SOOKMYUNG WOMEN'S UNIVERSITY

Lab 08

2023학년도 2학기 프로그래밍개론

01, 02, 03분반

숙명여자대학교 소프트웨어학부

데이터 지능 연구실

TA 유사라

4ra@sookmyung.ac.kr

실습

- 과제 1: polish notation (=prefix notation, 전위 표기법) 계산기 프로그램을 만들어보자.

Polish notation (prefix notation; 전위 표기법)

- Polish prefix notation = Polish notation = Prefix notation
- Polish notation은 Reverse Polish notation의 반대에 해당하는 수식 표현 방법
- 즉, 연산자가 피연산자들 앞에 나타난다.
- 예

```
1 + 2 + 3 + 4 + 5 : + + + + 1 2 3 4 5     /* 15 */
(5 - 2) * 3       : * - 5 2 3             /*  9 */
```

Polish notation (prefix notation; 전위 표기법)

- prefix 예시

- * / 15 – 7 + 1 1 3 + 2 + 1 1
- * / 15 – 7 2 3 + 2 + 1 1
- * / 15 5 3 + 2 + 1 1
- * 3 3 + 2 + 1 1
- 9 + 2 + 1 1
- 9 + 2 2
- 9 4

5

Polish notation (prefix notation; 전위 표기법)

- 연산자용 & 피연산자용으로 2개의 스택을 사용하여 쉽게 구현가능
- 피연산자가 연산자보다 나중에 나타나므로 연산자를 일단 연산자용 스택에 저장해 두었다가 피연산자들이 준비가 되면 해당 연산을 수행.
- 피연산자가 준비가 되어 있음
  = 피연산자 스택에 두 개의 숫자 꺼내야함.
  = 연산자를 저장할 당시에 비해 피연산자 스택에 두 개의 숫자가 더 있어야 함.

* 연산자를 저장할 당시 피연산자의 개수를 저장할 필요가 있음 (op_opr_size배열필요)

※ (다음 페이지 pseudo-code 참고)

