**Level order traversal and family:**

1. #include<iostream>
2. using namespace std;
3. struct node
4. {
5. node \*l,\*r;
6. int a;
7. }\*head='\0',\*root,\*temp,\*req,\*p,\*pa;
8. void create(node \*root);
9. int i;
10. node \*sea(node \*root,int data);
11. int height(node \*root)
12. {
13. if(root=='\0')
14. return 0;
15. else
16. {
17. int lheight=(height(root->l));
18. int rheight=(height(root->r));
19. if(lheight>rheight)
20. return lheight+1;
21. else
22. return rheight+1;
23. }
24. }
25. node \*printgivenlevel(node \*root,int level)
26. {
27. if(root=='\0')
28. return '\0';
29. if(level==1)
30. cout<<root->a<<" ";
31. else if(level>1)
32. {
33. printgivenlevel(root->l,level-1);
34. printgivenlevel(root->r,level-1);
35. }
36. }
37. void levelorder(node \*root)
38. {
39. int h=height(root);
40. for(int i=1;i<=h;++i)
41. printgivenlevel(root,i);
42. }
43. int main()
44. {
45. int x;
46. node \*n=new node;
47. n->l='\0';
48. n->r='\0';
49. cout<<"\n enter data:";
50. cin>>n->a;
51. root=n;
52. head=n;
53. create(root);
54. root=head;
55. cout<<"\n levelorder:";
56. levelorder(root);
57. cout<<"\n enter the data to be searched:";
58. cin>>x;
59. root=head;
60. p=sea(root,x);
61. if(req!=head)
62. {
63. cout<<"\n parent is:";
64. cout<<pa->a;
65. cout<<"\n sibling is:" ;
66. if(i=1 && pa->r!='\0' && pa->r->a!=req->a)
67. cout<<pa->r->a;
68. else if(i=2 &&pa->l!='\0' && pa->l->a!=req->a)
69. cout<<pa->l->a;
70. else
71. cout<<"\n no sibling";
72. cout<<"\n children:";
73. if(req->l!='\0')
74. {
    1. cout<<"\n leftchild:";
    2. cout<<req->l->a;
75. }
76. if(req->r!='\0')
77. {
    1. cout<<"\n rightchild:";
    2. cout<<req->r->a;
78. }}
79. else
80. {
81. cout<<"\n no parent :";
82. cout<<"\n no sibling";
83. cout<<"\n children:";
84. if(req->l!='\0')
85. {
    1. cout<<"\n leftchild:";
    2. cout<<req->l->a;
86. }
87. if(req->r!='\0')
88. {
    1. cout<<"\n rightchild:";
    2. cout<<req->r->a;
89. }
90. }
91. return 0;
92. }
93. void create(node \*root)
94. {
95. char ch;
96. cout<<"\n leftchild?";
97. cin>>ch;
98. if(ch=='y')
99. {
100. node \*n=new node;
101. n->l='\0';
102. n->r='\0';
103. cout<<"\n enter data:";
104. cin>>n->a;
105. root->l=n;
106. }
107. cout<<"\n rightchild?";
108. cin>>ch;
109. if(ch=='y')
110. {
111. node \*n=new node;
112. n->l='\0';
113. n->r='\0';
114. cout<<"\n enter data:";
115. cin>>n->a;
116. root->r=n;
117. }
118. if(root->l!='\0')
119. create(root->l);
120. if(root->r!='\0')
121. create(root->r);
122. }
123. node \*sea(node \*root,int data)
124. {
125. p=root;
126. if(head->a==data)
127. req=head;
128. if(root->l!='\0'&& root->a!=data)
129. {
130. temp=sea(root->l,data);
131. if(temp!='\0')
132. {
     1. i=1;
     2. req=temp;
     3. pa=root;
133. }
134. }
135. if(root->r!='\0'&& root->a!=data)
136. {
137. temp=sea(root->r,data);
138. if(temp!='\0')
139. {
140. i=2;
141. req=temp;
142. pa=root;
143. }
144. }
145. if(root->a==data)
146. return root;
147. else
148. return '\0';
149. }