자료구조응용

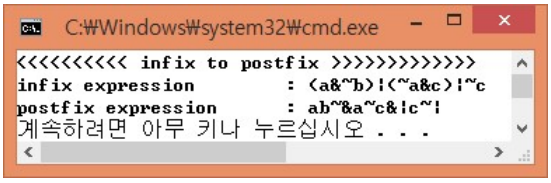
14. 트리: 명제식 이진트리 (20점)

2020.5.4.(월)

1. 괄호를 포함한 중위표현식(infix expression)의 명제식을 파일로부터 입력받아 후위표현식(postfix expression)으로 변환하여 화면 및 파일로 출력하는 프로그램을 작성하라. (5점)

(1) 입출력파일 형식 및 실행 예

- **입력파일(infix.txt)** : (a&~b)|(~a&c)|~c
- 피연산자(Operands) : 알파벳 소문자
- 연산자(Operators) : & | ~
- **출력파일(postfix.txt)** : 변환결과



(2) 구현세부사항

① 주어진 실습문제 소스파일(ds14_1_student.c)을 수정하여 구현한다.

※ 수정 : precedence, isp, icp, getToken, printToken

② logical and(&&), logical or(||), logical not (!)을 &, |, ~으로 대신하여 사용한다.

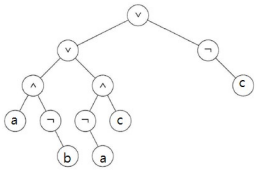
③ postfix expression은 괄호를 포함하지 않는다.

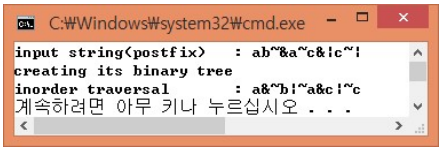
④ 반드시 화면 및 파일(postfix.txt)로 같이 출력되어야 한다.

2. postfix expression의 명제식을 파일로부터 입력받아 이진트리를 구성하여 중위순회(inorder traversal)한 결과를 화면에 출력하라. (5점)

(1) 입력파일, 이진트리 및 실행 예

- **입력파일(postfix.txt)** : ab~&a~c&|c~|
- 피연산자(Operands) : 알파벳 소문자
- 연산자(Operators) : & | ~





(2) 실행순서

① postfix expression의 명제식 (1번 문제의 postfix.txt 활용가능)으로 부터 이진트리를 생성한다.

② 이진트리에 대한 중위 순회를 수행하여 명제식을 출력한다.

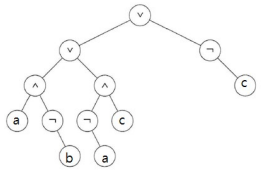
(3) 구현세부사항

① 자료구조응용13. 1번 소스를 수정하거나, 주어진 소스파일(ds14_2_student.c)에서 빈 곳을 채우거나 필요한 부분을 수정하여 구현한다.

3. postfix expression의 명제식을 파일로부터 입력받아 이진트리를 구성하여 변수 값의 가능한 모든 조합에 대해 그 결과를 출력하라. (10점)

(1) 입력파일 및 이진트리

입력파일(postfix.txt)
3
ab~&a~c&|c~|



※ 입력파일의 첫 줄은 변수 개수이며, 피연산자, 연산자 종류는 2번과 같다.

(2) 실행순서

① 입력파일의 postfix expression의 명제식으로 부터 이진트리를 생성한다.

② 변수 값의 가능한 모든 조합에 대해 그 결과를 계산하여 출력한다.

(3) 구현세부사항

※ 노드는 교재의 정의를 참고하여 변경 가능하다.

```
typedef enum {not,and,or,true,false} logical;
typedef struct node *treePointer;
typedef struct node {
        treePointer leftChild;
        logical      data;    // the value of a variable or an operator
        short int    value;   // TRUE/FALSE
        treePointer rightChild;
        } node;
```

```
for (all 2^n possible combinations) {
   generate the next combination;
   replace the variables by their values;
   evaluate root by traversing it in postorder;
   if (root→value) {
      printf(<combination>);
      return;
   }
}
printf("No satisfiable combination\n");
```

Program 5.8: First version of satisfiability algorithm

```
void postOrderEval(treePointer node)
{/* modified post order traversal to evaluate a
    propositional calculus tree */
  if (node) {
    postOrderEval(node→leftChild);
    postOrderEval(node→rightChild);
    switch(node→data) {
      case not:   node→value =
           !node→rightChild→value;
           break;
      case and:   node→value =
           node→rightChild→value &&
           node→leftChild→value;
           break;
      case or:    node→value =
           node→rightChild→value ||
           node→leftChild→value;
           break;
      case true:  node→value = TRUE;
           break;
      case false: node→value = FALSE;
    }
  }
}
```

Program 5.9: Postorder evaluation function

■ 제출 형식

- 솔루션 이름 : DS 14
- 프로젝트 이름 : 1, 2, 3
- 각 소스파일에 주석처리

  “학번 이름”

  “본인은 이 소스파일을 복사 없이 직접 작성하였습니다.”

- 솔루션 정리 후 솔루션 폴더를 “학번.zip”으로 압축하여 과제 게시판에 제출

■ 주의

- 소스 복사로는 실력향상을 기대할 수 없습니다!!!

- 1차 마감 : 5월 5일(화) 자정

- 2차 마감 : 5월 6일(수) 자정(만점의 80%)