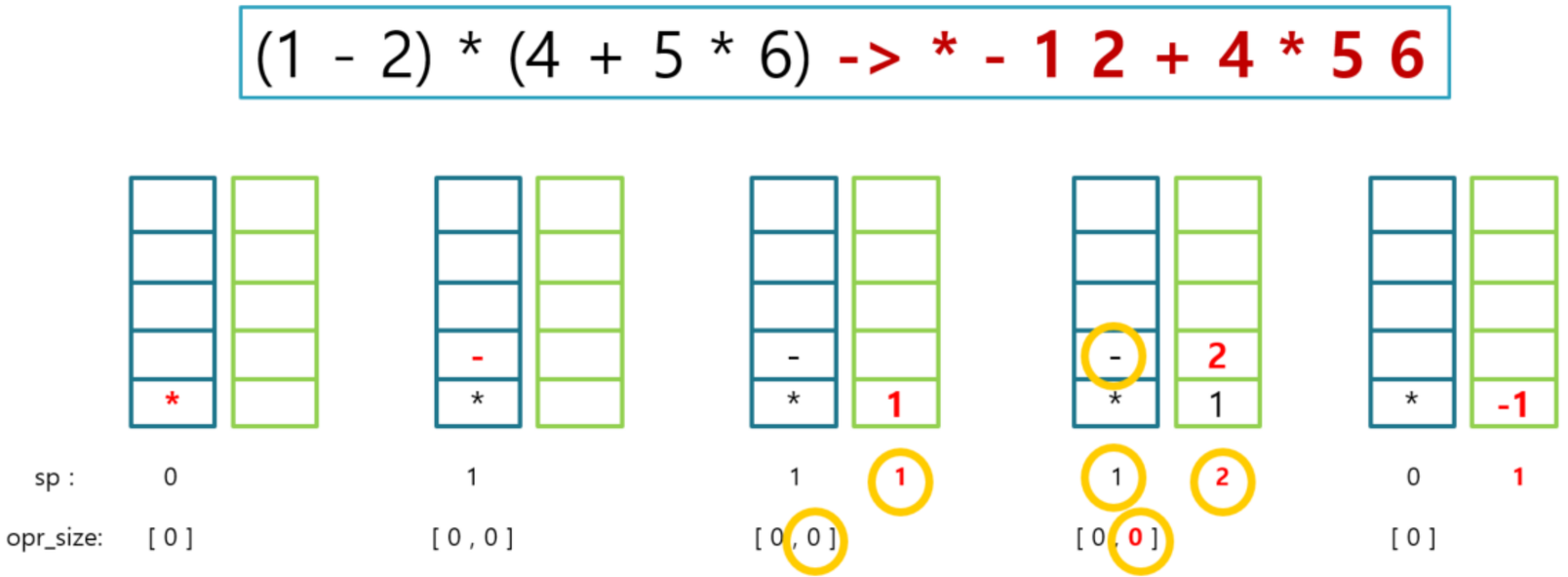
과제 1: prefix notation 계산기 프로그램

- 의사 코드

```
while (next operator or operand is not EOF indicator)
    if (operator)
        push it onto operator stack
            and remember the operand stack size
    else if (newline)
        pop and print top of operand stack
    else if (number)
        push it onto operand stack
        while (two operands are ready for the top operator)
            pop operands
            pop operation and do the operation
            push results onto the operand stack
    else
        error
```

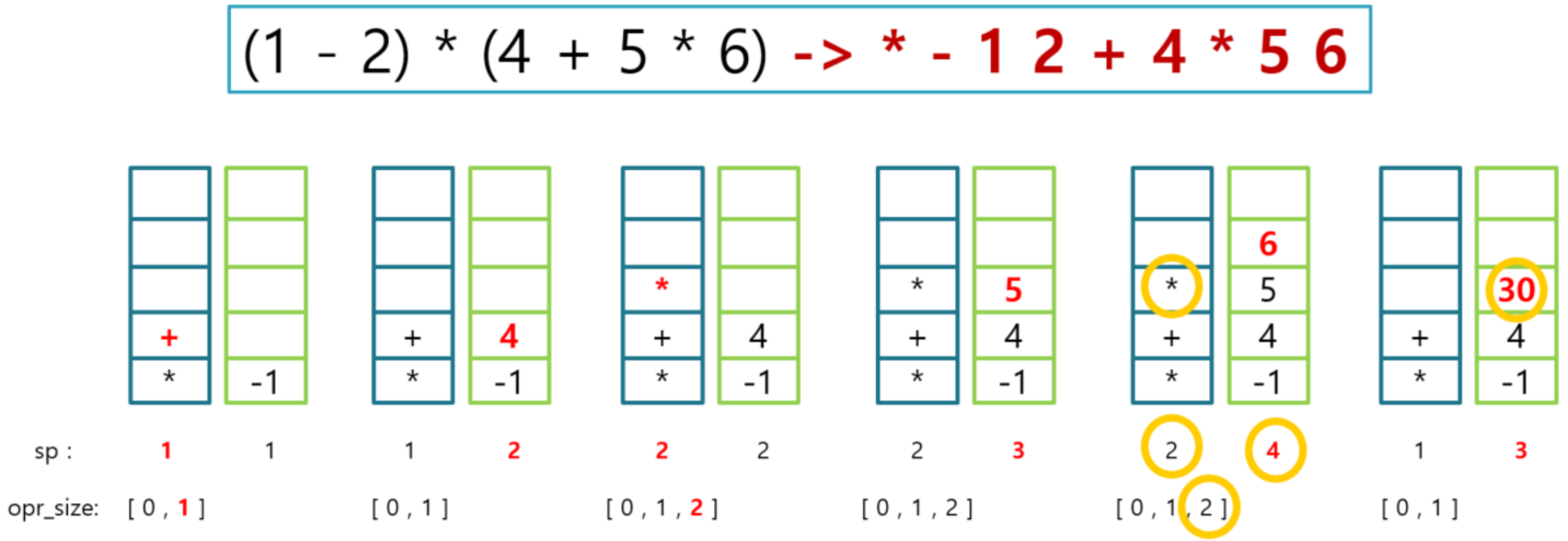
Polish notation (prefix notation; 전위 표기법)

