EX NO: 3 **NAME : P S NAVINA SHRI**

DATE: 12.06.2021 **REG NO : 140920104094**

LINEAR AND BINARY SEARCH

3.1Linear Search

AIM :

To write a python program to print the position of any element in a list using Linear Search algorithm.

ALGORITHM:

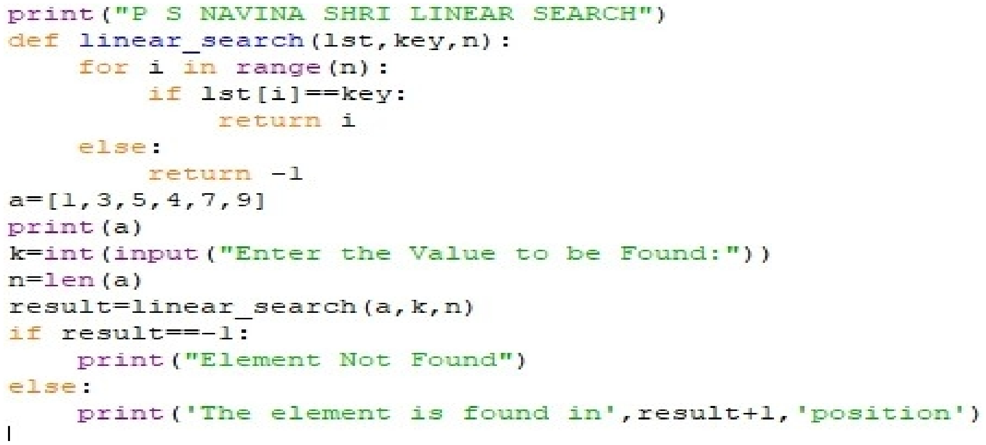
```
STEP 1: START
STEP 2: Get the list from user
STEP 3: Get the required element from the user
STEP 4: Start from the leftmost element of given arr[] and one by one compare element x with each
        element of arr[]
STEP 5: If x matches with any of the element, return the index value.
STEP 6: If x doesn’t match with any of elements in arr[] , return -1 or element not found
STEP 7: STOP
```

PROGRAM:

```
print("P S NAVINA SHRI LINEAR SEARCH")
def linear_search(lst,key,n):
   for i in range(n):
      if lst[i]==key:
         return i
   else:
      return -1
a=[1,3,5,4,7,9]
print(a)
```

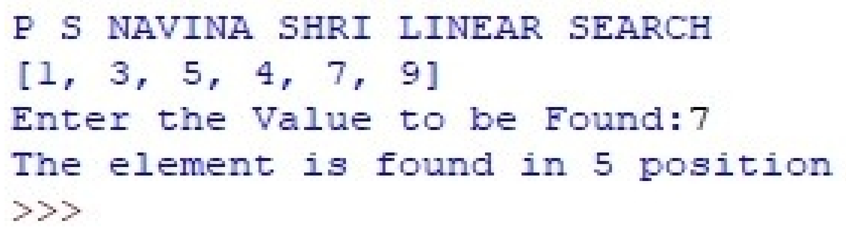
```
k=int(input("Enter the Value to be Found:"))

n=len(a)

result=linear_search(a,k,n)

if result==-1:

    print("Element Not Found")

else:

    print('The element is found in',result+1,'position')
```

EXECUTED PROGRAM SCREENSHOT:



```
print("P S NAVINA SHRI LINEAR SEARCH")
def linear_search(lst,key,n):
    for i in range(n):
        if lst[i]==key:
            return i
    else:
        return -1
a=[1,3,5,4,7,9]
print(a)
k=int(input("Enter the Value to be Found:"))
n=len(a)
result=linear_search(a,k,n)
if result==-1:
    print("Element Not Found")
else:
    print('The element is found in',result+1,'position')
```

OUTPUT SCREENSHOT:



```
P S NAVINA SHRI LINEAR SEARCH
[1, 3, 5, 4, 7, 9]
Enter the Value to be Found:7
The element is found in 5 position
>>>
```

RESULT:

Thus a python program to print the position of any element in a list using Linear Search algorithm is executed successfully and its output is verified.

3.2Binary Search

AIM :

To write a python program to print the position of any element in a list using Linear Search algorithm.

