자료구조응용

14. Trees : 이진트리 생성 및 순회 (2) (15점)

2022.4.27.

1. 후위표현식을 입력받아 Figure 5.16과 같은 산술식의 이진트리를 구성한 후, 이진트리 순회를 통해 중위표현식, 전위표현식, 후위표현식을 출력하는 프로그램을 작성하라. (5점)

[실행순서]

① 후위표현식으로 부터 산술식의 이진트리를 생성한다. ※ createBinTree();

데이터 입력형식
- 입력파일(input.txt) : AB/C*D*E+
- 피연산자(Operands) : 알파벳 한 글자
- 연산자(Operators) : +, -, *, /, %

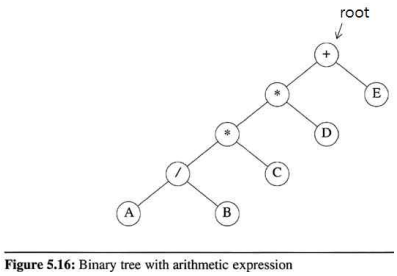


Figure 5.16: Binary tree with arithmetic expression

※ 자료구조응용 09. 1번 프로그램을 수정하여 구현할 수 있음
- ***eval()을 수정***하여 후위표현식을 이진트리로 만드는 createBinTree() 함수를 정의
- getToken() 과 precedence 형은 그대로 사용, stack은 일부 수정

※ 산술식의 이진트리에서 연산자 노드의 왼쪽 서브트리는 그 연산자의 왼쪽 피연산자로 사용되고 오른쪽 서브트리는 오른쪽 피연산자로 사용됨

② 이진트리 중위순회를 통해 중위표현식(infix expression)을 출력한다. ※ inorder(root);
③ 이진트리 전위순회를 통해 전위표현식(prefix expression)을 출력한다. ※ preorder(root);
④ 이진트리 후위순회를 통해 후위표현식(postfix expression)을 출력한다. ※ postorder(root);

[구현 세부사항]

```
typedef struct node *treePointer;
typedef struct node {
        char data; // 문자출력을 위해 char 형으로 지정
        treePointer leftChild, rightChild;
}node;
treePointer root;
treePointer stack[MAX_STACK_SIZE];
int top = -1;
char expr[81];   // postfix expression
```

```
treePointer createBinTree(void)
{ /* postfix expression으로부터 이진트리를 만든 후 루트 노드 포인터를 반환 */
        treePointer pNode;
        precedence token;
        char symbol;
        int n = 0; /* counter for the expression string */
        top = -1;
        token = getToken(&symbol, &n);
        while( token != eos )
        {
                pNode = createNode(symbol);
                if( token != operand )                          // operator node
                {
                        pNode->rightChild =  pop();             // link operand
                        pNode->leftChild = pop();               // link operand
                }
                push(pNode);

                token = getToken( &symbol, &n);
        }
        return pop(); /* return root pointer */
}
```

※ 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

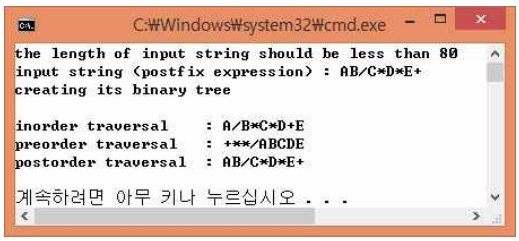
```
void preorder(treePointer ptr)
{/* preorder tree traversal */
   if (ptr) {
      printf("%d",ptr->data);
      preorder(ptr->leftChild);
      preorder(ptr->rightChild);
   }
}
```

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. 후위표현식을 입력받아 Figure 5.16과 같은 이진트리를 구성한 후, 반복문을 사용한 중위, 전위 및 level-order 순회를 하는 프로그램을 작성하라.(10점)

※ 중위, 전위, 레벨오더 순회는 각 3, 4, 3점 ※ 후위순회는 도전과제 (5점 추가)

