**Aim:** Program to implement binary search

(1). Using array

**Source code:**

#include<stdio.h>

#include<conio.h>

void main()

{ int a[20],i,flag=0,l=1,u,size,mid,key;

clrscr();

printf("\n Enter the size of array in positive integer ");

scanf("%d",&size);

u=size;

printf("\n Enter the integer elements of array");

for(i=1;i<=size;i++)

{ scanf("%d",&a[i]);

}

printf("\ Enter the key element to search");

scanf("%d",&key);

while((flag==0) && (l<u))

{ mid=(l+u)/2;

if(key==a[mid])

{printf("\n sucess");

printf("\n element is found at %d",mid);

flag=1;

}

else

{ if(key<a[mid])

{u=mid-1;

}

else

{ l=mid+1;

}

}

}

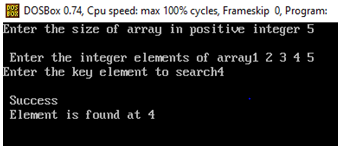
if(flag==0)

printf("\n unsuccessfull ");

getch();

}

**Output:**



(2). Using linked list

**Source code:**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

struct node{

int data;

struct node \*link;

};

struct node \*head,\*n;

void insert(){

int x;

struct node \*ptr;

ptr=head;

printf("Enter the value of new node in ascending order: ");

scanf("%d",&x);

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

n->data=x;

if(head==NULL){

n->link=NULL;

head=n;

}else{

while(ptr->link!=NULL){

ptr=ptr->link;

}

n->link=NULL;

ptr->link=n;

}

}

void traversal(){

struct node \*ptr;

ptr=head;

if(head==NULL){

printf("List is empty");

getch();

}

else{

while(ptr!=NULL){

printf("%d-> ",ptr->data);

ptr=ptr->link;

}

getch();

}

}

void srch(){

struct node \*ptr;

ptr=head;

int m,l=1,u=0,t,c=0,key;

if(head==NULL){

printf("List is empty");

}

else{

printf("Enter key value: ");

scanf("%d",&key);

while(ptr!=NULL){

ptr=ptr->link;

u++;

}

do{

t=0;

ptr=head;

m=(l+u)/2;

while(t!=m-1){

ptr=ptr->link;

t++;

}

if(ptr->data==key){

printf("%d found at position %d",key,m);

c=1;

}

else if(key>ptr->data){

l=m+1;

}

else{

u=m-1;

}

}while(l<=u&&c==0);

if(c==0){

printf("Value not found");

}

}

getch();

}

void main(){

int c;

L:system("cls");

printf("1. Insertion\n");

printf("2. Traversal\n");

printf("3. Search\n");

printf("4.Exit\n");

printf("Enter your choice: ");

scanf("%d",&c);

switch(c){

case 1:

insert();

goto L;

case 2:

traversal();

goto L;

case 3:

srch();

goto L;

case 4:

exit(0);

default:

printf("Invalid choice...Enter your choice again");

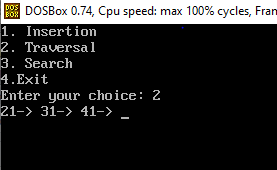
getch();

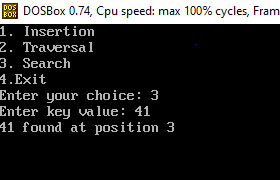
goto L;

}

}

**Output:**





**Analysis:**

1. Best case complexity = O(1)

2. Worst case complexity = O(log n)

3. Average case complexity = O(log n)