BINARY TREES USING ARRAYS

AIM

write a menu driven C program to implement binary trees using arrays and perform insertion, Deletion and search.

ALGORITHM

```
1 Start
2 Enter tree height, h.
3 compute size = $2^{h+1}-1$, all elements are initialized to -1
4 Enter data for root, create tree from root.
5 Buildtree (i, item)
  5.1 set a[i] = item
  5.2 if a(i) has left child
     5.2.1 Enter data of left child, newL
      5.2.2 call buildtree (2i, newL), goto step 5
  5.3 if a[i] has right child
     5.3.1 Enter data of right child, newR
     5.3.2 call buildtree (2i+1, newR), goto step 5
6 search (i, key)
  6.1 if a(i) = -1 or a(i) = key
     6.1.1 if a[i] = key
           6.1.1.1 set loc = 1
     6.1.2 return i
  6.2 Else if a[(search (2i, key))] = -1
```

Teacher's Signature:

```
    6.2.1 call search (2i+1, key), goto step 6
7 Enter choice for menu
8 case 1: insertion
    8.1 Enter parent node of new node.
    8.2 call search (i, parent), goto step 6, store
        location of parent in loc
    8.3 if a(loc) ! = key
        8.3.1 parent node not found, return
    8.4 if a(2×loc) = -1 or a(2×loc+1) = -1
        8.4.1 if insertion as left child
            8.4.1.1 if a(2×loc) = -1
                8.4.1.1.1 Enter data to insert
                8.4.1.1.2 if 2×loc > size
                    8.4.1.1.2.1 set n++, size = $2^{n+1}-1$
                8.4.1.1.3 set a(2×loc) = data
            8.4.1.2 Else left child is not empty
        8.4.2 if insertion as right child
            8.4.2.1 if a(2×loc+1) = -1
                8.4.2.1.1 Enter data to insert
                8.4.2.1.2 if 2×loc+1 > size
                    8.4.2.1.2.1 set n++, size = $2^{n+1}-1$
                8.4.2.1.3 set a(2×loc+1) = data
            8.4.2.2 Else right child is not empty
    8.5 else left and right children are not
        empty
```

```
9 case 2 : Deletion.
  9.1 enter data to delete.
  9.2 call search (1, node), goto step 6, store location.
  9.3 if a(loc) = data
        9.3.1.1 set a(loc) = -1
        9.3.1.2 if no element exist at same level
              9.3.1.2.1 set h--, size = 2^(h+1) - 1
        9.3.2 Else data is not a leaf node
  9.4 Else, node doesnot exist
10 case 3 : search.
  10.1 Enter node to search
  10.2 call search (1, node), goto step 6, store location
  10.3 if a(loc) = node.
        10.3.1 Node found
  10.4 Else not not found
11 stop
```

CONCLUSION

The program has been executed correctly and the output has been verified.