후위표기식 공개과제

1. 소스코드 + 주석 + 설명

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// 스택 구조체
typedef struct{
    int top; // 스택의 값이 몇 개 인지 알 수 있는 top
    int size;// 스택의 크기를 알 수 있는 size
    char* data; // 동적 배열 생성, 스택의 핵심
}Stack;

// 스택 초기화 함수
void Create(Stack *s)
{
    s->top = -1; // 스택 생성 시 비어있으므로 -1부터 시작
    s->size = 1; // 기본적으로 한 칸
    s->data = (char*)malloc(sizeof(char)); // 한 칸 동적 할당
}

// 스택 포화 여부
int isFull(Stack *s)
{
    // top과 (size-1)이 같다면 가득찬 상태
    if(s->top == s->size -1) return 1;
    else return 0;
}

// 스택 공백 여부
int isEmpty(Stack *s)
{
    // top이 -1부터 시작하니까 top이 -1이라면 공백
    if(s->top == -1) return 1;
    else return 0;
}
```

24번 라인 : top이 -1부터 시작하니까 값이 하나 들어있다면 top은 0이고
size는 1인데 -1을 하면 0이니까
top과 (size-1)이 같으므로 가득찬 상태

```
35
36 // 스택 값 삽입
37 void Push(Stack *s,char value)
38 {
39     // 스택이 가득 찼다면 메모리 재할당
40     if(isFull(s))
41         s->data = (char*)realloc(s->data,sizeof(char) * ++s->size);
42
43     // top은 -1부터 시작하니까 배열의 값을 넣기 위해서는 ++를 먼저 해줘야함
44     s->data[++(s->top)] = value;
45 }
46
47 // 스택 값 추출
48 char Pop(Stack *s)
49 {
50     // 스택이 비어있다면 공백 반환
51     if(isEmpty(s)) return ' ';
52
53     // 비어있지 않다면 top값을 꺼내고 top을 1 줄여준다
54     return s->data[s->top--];
55 }
56
57 // 스택의 탑 값 읽기
58 char Peek(Stack *s)
59 {
60     // 스택이 비어있다면 공백 반환
61     if(isEmpty(s)) return ' ';
62
63     // 비어있지 않다면 top값 반환
64     return s->data[s->top];
65 }
```

41번 라인 : s->size만큼 재할당하면 제자리걸음이니까 ++s->size를 통해서 size의 값을 1늘려주고 재할당

```
66
67 // 연산자 우선순위 반환, (* / 를 + - 보다 먼저 계산하기 위해)
68 int prec(char ch)
69 {
70     switch(ch)
71     {
72         case '(': case ')': return 0;
73         case '+': case '-': return 1;
74         case '*': case '/': return 2;
75     }
76     return -1;
77 }
78
79 // 후위 표기식 변환 함수
80 char* infix_to_postfix(char* arr)
81 {
82     int len = strlen(arr); // 입력받은 수식의 길이 저장
83     // 문자열 하나 씩 읽기 위한 ch
84     //스택에 마지막으로 저장된 연산자를 저장하는 top_op
85     char ch, top_op;
86
87     Stack s; // 변환에 사용 될 스택
88     Create(&s);
89
90     Stack postfix; // 변환 후 후위식을 저장할 스택
91     Create(&postfix);
92
93     for(int i = 0; i < len; i++)
94     {
95         // i가 0부터 len까지 반복하면서 문자열을 하나씩 읽음
96         ch = arr[i];
```

