

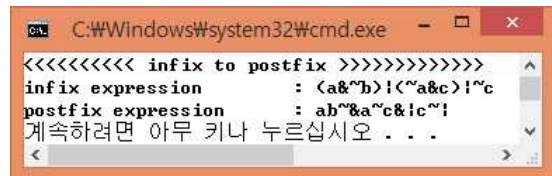
자료구조응용

1. Trees : 명제식 이진트리, heap

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

(1) 입출력파일 형식 및 실행 예

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



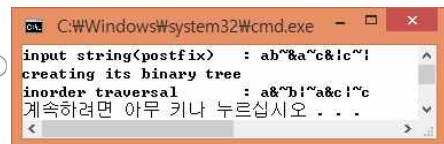
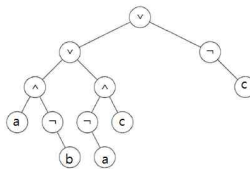
(2) 구현세부사항

- ① 주어진 실습문제 소스파일(`ds1_1_student.c`)을 수정하여 구현한다.
※ 수정 : precedence, isp, icp, getTocken, printTocken
- 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.

(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 주어진 소스파일([ds14_2_student.c](#))에서 빈 곳을 채우거나 필요한 부분을 수정하여 구현한다.

3. 다음 입력파일의 데이터를 사용하여 최대힙(Max Heap)에 대한 실습을 수행한다.

input.txt : 10 40 30 5 12 6 15 9 60

(1) 실행순서

① 파일입력을 받으면서 최대힙을 구성한다.

매 입력마다, 구성된 최대힙의 배열원소를 인덱스 순서대로 출력한다.

② 최대힙의 최대값을 연속으로 원소개수만큼 삭제한다.

매 삭제마다, 재구성된 최대힙의 배열원소를 인덱스 순서대로 출력한다.

(2) 구현 전, 노트에 연습하기

① 위 실행순서 ①의 최대힙이 만들어지는 과정을 최대힙 그림으로 보여라.

② 위 실행순서 ②의 삭제연산 과정을 최대힙 그림으로 보여라.

③ ①②의 각 트리 결과에 대해 배열로 표시해 보라.

※ 각각의 Key 추가/삭제에 대해 결과 트리만 표시하면 됨

※ 노트에 직접 그려서 스캐닝 혹은 사진을 찍어서 보고서에 포함 (가능하다면^^)

(3) 구현 세부사항

```
#define MAX-ELEMENTS 200 /* maximum heap size+1 */
#define HEAP-FULL(n) (n == MAX-ELEMENTS-1)
#define HEAP-EMPTY(n) (!n)
typedef struct {
    int key;
    /* other fields */
} element;
element heap[MAX-ELEMENTS];
int n = 0;
```

```
void push(element item, int *n)
{ /* insert item into a max heap of current size *n */
    int i;
    if (HEAP-FULL(*n)){
        fprintf(stderr, "The heap is full. \n");
        exit(EXIT_FAILURE);
    }
    i = ++(*n);
    while ((i != 1) && (item.key > heap[i/2].key)) {
        heap[i] = heap[i/2];
        i /= 2;
    }
    heap[i] = item;
}
```

Program 5.13: Insertion into a max heap

```

element pop(int *n)
{
    /* delete element with the highest key from the heap */
    int parent, child;
    element item, temp;
    if (HEAP_EMPTY(*n)) {
        fprintf(stderr, "The heap is empty\n");
        exit(EXIT_FAILURE);
    }
    /* save value of the element with the highest key */
    item = heap[1];
    /* use last element in heap to adjust heap */
    temp = heap[(*n)--];
    parent = 1;
    child = 2;
    while (child <= *n) {
        /* find the larger child of the current parent */
        if ((child < *n) && (heap[child].key < heap[child+1].key))
            child++;
        if (temp.key >= heap[child].key) break;
        /* move to the next lower level */
        heap[parent] = heap[child];
        parent = child;
        child *= 2;
    }
    heap[parent] = temp;
    return item;
}

```

Program 5.14: Deletion from a max heap

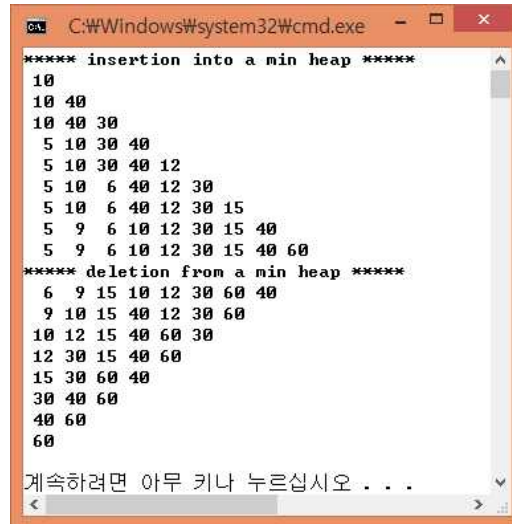
(4) 실행 예

```

C:\Windows\system32\cmd.exe
***** insertion into a max heap *****
10
40 10
40 10 30
40 10 30 5
40 12 30 5 10
40 12 30 5 10 6
40 12 30 5 10 6 15
40 12 30 9 10 6 15 5
60 40 30 12 10 6 15 5 9
***** deletion from a max heap *****
40 12 30 9 10 6 15 5
30 12 15 9 10 6 5
15 12 6 9 10 5
12 10 6 9 5
10 9 6 5
9 5 6
6 5
5
계속하려면 아무 키나 누르십시오 . . .

```

4. 4번 문제를 최소힙(Min Heap)에 대해 동일한 방법으로 연습하고 프로그래밍 하라.



■ 제출 형식

- 제출 : 12월 6일자 채점서버에 소스 업로드(12월 19일까지)
- 실행화면을 캡처하여 한글파일 보고서(DS 1_학번.hwp)에 추가 후 과제에 제출 (12월 19일까지)
- 각 소스파일에 주석처리 추가
“학번 이름”
“본인은 이 소스파일을 다른 사람의 소스를 복사하지 않고 직접 작성하였습니다.”

■ 주의

- **소스복사로는 실력향상을 기대할 수 없습니다!!!**
- 먼저 제출한 학생은 남은 시간 동안 자료구조 관련 개인학습을 하거나 동료를 도와 줄 것
- 수강생 끼리 서로 물어보고 논의를 해도 됨
- 채점서버에 제출하지 않는 경우 최종 점수의 0점으로 처리함
(사용법을 잘 모르겠다면 개인적으로 튜터나 TA를 찾아와서 물을 것)