- 연산자용 Stack & 피연산자용 Stack & stack point & operand size check



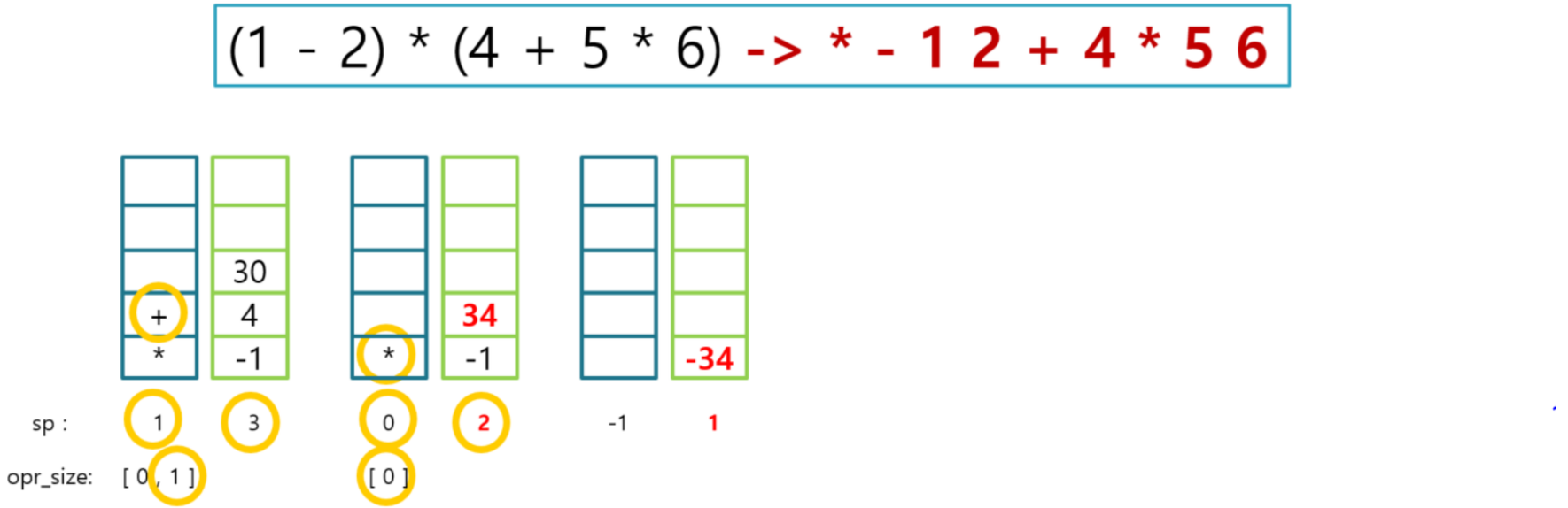
Polish notation (prefix notation; 전위 표기법)

- 연산자용 Stack & 피연산자용 Stack & stack point & operand size check



Polish notation (prefix notation; 전위 표기법)

- 연산자용 Stack & 피연산자용 Stack & stack point & operand size check



Polish notation (prefix notation; 전위 표기법)

- main.c

```
#include <stdio.h>
#include <stdlib.h>
#include "calc.h"
#define MAXOP 100
int main()
{
    int type;
    double op2;
    char s[MAXOP];
    int op_st[100],/* operator stack */
    op_opr_size[100],/* operand stack size at the time of pushing the operator */
    op_sp = -1;/* operator stack position */
    while ((type = getop(s))!= EOF) {
        switch (type) {
        case '+': case '-': case '*': case '/':
            ... (3줄)
            /* 읽어들인 type을 연산자 스택에 넣고, 해당 시점에서 현재 스택에 있는 피연산자 개수를 op_opr_size배열에 저장 */
 case '\n':
  printf("\t%.8g\n", pop());
  break;
```

Polish notation (prefix notation; 전위 표기법)

- main.c

```
case NUMBER:
    push(atof(s));
        while ( (1줄) ) { /* 연산자 존재 && 피연산자 2개 준비되어있는지 확인 */
            int op =(1줄) /* 연산자 스택에서 연산자 꺼내오기 */
            switch (op) {
            case '+':
                push(pop() + pop());
                break;
            case '*':
                push(pop() * pop());
                break;
            case '-':
                op2 = pop();
                push(pop() - op2);
                break;
```

Polish notation (prefix notation; 전위 표기법)

- main.c

```
            case '/':
                    op2 = pop();
                    if (op2 != 0.0)
                        push(pop() / op2);
                    else
                        printf("error: zero divisor\n");
                    break;
                   }
                }
                break;
            default:
                printf("error: unknown command %s\n", s);
                break;
            }
        }
        return 0;
}
```

