Topic

Page
Date 28/12/2020

Assignment :4

Q.1) Write algorithm / function for ternary search

→ Algorithm:

ternarysearch (array, start, end, key)

Begin
if start <= end then
midFirst := start + (end - start) /3
midsecond := midFirst + (end - start)/3
if array [midFirst] = key then
return midfirst
if array [midsecond] = key then
return midsecond
if key < array [midFirst] then
call ternarysearch (array, start, midFirst - 1,
key)
if key > array [midsecond] then
call ternarysearch (array, midFirst + 1, end,
key)
else
call ternarysearch (array, midFirst + 1,
midsecond - 1, key)
else
return invalid location
End

Pioneer™

Q.2) In binary search algorithm, suggested to calculate the mid as beg + (end - beg)/2 instead of (beg+end)/2
Why it is so?

→ The very first finding middle index is:

mid = (beg + end)/2

But there is problem with approach, what if value of beg or end or both is INT_MAX, it will cause integer overflow.
better way to calculate mid index is:

mid = beg + (end - beg)/2

Let's try both methods in c program:

```
#include<stdio.h>
#include<limits.h>
int main()
{
int beg = INT_MAX, end = INT_MAX;
printf("beg = %d \n", beg);
printf("end = %d \n", end);
//method 1
int mid1 = (beg + end)/2;
printf("mid using (beg+end)/2 = %d \n", mid1);
//method 2
int mid2 = beg + (end - beg)/2;
printf("mid using beg+(end-beg)/2 = %d \n", mid2);
return 0;
}
```

→ op: beg = 2147483647
end = 2147483647
mid using (beg + end)/2 = -1
mid using beg + (end - beg)/2 = 2147483647