```
 97
 98         switch(ch)
 99         {
100             // ch가 연산자인 경우
101             case '+': case '-': case '*': case '/':
102                 // 스택이 비어있지않고 방금 읽은 값이 스택 top의 연산자 보다 우선순위가 낮다면 반복
103                 while(!isEmpty(&s) && prec(ch) <= prec(Peek(&s)))
104                     Push(&postfix,Pop(&s)); // s의 값을 꺼내서 postfix에 삽입
105                 Push(&s,ch); // ch를 s에 삽입
106                 break;
107             // ch가 열린 괄호인 경우
108             case '(':
109                 Push(&s,ch); // ch를 s에 삽입
110                 break;
111             // ch가 닫힌 괄호인 경우
112             case ')':
113                 top_op = Pop(&s); // 괄호가 닫혔으니 스택의 연산자들을 모두 꺼내야함
114                 while(top_op != '('){ // 그래서 꺼낸 top_op가 열린 괄호가 나올때까지 반복
115                     Push(&postfix,top_op); // 꺼낸 top_op를 postfix에 저장
116                     top_op = Pop(&s); // 새로운 연산자를 pop
117                 }
118                 break;
119             // ch가 숫자인 경우
120             default:
121                 Push(&postfix,ch); // postfix에 숫자 삽입
122                 break;
123         }
124     }
```

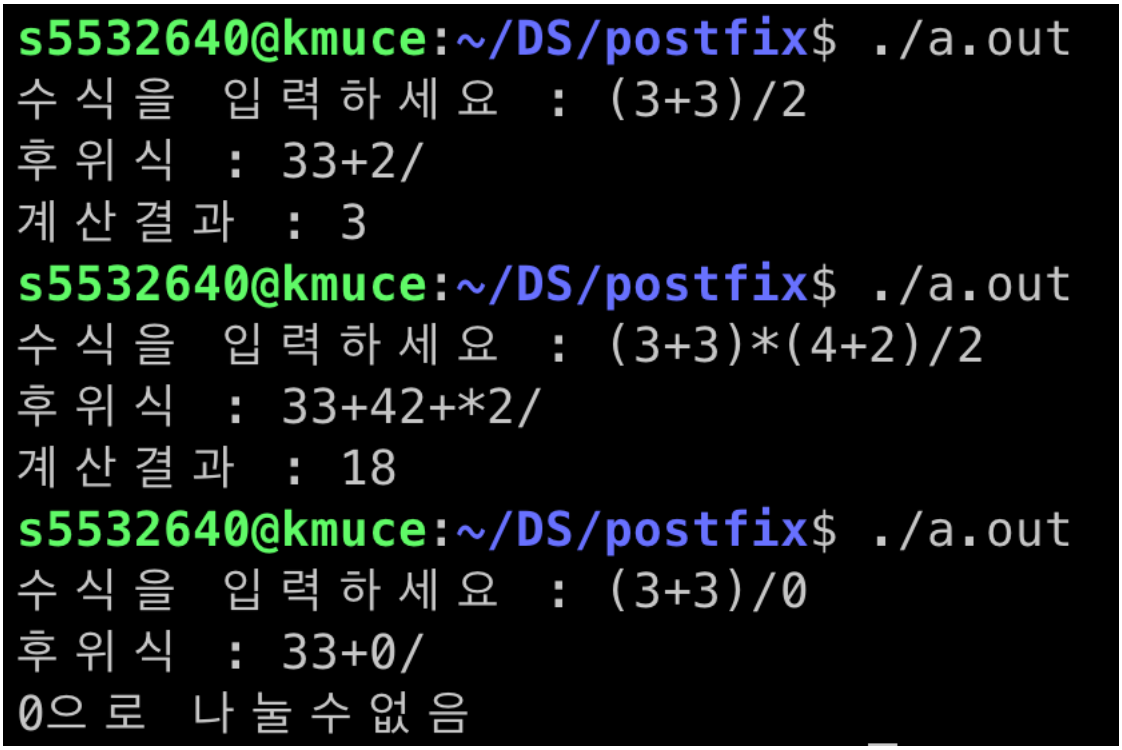
103번 라인 : 이 부분은 동영상 12:05 참고

!isEmpty(&s) ➔ 스택이 비어있지 않다면 ➔ 어떤 연산자가 들어있다면

prec(ch) <= prec(Peek(&s)) ➔ 방금 읽은 값 ch보다 스택에 들어있는 연산자의 우선순위가 더 높다면 ➔ 방금 읽은 건 +고 스택에는 *가 들어있다면 곱하기를 먼저 연산해야 하므로 pop (꺼낸) 후 postfix에 삽입, ch를 스택에 삽입

```
125
126     // 스택 s가 빌 때까지 pop하여 남은 연산자를 postfix에 삽입
127     while(!isEmpty(&s))
128         Push(&postfix,Pop(&s));
129
130     // 저장한 후위표기식을 문자열 형태로 반환
131     return postfix.data;
132 }
133
134 // 후위 표기식 연산
135 void Calc(char *arr)
136 {
137     int len = strlen(arr); // 입력받은 수식의 길이 저장
138     // 문자열 하나 씩 읽기 위한 ch
139     char ch, op1, op2; // 연산을 위한 op1 op2
140     int value; // 문자 숫자를 정수 숫자로 변환
141
142     Stack s; // 연산에 사용될 스택
143     Create(&s);
```

```
144
145     for(int i = 0; i < len; i++)
146     {
147         ch = arr[i];
148         // ch가 연산자가 아니라면, 즉 숫자라면
149         if(ch != '+' && ch != '-' && ch != '*' && ch != '/')
150         {
151             value = ch - '0'; // 정수로 변환하여
152             Push(&s,value); // 스택에 푸쉬
153         }
154         else // ch가 연산자라면
155         {
156             // 스택으로부터 정수 2개를 pop
157             op2 = Pop(&s);
158             op1 = Pop(&s);
159             switch(ch) // 연산자 종류에 따라 op1 op2를 연산하고 다시 푸쉬
160             {
161                 case '+': Push(&s,op1 + op2); break;
162                 case '-': Push(&s,op1 - op2); break;
163                 case '*': Push(&s,op1 * op2); break;
164                 case '/':
165                     if(op1 == 0 || op2 == 0) // 만약 둘 중 하나라도 0이라면 함수 종료
166                     {
167                         printf("0으로 나눌수없음\n");
168                         return;
169                     }
170                      Push(&s,op1 / op2); break;
171             }
172         }
173     }
174
175     printf("%d\n",Pop(&s));
176
177 }
178 int main(int argc,char *argv[])
179 {
180     // 문자열을 입력받고
181     char arr[100];
182     printf("수식을 입력하세요 : ");
183     scanf("%s",arr);
184
185     // 변환한 후위식 출력
186     printf("후위식 : %s\n",infix_to_postfix(arr));
187
188     //변환한 후위식 계산
189     Calc(infix_to_postfix(arr));
190     return 0;
191 }
```



:

2. **코딩 전 과정**
   https://youtu.be/wxfx5AvmL3Y
   스택 0:00 , 후위표기식 6:16

3. **느낀점** : 다른 사람에게 이해 시킨다는 마음으로 임했으며 어떤 생각을 가지고 코딩을 했는지 최대한 표현하기 위해서 노력했다. 생각을 정리하고 말을 하면서 코딩을 하는게 생각보다 쉽지않았고 코딩에 어려움이 있는 분들에게 많은 도움이 되었으면 좋겠다.