자료구조응용

13. Trees : 이진트리 생성 및 순회 (1) (15점)

2022.4.25.

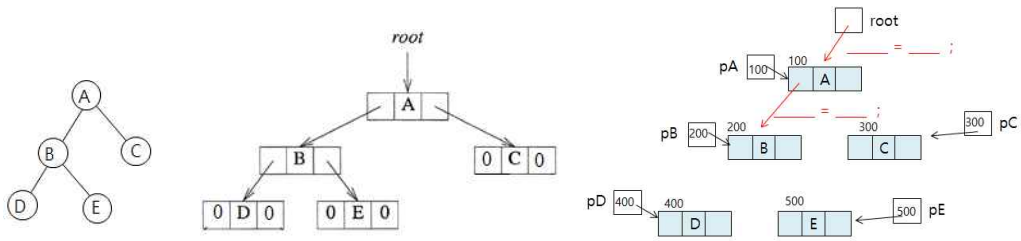
1. **[이진트리] 다음과 같은 트리를 생성하고 이진트리 순회방법 중 중위순회(inorder traversal), 전위순회(preorder traversal), 후위순회(postorder traversal)를 통해 출력하는 프로그램을 작성하라. (5점)**

[실행순서]

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);

[구현 세부사항]

① 트리 정의

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

② 함수정의

```
// binary tree
treePointer createNode( char data ); // 노드 생성 함수이며 링크 필드는 NULL로 지정
treePointer createBinTree();
※ deleteNode()는 구현하지 않음
// binary tree traversals
void inorder(treePointer ptr);
void preorder(treePointer ptr);
void postorder(treePointer ptr);
```

※ 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

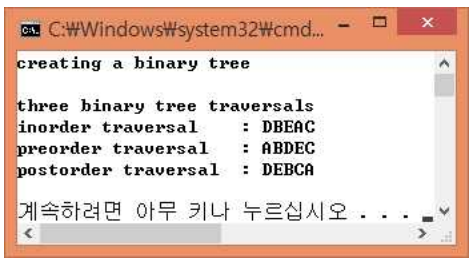
```
void preorder(treePointer ptr)
{/* preorder tree traversal */
   if (ptr) {
      printf("%d",ptr→data);
      preorder(ptr→leftChild);
      preorder(ptr→rightChild);
   }
}
```

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

[실행예]

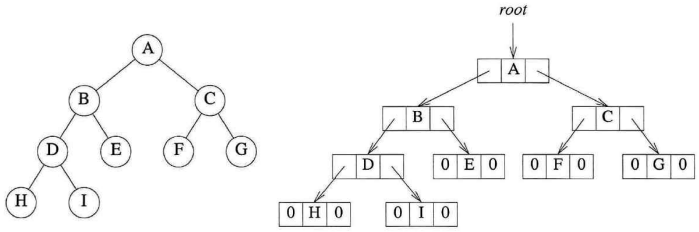


2. **[큐를 이용한 완전이진트리 생성] 파일입력을 받아 다음과 같은 완전이진트리(complete binary tree)를 구성하여, 이진트리 순회방법 중 중위순회, 전위순회, 후위순회를 통해 출력하는 프로그램을 작성하라. (10점)**

[실행순서]

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

입력파일(input.txt)
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);

[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;
}node;
treePointer root;
treePointer queue[MAX_QUEUE_SIZE];
```

② queue

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

③ 완전이진트리 생성관련 함수정의 ※ 아래 정의를 참고하되 각자의 방법으로 구현해도 됨

