Binary Search using Recursion

Saurabh | Apr 10, 2010 | Comments 3

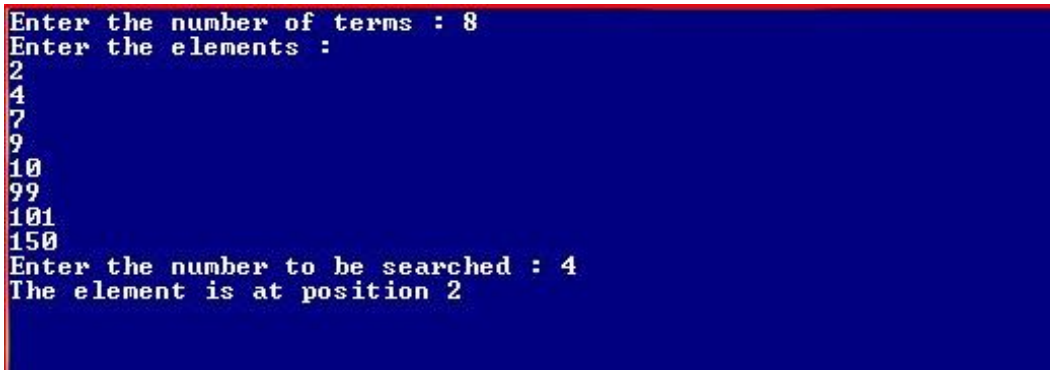
As a simple rule of recursion, any function can be computed using a recursive routine if :
1. The function can be expressed in its own form.
2. There exists a termination step, the point at which f(x) is known for a particular 'x'.

Therefore to write a recursive program for the binary search, we have to express the binary search function in a recursive form using the above 2 rules :

1. if(n < a[mid])
{ high = mid – 1;
binarysearch(a,n,low,high);
}
if(n > a[mid])
{ low = mid + 1;
binarysearch(a,n,low,high);
}

where 'n' is the element to be searched and low and high the lower and upper bounds of the array

2. if(n == a[mid]) print mid. (termination step)



The exe file can be downloaded from here : Recursive Binary Search

Using these 2 rules, the recursive program for binary search can be coded very easily as shown :

```
#include "stdio.h"
binarysearch(int a[],int n,int low,int high)
        { int mid;
```

```
          if (low > high)
           return -1;
          mid = (low + high)/2;
          if(n == a[mid])
            { printf("The element is at position %d\n",mid+1);
              return 0;
            }
          if(n < a[mid])
            { high = mid - 1;
              binarysearch(a,n,low,high);
            }
          if(n > a[mid])
            { low = mid + 1;
              binarysearch(a,n,low,high);
            }
         }

main()
        { int a[50];
          int n,no,x,result;
          printf("Enter the number of terms : ");
          scanf("%d",&no);
          printf("Enter the elements :\n");
          for(x=0;x&ltno;x++)
           scanf("%d",&a[x]);
          printf("Enter the number to be searched : ");
          scanf("%d",&n);
          result = binarysearch(a,n,0,no-1);
          if(result == -1)
           printf("Element not found");
          return 0;
        }
```

Related Posts

- C code for Radix Tree (or Trie)
- C Code for Binary Search Tree Traversal
- Sorting Objects Using Comparable in java
- Java code for Permutation using Steinhaus–Johnson–Trotter algorithm
- Java Code for generating Unique Random Numbers

Filed Under: C • Codes

Tags: recursion

About the Author: Software Engineer - Advanced Search & Recommendation at Rovi

Comments (3)

Leave a Reply | Trackback URL

1.  *eulogy uncle* says:
   May 3, 2010 at 2:04 pm

   Hey, Just what I was browsing for! I was researching articles and reviews for our online site when I came across your article on ” Binary Search using Recursion | programminggeeks.com ” which I found on AOL. We would love you to write for us, if interested. I’ve bookmarked this post for future reference. Good comments here as well -

   Reply
2.  *Ganesh Kamath* says:

October 3, 2011 at 11:48 pm

/*I made slight corrections so that the the code can be executed directly, very well written (^_^)*/

```
#include
void binarysearch(int a[],int n,int low,int high)
{
int mid;
if (low > high)
return -1;
mid = (low + high)/2;
if(n == a[mid])
{
printf(“The element is at position %d\n”,mid+1);
return 0;
}
if(n a[mid])
low = mid + 1;
binarysearch(a,n,low,high);
}

void main()
{
int a[50];
int n,no,x,result;
printf(“Enter the number of terms : “);
scanf(“%d”,&no);
printf(“Enter the elements :\n”);
for(x=0;x < no;x++)
scanf("%d",&a[x]);
printf("Enter the number to be searched : ");
scanf("%d",&n);
result = binarysearch(a,n,0,no-1);
if(result == -1)
printf("Element not found");
getch();
}
```

Reply

3.  *Mahesh vanol* says:
November 20, 2011 at 2:26 am

Nice work…Man

Reply

Leave a Reply

Name (required)

Mail (will not be published) (required)

Website

Submit Comment

CAPTCHA Code *

- Subscribe
- Archives
- Tags
- Popular

RSS Feed | Comments Feed

Get the latest updates via email.

enter email address Subscribe

Privacy guaranteed. We will not share your information.

- Search
- Login

Username

Password

Remember Me

Log In

Register | Forgot Password?

- Tags

Algorithms Android Articles C Codes Data Structures Featured Games Java Open Source TechNews Tricks Uncategorized

- Sponsors