Breadth First Traversal With Binary Search Tree C++

Maybe fast/simple Question. I have a a Binary Tree Implemented already, Then I was hoping to convert binary search tree into an array or at least print it out as if in an array. Where I am having trouble with is how to get the NULL/flags in there '\0'.

for example lets say I have a tree like:

```
          10
         /  \
        6   12
       / \   \
      1  8   15
       \
        4
```

And I want it to print how its supposed to print. Like:

```
[10,6,12,1,8,\0,15,\0,4,\0,\0,\0,\0,\0,\0]
^Something Like this^ I don't know if I counted the NULL correctly.
```

Or Another Option on how i want to go about showing Visually my Tree is how to get the spacing correctly outputted like with the '/' and '\' pointing to the keys from the parents:

```
          10
         /  \
        6   12
       / \   \
      1  8   15
       \
        4
```

Here is something that I tried elaborating on code wise but im stuck:

```
void BreadthFirstTravseral(struct node* root)
{
    queue<node*> q;

    if (!root) {
        return;
    }
    for (q.push(root); !q.empty(); q.pop()) {
        const node * const temp_node = q.front();
        cout<<temp_node->data << " ";

        if (temp_node->left) {
            q.push(temp_node->left);
        }
        if (temp_node->right) {
            q.push(temp_node->right);
        }
    }
}
```

Any Kind of Help or Link and or advice and or example code would be very much appreciated.

c++ binary-search-tree breadth-first-search

asked Jul 30 '13 at 5:55

Conor Fischer
124 1 6 14

5 Answers

It will be very hard to get the spacing correctly as a key may have multiple digits and this should affect the spacing for all levels above the given node.

As for how to add `NULL` - simply add else clauses for your ifs where you print a NULL:

```
if (root) {
  q.push(root);
  cout << root->data << " ";
} else {
  cout << "NULL ";
}
while (!q.empty()) {
    const node * const temp_node = q.front();
    q.pop();

    if (temp_node->left) {
      q.push(temp_node->left);
      cout << temp_node->left->data << " ";
    } else {
      cout << "NULL ";
    }


    if (temp_node->right) {
      q.push(temp_node->right);
      cout << temp_node->right->data << " ";
    } else {
      cout << "NULL ";
    }
}
```

edited Jul 30 '13 at 7:12

answered Jul 30 '13 at 6:29

Ivaylo Strandjev
50.4k 10 68 123

2 Won't the While loop go forever because nothing is ever being pop'ed off of the q? Infinite loop is what my computer system is implementing. – Conor Fischer Jul 30 '13 at 6:52

@Xaphen yeap thank you. I forgot to pop. Will fix that right away – Ivaylo Strandjev Jul 30 '13 at 7:12


```
void BreadthFirstTravseral(struct node* root)
{
    queue<node*> q;

    if (!root) {
        return;
    }
    for (q.push(root); !q.empty(); q.pop()) {
        const node * const temp_node = q.front();
        if( temp_node->special_blank ){
            cout << "\\0 " ;
            continue;//don't keep pushing blanks
        }else{
            cout<<temp_node->data << " ";
        }
        if (temp_node->left) {
            q.push(temp_node->left);
        }else{
            //push special node blank
        }
        if (temp_node->right) {
            q.push(temp_node->right);
        }else{
            //push special node blank
        }
    }
}
```

answered Jul 30 '13 at 6:12

jeremyvillalobos
1,253 1 15 26

What is the "special_blank" identifier? – Conor Fischer Jul 30 '13 at 6:22

You can add a bool to the node structure. It will be false if not special node, and true otherwise. That way you catch the special blank node and print the empty leaf nodes. – jeremyvillalobos Aug 2 '13 at 6:36

How about this:

```
std::vector<node*> list;
list.push_back(root);
int i = 0;
while (i != list.size()) {
  if (list[i] != null) {
```

```
    node* n = list[i];
    list.push_back(n->left);
    list.push_back(n->right);
  }
  i++;
}
```

Not tested but I think it should work.

answered Jul 30 '13 at 7:13

bb01234
305 2 13

```
void TreeBreadthFirst(Node*  treeRoot)
{
 Queue *queue  = new Queue();

      if (treeRoot == NULL) return;
       queue->insert(treeRoot);
       while (!queue->IsEmpty())
          {
          Node * traverse = queue->dequeue();
         cout<< traverse->data << " " ;
          if (traverse->left != NULL)
            queue->insert( traverse->left);
          if (traverse->right != NULL)
            queue->insert(traverse->right);
           }
      delete queue;
       }
```

edited Dec 2 '13 at 7:32

zanky
2,200 6 43 84

answered Dec 2 '13 at 7:25

nidhi
1

Please, add some documentation along with your code. – Max Dec 2 '13 at 7:42

I've made a program in c. This code will display somewhat like a tree.

```
struct node{
    int val;
    struct node *l,*r;
};

typedef struct node node;


int findDepth(node *t){
    if(!t) return 0;
    int l,r;
    l=findDepth(t->l);
    r=findDepth(t->r);

    return l>r?l+1:r+1;
}

void disp(node *t){
    if(!t)
        return;
    int l,r,i=0;
    node *a[100],*p;
    int front=0,rear=-1,d[100],dep,cur,h;
    a[++rear]=t;
    d[rear]=0;
    cur=-1;
    h=findDepth(t);
    printf("\nDepth : %d \n",h-1);

    while(rear>=front){
        dep = d[front];
        p=a[front++];
        if(dep>cur){
            cur=dep;
            printf("\n");
            for(i=0;i<h-cur;i++) printf("\t");
        }
        if(p){
            printf("%d\t\t",p->val);
            a[++rear]=p->l;
            d[rear]=dep+1;
            a[++rear]=p->r;
            d[rear]=dep+1;
        }
        else printf ("-\t\t");

    }
}
```

answered Apr 4 '15 at 13:56

Jayadeep KM
1

Add some explanation for the code and edit your answer. – gsamaras Apr 4 '15 at 14:02