[실행순서]

① 후위표현식으로 부터 산술식의 이진트리를 생성한다. ※ createBinTree();

※ 1번 실행순서 ①과 동일

데이터 입력형식
- 입력파일(input.txt) : AB/C*D*E+
- 피연산자(Operands) : 알파벳 한 글자
- 연산자(Operators) : +, -, *, /, %

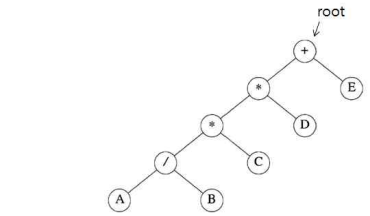


Figure 5.16: Binary tree with arithmetic expression

② **반복문**을 사용한 이진트리 **중위순회, 전위순회**를 통해 데이터를 출력한다.

※ iterInorder(root); iterPreorder(root);

※ ①에서 사용한 스택을 그대로 사용

③ **level-order traversal**을 통해 각 노드의 데이터를 출력한다.

※ levelOrder(root); // circular queue 사용

[구현 세부사항]

```
typedef struct node *treePointer;
typedef struct node {
        char data; // 문자출력
        treePointer leftChild, rightChild;
}node;
treePointer root;

// stack
treePointer stack[MAX_STACK_SIZE];
int top = -1;

// circular queue
treePointer queue[MAX_QUEUE_SIZE];
int front=0, rear=0;
```

- 데이터 출력 시 printf("%c", ptr->data); 사용
- stack과 queue의 각 항목은 treePointer 타입
- stack empty 및 queue empty는 NULL을 리턴

① 이진트리 순회

```
int top = -1; /* initialize stack */
treePointer stack[MAX_STACK_SIZE];
```

```
void iterInorder(treePointer node)
{
   top = -1;
   for (;;) {
      for(; node; node = node->leftChild)
         push(node); /* add to stack */
      node = pop(); /* delete from stack */
      if (!node) break; /* empty stack */
      printf("%d", node->data);
      node = node->rightChild;
   }
}
```

Program 5.4: Iterative inorder traversal

```
int front = rear = 0; // circular queue
treePointer queue[MAX_QUEUE_SIZE];
```

```
void levelOrder(treePointer ptr)
{/* level order tree traversal */
   front = rear = 0;
   if (!ptr) return; /* empty tree */
   addq(ptr);
   for (;;) {
      ptr = deleteq();
      if (ptr) {
         printf("%d",ptr->data);
         if(ptr->leftChild)
            addq(ptr->leftChild);
         if (ptr->rightChild)
            addq(ptr->rightChild);
      }
      else break;
   }
}
```

Program 5.5: Level-order traversal of a binary tree

② 정적인 환형큐(circular queue)

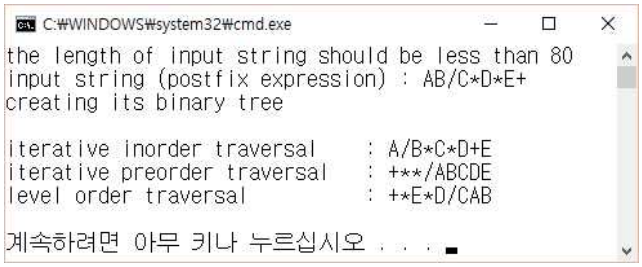
```
void addq(element item)
{/* add an item to the queue */
   rear = (rear+1) % MAX_QUEUE_SIZE;
   if (front == rear)
      queueFull(); /* print error and exit */
   queue[rear] = item;
}
```

Program 3.7: Add to a circular queue

```
element deleteq()
{/* remove front element from the queue */
   element item;
      if (front == rear)
         return queueEmpty(); /* return an error key */
      front = (front+1) % MAX_QUEUE_SIZE;
      return queue[front];
}
```

Program 3.8: Delete from a circular queue

[실행예]



■ 제출 형식

- 솔루션 이름 : DS 14
- 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!)**