Polish notation (prefix notation; 전위 표기법)

- getop.c

```
#include <stdio.h>
#include <ctype.h>
#include "calc.h"
int getop(char s[]) {
	int i, c;
	while ((s[0] = c = getchar()) == ' ' || c == '\t')
		;
	s[1] = '\0';
	if (!isdigit(c) && c != '.')
		return c; /* not a number */
	i = 0;
	if (isdigit(c))
		while (isdigit(s[++i] = c = getchar()))
			;
	if (c == '.')
		while (isdigit(s[++i] = c = getchar()))
			;
	s[i] = '\0';
	if (c != EOF)
		ungetc(c, stdin);
	return NUMBER;
}
```

Polish notation (prefix notation; 전위 표기법)

- stack.c : 스택의 현재 크기를 알려주는 size() 함수가 추가되어 있음 (main에서 size함수 이용)

```
#include <stdio.h>
#define MAXVAL 100 /* maximum depth of val stack */
static int sp = 0; /* next free stack position */
static double val[MAXVAL]; /* value stack */
void push(double f) {
        if (sp < MAXVAL)
                val[sp++] = f;
        else
                printf("error: stack full, can't push %g\n", f);
}
double pop(void) {
        if (sp > 0)
                return val[--sp];
        else {
                printf("error: stack empty\n");
                return 0.0;
        }
}
/* return the stack size */
int size(void) {
        return sp;
}
```

Polish notation (prefix notation; 전위 표기법)

- calc.h

```
#define NUMBER '0'

void push(double);
double pop(void);
int size(void);
int getop(char[]);
```

과제 1: prefix notation 계산기 프로그램

- 주어진 소스 파일에서 main.c만을 변경하여 Polish notation을 위한 계산기 프로그램을 작성하라.

EX)	* 4 10	결과) 40
	+ 5.1 * 3 4.2	결과) 17.7
	+ 4 / * 6.2 - 5 2 3	결과) 10.2

- 다른 소스 파일 calc.h, stack.c, getop.c 에서는 단 한 글자도 변경하지 않아야 한다. (+추가 헤더 파일, 추가 소스 파일 생성 금지)
- 변경된 main.c만을 파일이름을 변경하여 과제 결과물로 제출한다.

과제 제출 기한

- 과제 제출 기한
  - 01, 02분반 : 11월 14일 (화) PM 11:59 까지
  - 03분반 : 11월 15일 (수) PM 11:59 까지

- 제출 장소
  - 스노우보드 과제 제출 페이지에 업로드

- 추가 제출 받지 않음

과제 제출 양식

- 소스파일(.c)과 과제보고서(.docx)가 담긴 압축파일(.zip) 제출
  - 압축 파일 이름: Lab08_학번_이름.zip
    'Lab08_학번_이름'으로 된 c파일(main.c 파일명 변경) + 'Lab08_학번_이름'으로 된 .docx파일

- 소스 파일 이름
  - PPT에 제시

- 과제보고서 양식
  - 스노우보드에서 다운로드
  - 실행 결과 화면 캡처한 이미지 첨부
    1. 실행 결과 화면 캡처한 이미지 첨부
    2. 소스 코드
    3. 소스 코드에 대한 설명 (간략하게 3-4줄)

과제 질문

- 조교 메일로 질문 보내기
  - 4ra@sookmyung.ac.kr

- 질문시 주의사항

"• 메일에 반드시 과목, 분반, 전공, 이름, 학번 명시

제목 : 프로그래밍개론 001분반] 2331297 유사라 Lab01 질문 드립니다.

• 몇 번 과제에서 어떤 부분이 막혔는지, 어떤 부분이 문제인지 코드와 함께 설명 첨부

(그냥 코드만 보내면 어디가 문제인지 알 수 없어요)

• 답장이 늦을수도 있으니 이 점 고려하여 미리 질문 (특히 과제 제출 마지막날 유의!)

• 질문 내용을 구체적으로 명확하게 적어 주시기 바랍니다.

• 오류 메시지를 첨부하고 싶을 경우, 오류 캡쳐 화면 + 전체 코드 c 파일을 첨부하여 보내주세요. (코드 캡쳐 사진 X)

• 그 외 출석 등 다른 질문들도 메일로 "