

```

1  /*
2  给出一个完全二叉树，求出该树的节点个数。
3
4  说明：
5
6  完全二叉树的定义如下：在完全二叉树中，除了最底层节点可能没填满外，其余每层节点数都达到最大值，并且最下面一层的节点都集中在该层最左边的若干位置。若最底层为第  $h$  层，则该层包含  $1 \sim 2^h$  个节点。
7
8  示例：
9
10 输入：
11      1
12     /\
13    2  3
14   /\ /\
15  4 5 6
16
17 输出：6
18
19 来源：力扣（LeetCode）
20 链接：https://leetcode-cn.com/problems/count-complete-tree-nodes
21 著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
22 */

```

分析:

- 方法一:常规方法,直接递归或迭代dfs统计二叉树的节点个数即可.
- 方法二:使用迭代方法,记录下二叉树的层数和空节点数,根据二叉树层数与节点数的关系.

奇怪的是方法二居然比方法一耗时多了.

方法一:C++_dfs递归

```

1  /**
2   * Definition for a binary tree node.
3   * struct TreeNode {
4   *     int val;
5   *     TreeNode *left;
6   *     TreeNode *right;
7   *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
8   * };
9   */
10 /**
11  * Definition for a binary tree node.
12  * struct TreeNode {
13  *     int val;
14  *     TreeNode *left;
15  *     TreeNode *right;
16  *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
17  * };
18  */
19 class solution

```

```

20 {
21     private:
22         int __nodeNum(TreeNode* node)
23         {
24             if(node == NULL)
25             {
26                 return 0;
27             }
28             return 1 + __nodeNum(node->left) + __nodeNum(node->right);
29         }
30
31     public:
32         int countNodes(TreeNode* root)
33         {
34             if(root!=NULL)
35             {
36                 return 1 + __nodeNum(root->left) + __nodeNum(root->right);
37             }
38             return 0;
39         }
40 };
41 /*
42 执行结果:
43 通过
44 显示详情
45 执行用时 :48 ms, 在所有 C++ 提交中击败了74.73% 的用户
46 内存消耗 :28.6 MB, 在所有 C++ 提交中击败了86.43%的用户
47 */

```

方法一:C++_迭代BFS

```

1  /**
2   * Definition for a binary tree node.
3   * struct TreeNode {
4   *     int val;
5   *     TreeNode *left;
6   *     TreeNode *right;
7   *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
8   * };
9   */
10 /**
11  * Definition for a binary tree node.
12  * struct TreeNode {
13  *     int val;
14  *     TreeNode *left;
15  *     TreeNode *right;
16  *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
17  * };
18  */
19 class Solution
20 {
21     public:
22         int countNodes(TreeNode* root)
23         {
24             queue<TreeNode*> qtn;
25             TreeNode* temp = NULL;

```

```

26         int                ret_val = 0        ;
27         int                num_1  = 0        ;
28         int                num_2  = 0        ;
29
30         if(root)
31         {
32             qtn.push(root);
33             num_2 = 1;
34             num_1 = 0;
35
36             while(!qtn.empty())
37             {
38                 ret_val += num_2;
39                 num_1 = num_2;
40                 num_2 = 0;
41                 for(int i = 0 ; i < num_1;i++)
42                 {
43                     temp = qtn.front();
44                     qtn.pop();
45                     if(temp->left)
46                     {
47                         qtn.push(temp->left);
48                         num_2++;
49                     }
50                     if(temp->right)
51                     {
52                         qtn.push(temp->right);
53                         num_2++;
54                     }
55                 }
56             }
57         }
58         return ret_val;
59     }
60 };
61
62 /*
63 执行结果:
64 通过
65 显示详情
66 执行用时 :80 ms, 在所有 C++ 提交中击败了15.54% 的用户
67 内存消耗 :29 MB, 在所有 C++ 提交中击败了17.44%的用户
68 */

```

方法二:C++_迭代BFS

```

1  /**
2   * Definition for a binary tree node.
3   * struct TreeNode {
4   *     int val;
5   *     TreeNode *left;
6   *     TreeNode *right;
7   *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
8   * };
9   */
10 /**

```

```

11  * Definition for a binary tree node.
12  * struct TreeNode {
13  *      int val;
14  *      TreeNode *left;
15  *      TreeNode *right;
16  *      TreeNode(int x) : val(x), left(NULL), right(NULL) {}
17  * };
18  */
19  /**
20  * Definition for a binary tree node.
21  * struct TreeNode {
22  *      int val;
23  *      TreeNode *left;
24  *      TreeNode *right;
25  *      TreeNode(int x) : val(x), left(NULL), right(NULL) {}
26  * };
27  */
28  class Solution
29  {
30      public:
31          int countNodes(TreeNode* root)
32          {
33              queue<TreeNode*>    qtn                ;
34              TreeNode*          temp              =  NULL    ;
35              int                 ret_val           =  0        ;
36              int                 numNull           =  0        ;
37              int                 numLevel          =  0        ;
38              int                 num_1             =  0        ;
39              int                 num_2             =  0        ;
40
41              if(root)
42              {
43                  qtn.push(root);
44                  num_2 = 1;
45                  num_1 = 0;
46                  while(1)
47                  {
48                      numLevel++; /*计算层数*/
49                      num_1 = num_2;
50                      num_2 = 0;
51                      for(int i = 0; i < num_1 ; i++)
52                      {
53                          temp = qtn.front();
54                          qtn.pop();
55                          if(temp->left!=NULL)
56                          {
57                              qtn.push(temp->left);
58                              num_2++;
59                          }
60                          else
61                          {
62                              numNull++;
63                          }
64                          if(temp->right!=NULL)
65                          {
66                              qtn.push(temp->right);
67                              num_2++;
68                          }

```

```

69         else
70         {
71             numNull++;
72         }
73     }
74     /*完全二叉树非满二叉树*/
75     if(numNull!=0)
76     {
77         numLevel++;
78         break;
79     }
80     /*满二叉树*/
81     if(num_2==0)
82     {
83         numNull = 0;
84     }
85 }
86 return pow(2,numLevel)-1-numNull;
87 }
88 else
89 {
90     return 0;
91 }
92 }
93 };
94
95 /*
96 执行结果:
97 通过
98 显示详情
99 执行用时 :72 ms, 在所有 C++ 提交中击败了27.59% 的用户
100 内存消耗 :29.3 MB, 在所有 C++ 提交中击败了6.98%的用户
101 */

```