DataStructureHW4

2013011372박병욱

1. Assignment

이진트리를 만들고 preorder, inorder, postorder traverse한다.

1. Result

$./hw4



$./hw4 input2.txt



1. 소스코드
2. //for visual studio 2013(fscanf\_s problem)
3. #define \_CRT\_SECURE\_NO\_WARNINGS
4. #include <stdio.h>
5. #include <stdlib.h>
6. #include <memory.h>
7. #include <string.h>
8. typedef int element;
9. typedef struct \_TreeNode
10. {
11. element data;
12. struct \_TreeNode \* left;
13. struct \_TreeNode \* right;
14. } TreeNode;
15. //새로운 트리노드를 만든다.
16. TreeNode \* createTreeNode(void)
17. {
18. TreeNode\* node;
19. node = (TreeNode\*)malloc(sizeof(TreeNode));
20. node->left = NULL;
21. node->right = NULL;
22. return node;
23. }
24. //인자로 넘어온 노드에 데이터를 넣어준다.
25. void setData(TreeNode \* node, element data)
26. {
27. node->data = data;
28. }
29. //인자로 넘어온 노드의 값을 얻는다.
30. element getData(TreeNode \* node)
31. {
32. return node->data;
33. }
34. //main 인자로 넘어온 노드의 왼쪽서브트리로 sub를 넣는다.
35. void makeLeftSubTree(TreeNode \* main, TreeNode \* sub)
36. {
37. main->left = sub;
38. }
39. //main 인자로 넘어온 노드의 오른쪽서브트리로 sub를 넣는다.
40. void makeRightSubTree(TreeNode \* main, TreeNode \* sub)
41. {
42. main->right = sub;
43. }
44. //인자로 넘어온 노드의 왼쪽서브트리를 얻는다.
45. TreeNode \* getLeftSubTree(TreeNode \* node)
46. {
47. return node->left;
48. }
49. //인자로 넘어온 노드의 오른서브트리를 얻는다.
50. TreeNode \* getRightSubTree(TreeNode \* node)
51. {
52. return node->right;
53. }
54. //중위순회하며 트리를 출력한다.
55. void printInorder(TreeNode \*root){
56. if(root == NULL) return;
57. printInorder(getLeftSubTree(root));
58. printf("%d ", root->data);
59. printInorder(getRightSubTree(root));
60. }
61. //전위순회하며 트리를 출력한다.
62. void printPreorder(TreeNode \*root){
63. if(root == NULL) return;
64. printf("%d ", root->data);
65. printPreorder(getLeftSubTree(root));
66. printPreorder(getRightSubTree(root));
67. }
68. //후위순회하며 트리를 출력한다.
69. void printPostorder(TreeNode \*root){
70. if(root == NULL) return;
71. printPostorder(getLeftSubTree(root));
72. printPostorder(getRightSubTree(root));
73. printf("%d ", root->data);
74. }
75. int main(int argc, char \*argv[]){
76. FILE \*input;
77. char command;
78. TreeNode \*root;
79. root = createTreeNode();
80. //트리노드를 만든다.
81. TreeNode \* t1 = createTreeNode();
82. TreeNode \* t2 = createTreeNode();
83. TreeNode \* t3 = createTreeNode();
84. TreeNode \* t4 = createTreeNode();
85. TreeNode \* t5 = createTreeNode();
86. TreeNode \* t6 = createTreeNode();
87. TreeNode \* t7 = createTreeNode();
88. TreeNode \* t8 = createTreeNode();
89. TreeNode \* t9 = createTreeNode();
90. //트리노드들에 값을 넣어준다.
91. setData(root, 54);
92. setData(t1, 27);
93. setData(t2, 13);
94. setData(t3, 1);
95. setData(t4, 44);
96. setData(t5, 37);
97. setData(t6, 89);
98. setData(t7, 71);
99. setData(t8, 64);
100. setData(t9, 92);
101. //트리노드의 구조들을 연결해준다.
102. makeLeftSubTree(root, t1);
103. makeRightSubTree(root, t6);
104. makeLeftSubTree(t1, t2);
105. makeRightSubTree(t1, t4);
106. makeLeftSubTree(t2, t3);
107. makeLeftSubTree(t4, t5);
108. makeLeftSubTree(t6, t7);
109. makeRightSubTree(t6, t9);
110. makeLeftSubTree(t7, t8);
111. //입력 파일을 지정하지 않았을 경우
112. if(argc == 1)
113. input = fopen("input.txt", "r");
114. //입력 파일을 지정했을 경우
115. else if(argc == 2)
116. input = fopen(argv[1], "r");
117. //아규먼트가 많을 경우
118. else error("Usage : $HW4 (arg)\n");
119. //파일을 찾을 수 없을경우
120. if(input == NULL){
121. printf("Can't find the file\n");
122. exit(0);
123. }
124. while(1){
125. command = fgetc(input);
126. if(feof(input)) break;//파일을 끝까지 읽었을경우
127. switch(command) {
128. case 'r':
129. printf("Preorder:\t");
130. printPreorder(root);
131. printf("\n");
132. break;
133. case 'i':
134. printf("Inorder:\t");
135. printInorder(root);
136. printf("\n");
137. break;
138. case 'o':
139. printf("Postorder:\t");
140. printPostorder(root);
141. printf("\n");
142. break;
143. default:
144. break;
145. }
146. }
147. fclose(input);
148. return 0;
149. } 소스코드 설명

주어진 given..c를 이용하여 구현하였다.

arg[1]를 받아 인풋파일로 이용하였으며,

아규먼트가 다수일 경우 프로그램을 종료하였다.

파일을 열 수 없을 경우 프로그램을 종료하였다..

While문 안에서 fgetc로 input파일을 읽었다.

줄의 시작에 따라 preorder, inorder, postorder함수를 구현하였고, 함수에 따라 재귀 구조를 달리하여 구현하였다.