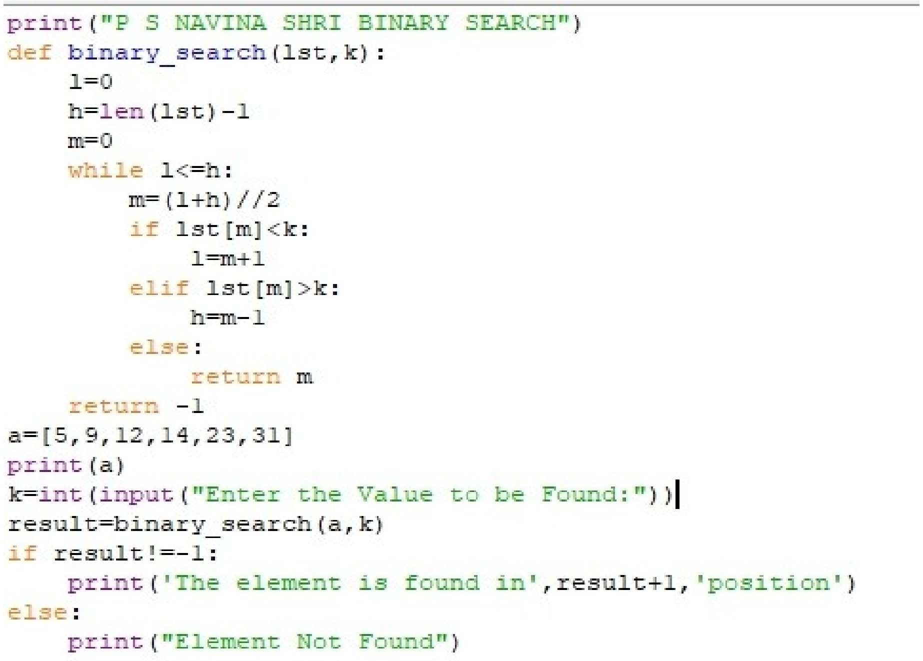
ALGORITHM:

```
Step 1: START
Step 2: Compare x with the middle element.
 Step 3: If x matches with the middle element, we return the mid index.
Step 4: Else If x is greater than the mid element, then x can only lie in the right half subarray
after the mid element. So we recur for the right half.
Step 5: Else (x is smaller) recur for the left half.
Step 6:  STOP
```

PROGRAM:

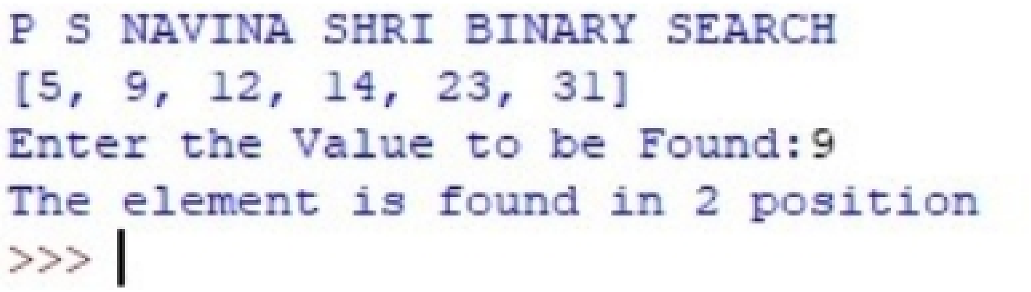
```
print("P S NAVINA SHRI BINARY SEARCH")
def binary_search(lst,k):
   l=0
   h=len(lst)-1
   m=0
   while l<=h:
      m=(l+h)//2
      if lst[m]<k:
         l=m+1
      elif lst[m]>k:
            h=m-1
        else:
            return m
    return -1
a=[5,9,12,14,23,31]
print(a)
k=int(input("Enter the Value to be Found:"))
result=binary_search(a,k)
if result!=-1:
    print('The element is found in',result+1,'position')
else:
    print("Element Not Found")
```

EXECUTED PROGRAM SCREENSHOT:



```
print("P S NAVINA SHRI BINARY SEARCH")
def binary_search(lst,k):
    l=0
    h=len(lst)-1
    m=0
    while l<=h:
        m=(l+h)//2
        if lst[m]<k:
            l=m+1
        elif lst[m]>k:
            h=m-1
        else:
            return m
    return -1
a=[5,9,12,14,23,31]
print(a)
k=int(input("Enter the Value to be Found:"))
result=binary_search(a,k)
if result!=-1:
    print('The element is found in',result+1,'position')
else:
    print("Element Not Found")
```

OUTPUT SCREENSHOT:



```
P S NAVINA SHRI BINARY SEARCH
[5, 9, 12, 14, 23, 31]
Enter the Value to be Found:9
The element is found in 2 position
>>> 
```

RESULT:

Thus a python program to print the position of any element in a list using Binary Search algorithm is executed successfully and its output is verified.