```
treePointer createNode( char data );  // 노드 생성 함수이며 링크 필드는 NULL로 지정
treePointer createCompBinTree(FILE *fp);
void insert( treePointer *pRoot, treePointer pNode );
        // pRoot가 가리키는 완전이진트리에 pNode가 가리키는 새 노드를 추가한다.
```

④ 큐를 사용한 완전이진트리 생성

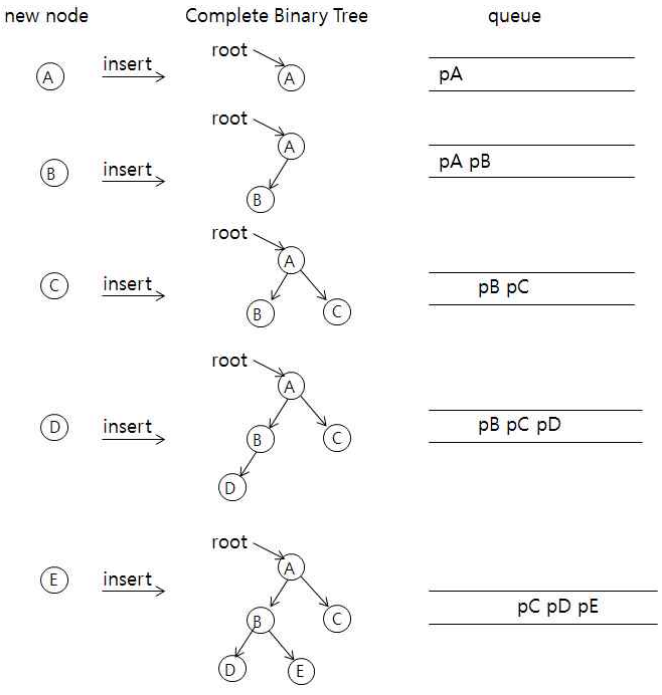
- createCompBinTree : 데이터를 입력받을 때 마다 노드를 생성하여 insert 수행
- insert 알고리즘 :

완전이진트리 생성을 위한 노드 삽입 알고리즘

```
void insert( treePointer *pRoot, treePointer pNode )
{
        if  the tree is empty,
                initialize the root with the new node.
        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 dosen't exist,
                        set the right child as the new node.
                        deleteq() // ∵ The front node has both children.
        addq() the new node.
}
```

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

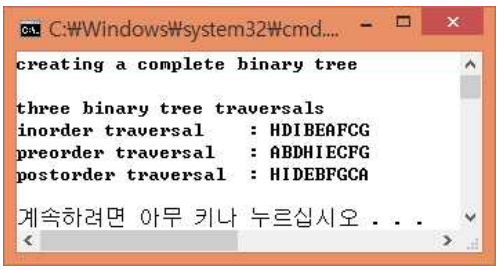
완전이진트리 생성 과정



※ pA는 노드 A 대한 포인터

※ Self-check
- 언제 큐에 추가하나?
- 언제 큐에서 삭제하나?
- 큐에는 항상 어떤 특성의 노드를 유지하는가?
- 마지막 노드 삽입 후 큐에는 어떤 정보들이 남아 있나?

[실행예]



[문자 형식으로 파일입력 시 주의사항]

```
FILE *fp;
char data;

if((fp = fopen("input.txt", "r"))  == NULL )
{           fprintf(stderr, "cannot open the file");
            exit(EXIT_FAILURE);
}

// <<<<<<<<<<<<<<<<<< 파일입력 방법 1~2 >>>>>>>>>>>>>>>>>>>>>>>>>
// 파일 끝의 white space(엔터키, 탭, 스페이스바 공백 등) 유무에 상관없이 제대로 입력됨

// 입력방법 1 : ( input.txt ABCDEFGHI ) or ( input.txt A B C D E F G H I )
while (fscanf(fp, "%c ", &data) != EOF)
{
            ⋮
}

// 입력방법 2 : ( input.txt ABCDEFGHI ) or ( input.txt A B C D E F G H I )
/*
fscanf(fp,"%c ", &data);
while(!feof(fp))
{
            ⋮
            fscanf(fp,"%c ", &data);
}
*/
```

■ 제출 형식

- 솔루션 이름 : DS 13
- 프로젝트 이름 : 1, 2
- 각 소스파일에 주석처리

  “학번 이름”

  “본인은 이 소스파일을 다른 사람의 소스를 복사하지 않고 직접 작성하였습니다.”
- 실행화면을 캡쳐한 보고서를 작성 후 pdf 파일로 변환하여 솔루션 폴더에 포함
- 솔루션 정리 메뉴를 수행 후 전체 솔루션을 “학번.zip”으로 압축하여 제출

■ 주의

- 소스 복사로는 실력향상을 기대할 수 없습니다!!!

- 1차 마감 (LMS 과제 마감일) : 수업일 자정

- 2차 마감 (LMS 과제 열람기한) : 수업 익일 자정(만점의 50%, 반올림)

- 1, 2차 마감 이외의 제출은 허용하지 않습니다. (이메일 제출 불가!)