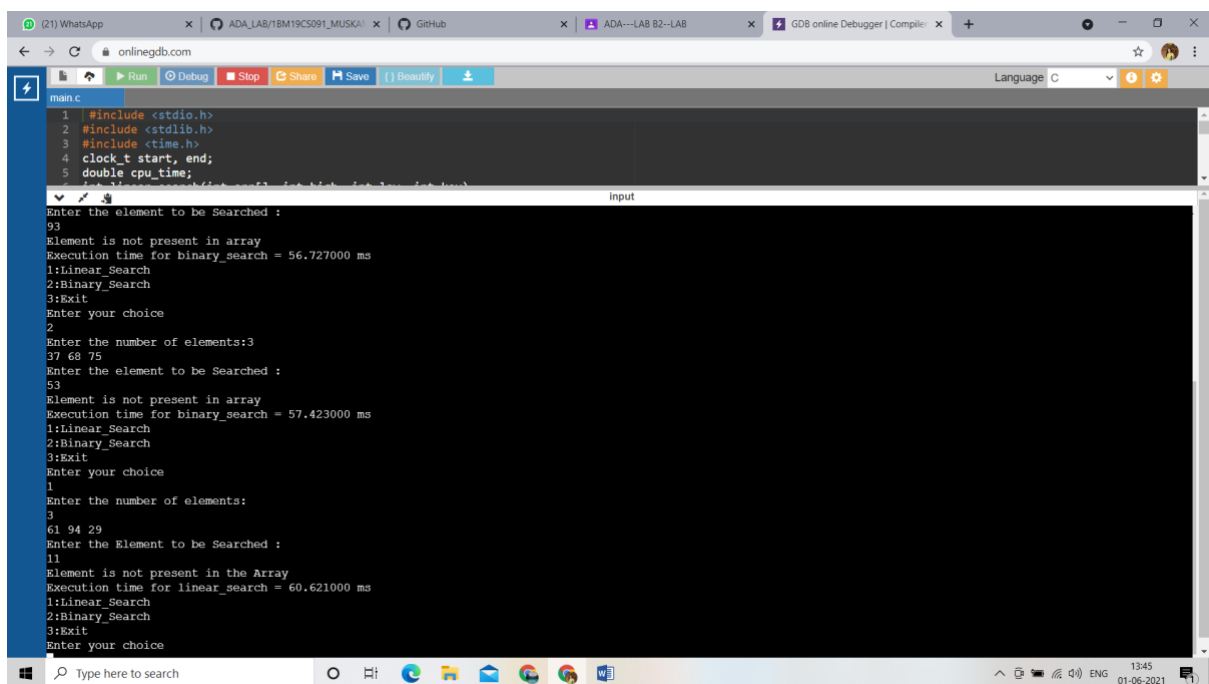
```

OUTPUT:



```
main.c
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 clock_t start, end;
5 double cpu_time;

1:Linear_Search
2:Binary_Search
3:Exit
Enter your choice
1
Enter the number of elements:
3
32 79 39
Enter the Element to be Searched :
32
Element is present at the Position 1
Execution time for linear_search = 56.444000 ms
1:Linear_Search
2:Binary_Search
3:Exit
Enter your choice
2
Enter the number of elements:3
60 89 93
Enter the element to be Searched :
93
Element is not present in array
Execution time for binary_search = 56.727000 ms
1:Linear_Search
2:Binary_Search
3:Exit
Enter your choice
2
Enter the number of elements:3
37 68 75
Enter the element to be Searched :
```



```
main.c
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 clock_t start, end;
5 double cpu_time;

Enter the element to be Searched :
93
Element is not present in array
Execution time for binary_search = 56.727000 ms
1:Linear_Search
2:Binary_Search
3:Exit
Enter your choice
2
Enter the number of elements:3
37 68 75
Enter the element to be Searched :
53
Element is not present in array
Execution time for binary_search = 57.423000 ms
1:Linear_Search
2:Binary_Search
3:Exit
Enter your choice
1
Enter the number of elements:
3
61 94 29
Enter the Element to be Searched :
11
Element is not present in the Array
Execution time for linear_search = 60.621000 ms
1:Linear_Search
2:Binary_Search
3:Exit
Enter your choice
```