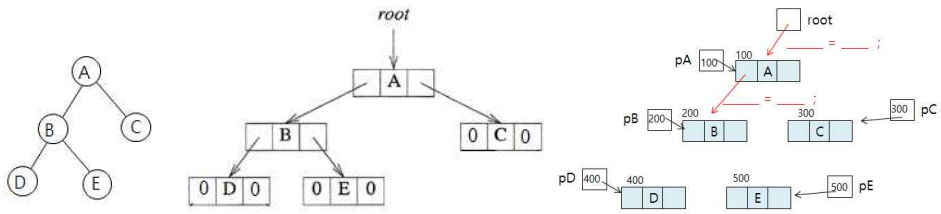
자료구조응용

Trees : 이진트리 생성

1. **[이진트리] 다음과 같은 트리를 생성하고 이진트리 순회방법 중 중위순회(inorder traversal), 전위순회(preorder traversal), 후위순회(postorder traversal)를 통해 출력하는 프로그램을 작성하라. (3점)**

① 다음과 그림과 같은 이진트리를 생성한다. ※ createBinTree();
※ 데이터는 그림과 같이 A, B, C, D, E를 사용한다.
※ 개별적으로 노드를 생성한 후, 그림과 같은 형태의 이진트리가 되게 링크를 연결한다.



② 이진트리 중위순회를 통해 데이터를 출력한다. ※ inorder(root);
③ 이진트리 전위순회를 통해 데이터를 출력한다. ※ preorder(root);
④ 이진트리 후위순회를 통해 데이터를 출력한다. ※ postorder(root);

(2) 구현 세부사항

① 트리 정의

```
typedef struct node *treePointer;
typedef struct node {
        char data; // 문자출력을 위해 char 형으로 지정
        treePointer leftChild, rightChild;
}tNode;
treePointer root;
```

② 함수정의
직접 정의 : createNode, createBinTree
교재 함수 : inorder, preorder, postorder

※ createNode()에서 생성하는 노드의 leftChild, rightChild는 초기값이 NULL이다.
※ deleteNode()는 구현하지 않음
※ 아래 세 함수에서 printf("%c", ptr->data); 사용

```
void inorder(treePointer ptr)
{/* inorder tree traversal */
   if (ptr) {
      inorder(ptr→leftChild);
      printf("%d",ptr→data);
      inorder(ptr→rightChild);
   }
}
```

Program 5.1: Inorder traversal of a binary tree

```
void preorder(treePointer ptr)
{/* preorder tree traversal */
   if (ptr) {
      printf("%d",ptr→data);
      preorder(ptr→leftChild);
      preorder(ptr→rightChild);
   }
}
```

Program 5.2: Preorder traversal of a binary tree

```
void postorder(treePointer ptr)
{/* postorder tree traversal */
   if (ptr) {
      postorder(ptr→leftChild);
      postorder(ptr→rightChild);
      printf("%d",ptr→data);
   }
}
```

Program 5.3: Postorder traversal of a binary tree

(3) 실행 예

DBEAC //inorder

ABDEC //preorder

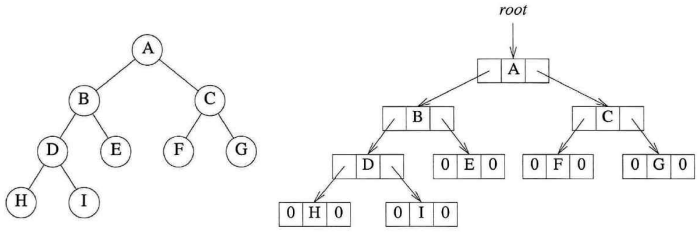
DEBCA //postorder

2. **[큐를 이용한 완전이진트리 생성] 파일입력을 받아 다음과 같은 완전이진트리(complete binary tree)를 구성하여, 이진트리 순회방법 중 중위순회, 전위순회, 후위순회를 통해 출력하는 프로그램을 작성하라. (7점)**

(1) 실행순서

① 입력파일(input.txt)로부터 다음과 같은 완전이진트리를 생성한다.

입력
ABCDEFGHI



② 이진트리 중위순회를 통해 데이터를 출력한다. ※ inorder(root);

③ 이진트리 전위순회를 통해 데이터를 출력한다. ※ preorder(root);

④ 이진트리 후위순회를 통해 데이터를 출력한다. ※ postorder(root);

(2) 구현 세부사항

① 트리 노드 정의 및 큐 선언

```
typedef struct node *treePointer;
typedef struct node {
        char data; // 문자출력을 위해 char 형으로 지정
        treePointer leftChild, rightChild;
}tNode;
treePointer root;
treePointer queue[MAX_QUEUE_SIZE];
```

② queue

- MAX_QUEUE_SIZE를 100으로 한 선형큐(linear queue)를 사용
- addq, deleteq, queueFull, deleteEmpty를 정의함
- queueFull은 간단한 메시지를 출력하고 프로그램을 종료함
- deleteEmpty는 간단한 메시지를 출력하고 NULL 포인터가 반환되도록 함
- getFront : 큐의 가장 선두항목 값을 반환하는 함수. 큐의 항목을 삭제하지 않음

③ 완전이진트리 생성 함수정의

```
treePointer createNode( char data );
treePointer createCompBinTree(FILE *fp);
void insert( treePointer *pRoot, treePointer pNode );
int hasBothChild(treePointer pNode);
```

④ 큐를 사용한 완전이진트리 생성

- createCompBinTree : 데이터를 입력받을 때 마다 노드를 생성하여 insert 수행
- insert 알고리즘 :

완전이진트리에 대한 노드 삽입 (insert)
1. *If* the tree is empty, initialize the root with *new node*. 2. *Else*, **get the front node** of the queue. *if* the left child of this front node doesn't exist, set the left child as the new node. *else if* the right child of this front node doesn't exist, set the right child as the new node. *If* the front node has both the left child and right child, **Dequeue()** it. 3. **Enqueue()** the *new node*.

※ 큐에 항목을 추가하는 연산을 enqueue, 삭제하는 연산을 dequeue라고 하며, 교재에서는 addq, deleteq 함수로 구현되어 있다.

※ 2번에서 큐의 front node를 가져온다는 것은 front 노드에 대한 삭제가 아니고 선두 항목의 값을 return받아 사용한다는 의미

※ Self-check

- 언제 큐에 추가하나?
- 언제 큐에서 삭제하나?
- 큐에는 항상 어떤 특성의 노드를 유지하는가?

(3) 실행 예

ABCDEFGHI
HDIBEAFCG //inorder
ABDHIECFG //preorder
HIDEBFGCA //postorder