**Finger Search Tree**

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

int binsearch(int sr\_arr[],int l,int r,int x);

int main()

{

int arr[50];

int sr\_arr[50];

int len\_arr=35;

int finger;

int ex;

int dat;

int c1=0;

int c=0;

int c2=0;

int flag=0;

int flag\_val=0;

int prv\_c2=0;

FILE \*in\_file  = fopen("code\_in.txt", "r"); // read only

    FILE \*out\_file = fopen("code\_out.txt", "w"); // write only

if (in\_file == NULL || out\_file == NULL)

            {

              printf("Error! Could not open file\n");

              exit(-1); // must include stdlib.h

            }

for(int um=0;um<=35;um++)

{

fscanf(in\_file, "%d", &arr[um]);

   }

printf("Enter finger number:");

scanf("%d",&finger);

printf("\nEnter desired array data");

scanf("%d",&dat);

c=finger-1;

for(;c1<=len\_arr;c1++)

{

sr\_arr[c1]=arr[c];

c++;

}

int lim=c1-finger;

for(int u=0;u<lim;u++)

{

if(sr\_arr[c2]==dat)

{

printf("Data found\n Data is: %d \nIndex is: %d \nFinger is: %d",sr\_arr[c2],c2,finger);

fprintf(out\_file,"Data found\n Data is: %d \nIndex is: %d \nFinger is: %d",sr\_arr[c2],c2,finger);

u=lim;

}

else if(sr\_arr[c2]!=dat)

{

if(sr\_arr[c2]<dat)

{

prv\_c2=c2;

flag=0;

}

else if(sr\_arr[c2]>dat)

{

flag=1;

}

if(flag==1)

{

printf("Data is in between %d and %d",sr\_arr[prv\_c2],sr\_arr[c2]);

fprintf(out\_file,"Data is in between %d and %d",sr\_arr[prv\_c2],sr\_arr[c2]);

int result = binsearch(sr\_arr,prv\_c2,c2,dat);

if(result==-1)

{

printf("Element is not present in array");

fprintf(out\_file,"Element is not present in array");

}

else

{

printf("\nData is: %d at array index: %d with finger at: %d",sr\_arr[result],result,finger);

fprintf(out\_file,"\nData is: %d at array index: %d with finger at: %d",sr\_arr[result],result,finger);

}

u=lim;

}

else if(flag==0)

{

ex=pow(2,u);

c2=ex-1;

}

}

}

}

int binsearch(int sr\_arr[], int l, int r, int x)

{

if(r>=l)

{

int mid=l+(r-l)/2;

if(sr\_arr[mid]==x)

return mid;

if(sr\_arr[mid]>x)

return binsearch(sr\_arr,l,mid-1,x);

else

return binsearch(sr\_arr,mid+1,r,x);

}

return -1;

}