과제 개요 #1: 스택 계산기

1. 개요

연산자와 피연산자로 표현되는 수식은 전위식(prefix), 중위식(infix), 후위식(postfix)로 나타낼 수 있다. 우리는 수식을 표현하고 계산할 때, 이항 연산자를 두 개의 피연산자 사이에 두는 중위식을 사용한다. 이에 비해 컴퓨터에서는 컴파일러에 의해 모든 수식이 후위식으로 변환된 후 연산된다. 후위식이 컴퓨터에서 사용되는 이유는 중위식 수식과 달리 괄호처리가 필요 없고 단순히 왼쪽에서 오른쪽으로 스캔하면서 연산자와 피연산자를 구별하여 일련의 연산을 수행함으로 중위식 계산보다 빠르게 처리할 수 있기 때문이다.

2. 목표

링크드 리스트로 스택을 구현함으로써 스택의 동작과정을 이해하고, 컴퓨터의 계산방식과 유사한 스택 계산기를 설계 및 구현함으로써 자료구조 설계능력과 프로그래밍 능력을 향상한다.

3. 팀 구성

개인별 프로젝트

4. 개발환경

1) OS: Ubuntu 22.04
2) Tools: vscode, vim, gcc

5. 보고서 제출 방법

1) 제출할 파일 **(포맷 형식 반드시 준수)**
   - **보고서([P1]_학번_ver01.pdf)**와 **소스파일([P1]_학번_ver01.c)**을 함께 압축하여 제출
   - 압축파일명: **[P1]_학번_ver01.zip**
2) 제출할 곳
   - 온라인(e러닝) 제출

6. 보고서 양식

1. 과제 개요
2. 설계
3. 구현
   - 각 함수별 기능
4. 테스트 및 결과
   - 테스트 프로그램의 실행 결과를 분석 (캡쳐화면과 함께)
5. **생성형AI 사용 부분 (e.g., ChatGPT, Coilot 등)**
6. 소스코드 (주석포함)

7. 설계 및 구현

1) 구현 내용

사용자로부터 수식을 입력받아 그 수식을 **①검사, ②변환** 및 **③계산**한다. 사용자에게 입력받은 수식을 검사하여 에러가 검출될 시 에러 처리하고 중위식 형태의 수식을 전위식과, 후위식형태로 변환시키고 계산한다.

2) 세부 기능 및 기능별 요구 조건

① 스택은 링크드 리스트를 사용하여 구현
- pop, push, peek, isEmpty, isFull 기본함수 구현
- 자기참조 구조체 사용

② 기본적인 사칙 연산수행
- **+, -, /, *** 기본적인 사칙 연산이 수행 가능해야 함

③ '(', ')' 괄호를 통한 우선순위 연산
- 괄호안에 있는 식이 먼저 계산되야 함

④ 올바른 식인지 검사
- 괄호의 짝이 올바른지 검사
  예) (10.1 * 20) + 4.0) --> (X)
- 식이 올바른지 검사
  예) a + 10.1 * 20 --> (X) 문자열 포함
  예) + 10.1 * 20 --> (X) 부호 포함

⑤ 중위식을 후위식으로 변환하는 함수 구현
- 입력: 중위식 문자열
- 출력: 성공 - 후위식 문자열
  실패 - 오류코드 리턴

⑥ 중위식을 전위식으로 변환하는 함수 구현
- 입력: 중위식 문자열
- 출력: 성공 - 전위식 문자열
  실패 - 오류코드 리턴

⑦ 변환된 후위식을 계산하는 함수 구현
- 입력: 후위식 문자열
- 출력: 성공 - 계산결과
  실패 - 오류코드 리턴

⑧ 소수점 처리
- 예) (중위식) 3.14 + 4 => (후위식) 3.14 4 +

3) 프로그램 실행 예

```
cclab@cclab:~$ ./stackCalc
중위식을 입력하세요: 5*(3.5-2)+4.68/3 [Enter]
전위식: + * 5 - 3.5 2 / 4.68 3
후위식: 5 3.5 2 - * 4.68 3 / +
계산결과: 9.060000
```

4) 설계 구성 요소

* 목표 설정
  - 주어진 요구 조건을 이해하고 명확한 설계 목표를 설정한다.
  - 설계의 목표 및 요구조건을 문서화한다.
* 분석
  - 목표 설정에서 명시한 요구 조건을 분석하고 해결을 위한 기본 전략을 수립한다.
  - 문제 해결을 위한 배경 지식을 이론 강의에서 듣고 개별적으로 학생들이 다양한 경로를 통해 자료를 찾아 분석한다.
* 합성 (자료구조 설계)
  - 분석 결과를 토대로 적절한 자료구조를 도출한다.
  - 도출된 자료구조를 토대로 모듈을 작성한다.
* 제작 (구현)
  - 각 모듈의 입출력을 명확하게 기술한다.
  - 자료구조와 모듈을 C언어로 구현하고 이를 컴파일하여 실행 파일을 만든다.
* 시험 및 평가 (성능평가)
  - 모듈의 입출력 명세에 따라, 다양한 입력 데이터를 작성하고 모듈을 테스트한다.
  - 작성된 실행 파일이 안정적으로 수행되는지 다양하게 테스트하고 이를 평가한다.
* 결과 도출
  - 안정적으로 수행되는 최종 결과(Output)을 캡처하여 최종 결과물은 보고서에 반영한다.

8. 평가 도구

1) 실행 보고서 평가
2) 실행 평가

9. 평가 방법

1) 평가 도구 1) 에 대한 평가 방법

* 소스 코드 분석 및 새로운 모듈의 설계가 제대로 이루어졌는가?
  - 설계 요구 사항을 제대로 분석하였는가?
  - 설계의 제약 조건을 제대로 반영하였는가?
  - 자료구조가 적절히 설계되었는가?
  - 모듈 설계 방법이 적절한가?
* 문서화
  - 소스 코드에 주석을 제대로 달았는가?
  - 설계 보고서가 잘 조직화하고, 잘 쓰였는가?

2) 평가 도구 2) 에 대한 평가 방법

* 요구조건에 따라 올바르게 수행되는가?

10. 기타

1) 지연 제출 시 감점: 1일 지연 시 마다 30% 감점, 3일 지연 후 부터는 미제출 처리
2) 압축 오류 or 파일 누락: 50% 감점 처리
3) copy 발견시 F 처리 (이의제기 안받음)