과제 2. 배열/스택/큐

과제 2

- 제출 마감일 : 4월 19일 (금요일) 10:59pm
- eClass 과제방에 제출
- 사용 언어
  - 각 문제에서 명시
- 제출 양식
  - 보고서 (표지 + 문제 내용과 해결 방안 + 결과 캡처)
  - 소스코드 **(별도 파일로 제출) :**
  - **잘못 제출하는 일이 없도록 각별히 유의**
    - **예) .sln, .proj 등 소스코드 없이 프로젝트 파일만 제출하지 않도록 주의**
- 보고서는 PDF 로 제출
- 강의 자료에 있는 코드를 이용하지 않아도 좋다.

타인의 과제를 복사하지 말 것

(1) 문자열 구현하기

C 언어로 구현

- **새로운 방식의 문자열 저장 방식을 제안하려 한다.**
  - 기존의 문자열(C-string)은 문자열의 끝에 널 문자(₩0)를 넣어 문자열의 마지막임을 지정한다.
  - 기존 방식은 문자열의 길이를 확인하는 strlen( )이 O(n) 인데, strlen( )을 자주 써야 하므로 O(n)은 상당히 불만스러운 성능이라 할 수 있다.
  - 새로운 방식을 제안하여 new_string을 만들고, strlen( )이 O(1)이 되게 만들어라. 또 new_string에서 동작하는 strcpy, strcat도 새로 만들하라.
    - 새로운 방식으로 동작하는 strcpy는 new_strcpy( ) 로, strcat는 new_strcat( )로 만들어라.
    - 우리는 strlen( )의 성능만 관심이 있다. 저장 공간이나 다른 함수의 시간 복잡도는 중요하지 않다.

(1) 문자열 구현하기

프로그램의 예(사용 방법을 수정할 수 있다)

– 새로운 문자열의 자료형을 new_string 라고 가정하자.

```
new_string  a, b, c, d;

a = create_str(  " Hello world "  );
b = create_str( " c program " );

new_strcpy(&c, a);       //  a의 내용을 c에 복사한다.
new_strcpy(&d, a);       // a의 내용을 d에 복사한다.
new_strcat(&d, b);       // b의 내용을 d에 붙인다.

printf("%s %s %d %d₩n", new_str(c), new_str(d), new_strlen(c), new_strlen(d) );

// new_str( )은 new_string을 기존의 printf 에서 쓸 수 있게 변환해주는 함수이다.
// new_strlen( )은 문자열의 길이를 O(1)로 반환해주는 함수이다.
```

(2) 계산기

파이썬 언어로 구현

- **계산식을 입력하면 결과를 돌려주는 계산기 프로그램을 작성하라.**
  - 사용되는 연산자는 +, -, *, /, ^^, ( ) 이다.
    - ^^는 지수이다. 3^^4 = 81 ^^의 우선순위는 *보다 높다.
    - ^^ 연산자는 수식 안에서 1회만 사용된다.
  - 입력에 사용되는 수는 자릿수 제한이 없는 실수(정수 포함)이다.
  - 잘못된 수식이라면 어느 위치에 문제가 있는지 알려줘야 한다. 오류는 프로그램이 파악한 가장 첫번째 것만 표시하고, 오류가 있는 수식은 계산될 수 없다.
  - 공백은 입력으로 들어올 수도 있으나, 숫자와 연산자를 구분하는 역할을 할 뿐이고, 계산의 결과에 영향을 주지 않는다.

- **실행 예)**

  2+3*4*4-1

  = 49

(2) 계산기

- **실행 예**

회색 글씨는 참고용이며,
화면에 표시하지 않음

*2.1+(3.2*4)*2.1-12.2*

= 16.78

*2.1+(3.2*4*2.1-12.2*

^ 이 위치에 오류가 있습니다. (대응되는 괄호 없음)

*(11.12+3.4)*2*3.2-12.7*

=80.228

소수점 이하 표시는 최적화해야 한다.
불필요한 자리를 표시하지 않는다.

