**Aim:** Program to implement Linear search.

(1). Using array

**Source code:**

#include<iostream.h>

#include<conio.h>

#include<stdio.h>

void main()

{

clrscr();

int item,i,c=0,size,a[20];

cout<<"Enter the size of an array:";

cin>>size;

cout<<"Enter the elements";

for(i=0;i<size;i++)

{

cin>>a[i];

}

cout<<"Enter the element to search:";

cin>>item;

for(i=0;i<size;i++)

{

if(a[i]==item)

{

c++;

}

}

cout<<"Item Found" <<" "<<c<<" times in the list";

if (c==0)

{

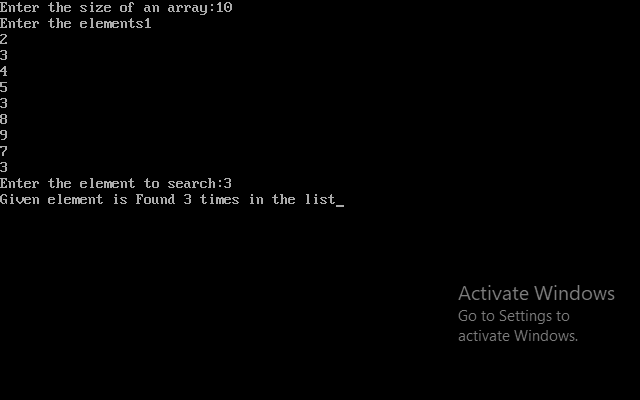
cout<<"Item not found in the list";

}

getch();

}

**Output:**



(2). Using linked list

**Source code:**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#include<alloc.h>

struct NODE{

int element;

struct NODE \*link;

}; NODE \*Head;

void ins(){

int p;

NODE \*n,\*pntr;

pntr=Head;

printf("\n enter the element to insert");

scanf("%d",&p);

n=(NODE\*)malloc(sizeof(struct NODE));

n->element=p;

if(Head==NULL)

{

n->link=NULL;

Head=n;

}

else

{

while(pntr->link!=NULL)

{

pntr=pntr->link;

}

pntr->link=n;

n->link=NULL;

}

}

void trav(){

struct NODE \*pntr;

pntr=Head;

if(Head==NULL)

{

printf("\n The given list is empty");

getch();

}

else{

while(pntr!=NULL) {

printf("%d->\t",pntr->element);

pntr=pntr->link;

}

getch();

}

}

void search()

{

int count=0,key\_value,i=0;

struct NODE\*pntr;

pntr=Head;

printf("\n enter number to be search");

scanf("%d",&key\_value);

while(pntr!=NULL||count==1){

if(pntr->element==key\_value)

{

count=1;

break;

}

else{

i++;

pntr=pntr->link;

}

}

if(count==1)

{

printf("\n %d is found at %d",key\_value,i+1);

}

else{

printf("\n The no. is not present");

}

getch();

}

void main(){

int ch;

A: clrscr();

printf("1. Insert the element\n");

printf("2. Traverse the element\n");

printf("3. Search the element\n");

printf("4. Exit\n");

printf("Enter your choice");

scanf("%d",&ch);

switch(ch){

case 1: ins();

goto A;

case 2: trav();

goto A;

case 3: search();

goto A;

case 4: exit(0);

default:

printf("Choice entered is invalid.");

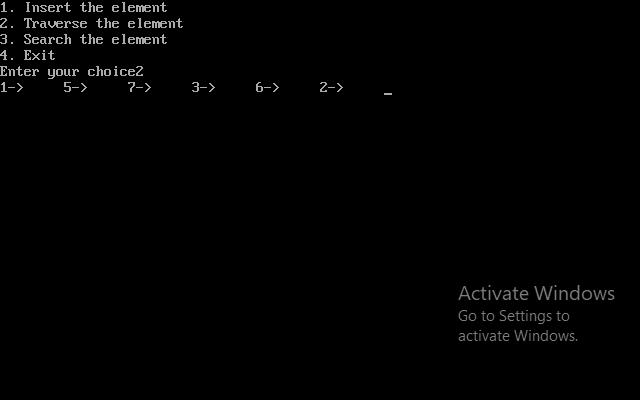
getch();

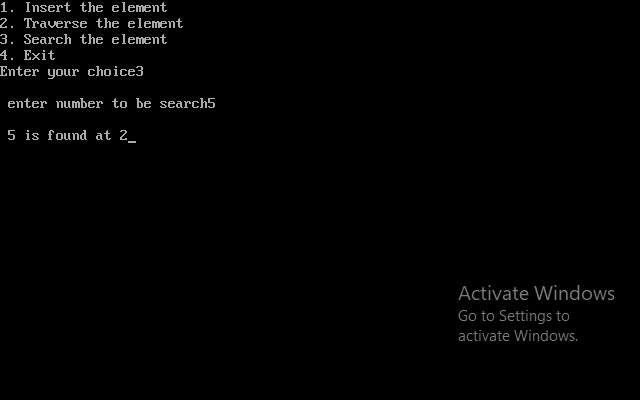
goto A;

}

}

**Output:**

****

****

**Analysis:**

1. Best case complexity = O(1)

2. Worst case complexity = O(n)

3. Average case complexity = O(n)