《数据结构》课程实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 专业名称 | 计算机科学与技术（国际联合培养） | 年级 | 大二 | 班级 | 计联培1901 |
| 学生姓名 | 姬彬荃 | 学号 | 201911020125 | 指导教师 | 魏晓超 |
| 实验题目 | 实验七：树的基本操作和遍历 | | | 提交时间 | 2020.10.28 |

一、实验目的和要求

1、前序遍历生成二叉树；

2、前序、中序、后序遍历上述生成的二叉树（使用递归）；

3、使用非递归方式中序遍历二叉树；

4、输出二叉树的深度、节点个数。

二、实验环境

Visual Studio 2019

三、实验内容及实施

（包括设计思想、源程序、参数说明、采用的关键理论、技术、算法或数据结构等的说明。）

实验要求：

1、前序遍历生成二叉树；

2、前序、中序、后序遍历上述生成的二叉树（使用递归）；

3、使用非递归方式中序遍历二叉树；

4、输出二叉树的深度、节点个数。

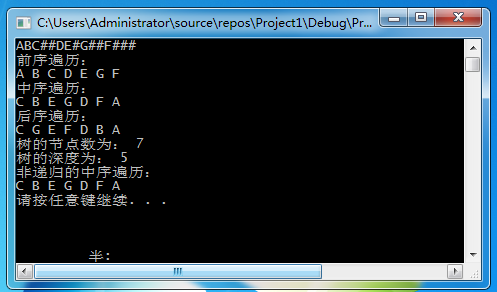
**【源程序】**

****

1. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
2. \*\* 功能 : 数据结构实验七：二叉树
3. \*\* 作者 : 2019Jibinquan
4. \*\* 创建 : 2020-10-21 / 15:59
5. \*\* 联系 : 431263064@qq.com
6. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
8. #include <iostream>
9. #include <cstdio>
10. #include <cmath>
11. #include <algorithm>
12. #include <cstring>
13. #include <queue>
14. #include <stack>
15. #include <vector>
16. #include <map>
17. #include <string>
18. #include <set>
19. #define ios ios::sync\_with\_stdio(false);cin.tie(0);cout.tie(0);
20. #define debug(a) cout << #a << " " << a << endl
21. **using** **namespace** std;
22. **typedef** **long** **long** ll;
23. **const** **double** pi = acos(-1);
24. **const** **double** eps = 1e-8;
25. **const** **int** inf = 0x3f3f3f3f;
26. **const** **int** maxn = 100007;//1e5+7
27. **const** ll mod = 1000000007;//1e9+7
29. **typedef** **struct** node {
30. **char** data;
31. node\* lson, \*rson;
32. }node, \*root;
34. string st;
35. **int** pos;
37. **void** build(root &o)
38. {
39. //cout << st[pos] << endl;
40. **if** (st[pos] == '#') {
41. o = NULL;
42. }
43. **else** {
44. o = **new** node;
45. o->data = st[pos];
46. **if** (pos + 1 < st.length()) {
47. pos++;
48. build(o->lson);
49. }
50. **else** {
51. o->lson = NULL;
52. }
53. **if** (pos + 1 < st.length()) {
54. pos++;
55. build(o->rson);
56. }
57. **else** {
58. o->rson = NULL;
59. }
60. }
62. }
64. **void** DLR(root o)
65. {
66. **if** (o != NULL) {
67. cout << o->data << ' ';
68. DLR(o->lson);
69. DLR(o->rson);
70. }
71. }

74. **void** LDR(root o)
75. {
76. **if** (o != NULL) {
77. LDR(o->lson);
78. cout << o->data << ' ';
79. LDR(o->rson);
80. }
81. }
83. **void** LRD(root o)
84. {
85. **if** (o != NULL) {
86. LRD(o->lson);
87. LRD(o->rson);
88. cout << o->data << ' ';
89. }
90. }


94. stack<root>sta;
96. **void** LDR\_stack(root o)
97. {
98. node\* p = o;
99. node\* q = **new** node;
100. **while** (p || !sta.empty()) {
101. **if** (p) {
102. sta.push(p);
103. p = p->lson;
104. }
105. **else** {
106. q = sta.top();
107. sta.pop();
108. cout << q->data << ' ';
109. p = q->rson;
110. }
111. }
112. }
114. **int** countNode(root o)
115. {
116. **if** (o != NULL) {
117. **return** countNode(o->lson) + countNode(o->rson) + 1;
118. }
119. **else** **return** 0;
121. }
123. **int** mxdep(root o)
124. {
125. **if** (o != NULL) {
126. **return** max(mxdep(o->lson), mxdep(o->rson)) + 1;
127. }
128. **return** 0;
130. }
132. /\*----------------------------------\*
133. Main Function
134. \*-----------------------------------\*/
136. **int** main()
137. {
138. node\* o = NULL;
139. cin >> st;
140. pos = 0;
141. build(o);
142. cout << "前序遍历：" << endl;
143. DLR(o);
144. cout << endl;
145. cout << "中序遍历：" << endl;
146. LDR(o);
147. cout << endl;
148. cout << "后序遍历：" << endl;
149. LRD(o);
150. cout << endl;
151. cout << "树的节点数为： " << countNode(o) << endl;
152. cout << "树的深度为： " << mxdep(o) << endl;
153. cout << "非递归的中序遍历：" << endl;
154. LDR\_stack(o);
155. cout << endl;
156. system("pause");
157. **return** 0;
158. }

****

**五、实验讨论（可选）**

暂无