*2+(3+*4^^2-12*

^ 이 위치에 오류가 있습니다. (+ 뒤에 숫자 또는 (만 올 수 있음))

(닫는 괄호가 없다는 오류를 지적할 수도 있음)

2.1.3+3.2

^ 이 위치에 오류가 있습니다. (잘못된 숫자)

(2) 계산기

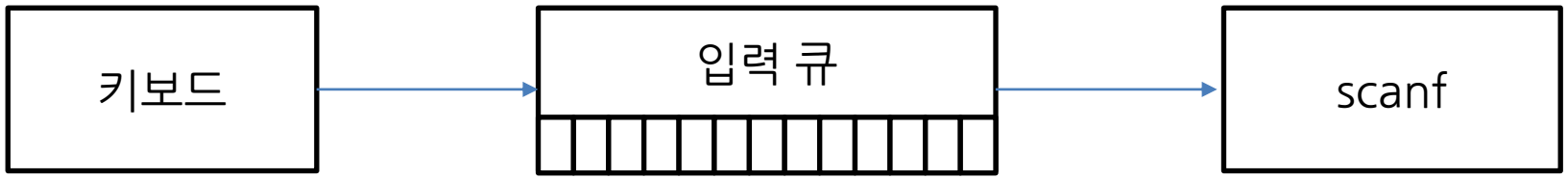
참고할 점

- 구현을 위해서는 스택이 필요하다. 스택은 직접 만들어 사용해도 되고(class stack), 리스트를 활용해도 된다.(파이썬의 리스트가 스택과 동일)
- 괄호는 2회 이상 나타날 수 있다.
  - 2*(3*(4+5)+2)-3
- regex(정규표현)를 쓰지 않는다.
- 계산은 직접 해야 하며, 파이썬 함수인 eval( )을 써서는 안된다.

(3) scanf 만들기

C 언어로 구현

- **scanf는 표준입력 장치로부터 들어오는 문자를 서식에 맞게 변수에 값을 넣는 기능을 가진 함수이다.**
  - 표준 입력은 큐를 사용하는 대표적인 시스템이다.
  - 우리는 scanf와 동일하게 동작하는 함수를 만들어 사용하려 한다.
  - 새로 만들 scanf 함수의 이름을 my_scanf 라고 한다.



(3) scanf 만들기

참고할 점

- scanf는 가변 인자를 사용하는 함수이다. 가변 인자는 파이썬에도 있으며, C에서 가변 인자를 사용하는 방법을 알아야 한다.
- scanf 함수의 동작을 잘 이해해야 한다. scanf 함수가 입력 큐를 사용한다는 점을 기억하라.
- scanf 함수는 사용자가 다수의 데이터를 입력하더라도 서식 문자에 포함된 데이터만 가져온다. 남은 사용자의 입력은 다음 scanf에서 사용한다(입력 큐 이므로).

(3) scanf 만들기

주의할 점

- scanf 내부에서 표준 입력은 fgets( )를 이용한다. 이것은 사용자가 입력한 내용을 큐에 넣는 역할을 한다.
- scanf에는 다수의 서식 문자를 포함해 다양한 기능이 있다.
  - 본인이 구현할 부분을 사전에 정한 후 구현한다.
  - scanf의 모든 기능, 서식 문자를 구현하지 못할 수 있다.
- scanf를 구현하기 위해 내부에 scanf를 쓰거나 scanf 계열 함수 (sscanf, fscanf, wscanf 등)를 써서는 안된다.
- 컴파일러에 포함된 scanf의 소스코드를 그대로 제출해서는 안된다. 본인이 직접 만든 코드를 제출해야 한다.
- 본인이 목표한 내용이 구현되어야 한다. 본인이 목표한 내용이 잘 동작하고 있음을 실행 예시 화면을 통해 증명하라.

(3) scanf 만들기

실행 예

my_scanf(“%d %f”, &I, &f); // 우리가 만든 scanf

만일 사용자가 20 3.14 5.6 이라고 입력했다면 세 번째 데이터인 5.6은 이번 scanf가 아닌 다음번 scanf에서 들어온다.

my_scanf(“%lf”, &df); // 입력 큐에 데이터가 이미 있으므로
// 새로 fgets( )를 수행하지 않고
// 5.6을 가져와 df 변수에 넣는다.