구현과제1 (40점)

다음의 EBNF로 문법이 정의되는 언어를 위한 해석기를(Recursive-Descent Parser 구현 포함) C/C++, Python으로 각각 구현하시오. (각 언어별로 소스코드 파일 1개씩 총 2개의 소스코드 파일 제출, 구현 언어별 과제 점수 20점씩, 총 40점)

<program> → {<statement>}
<statement> → <var> = <expr> ; | print <var> ;
<expr> → <term> {+ <term> | * <term>}
<term> → <factor> {- <factor>}
<factor> → [-] (<number> | <var> | '('<expr>')')
<number> → <digit> {<digit>}
<digit> → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<var> → <alphabet>{<alphabet>}
<alphabet> → a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z

▸ **입력**
- 실행하면 사용자는 바로 코드를 입력할 수 있도록 구현함. (엔터키 입력까지를 하나의 입력 코드로 인식함)
- 입력 코드의 토큰과 토큰 사이에는 공백 문자가 1개 이상 들어감
- 입력 코드의 <expr>의 결과값은 <number> 범위 숫자로 한정함
- 대입 연산이 수행되지 않은 변수의 값은 0으로 가정함
- 산술연산은 왼쪽 결합성을 가정함 (Left Associativity)
- <var>, <number>의 최대 길이는 10개로 한정함

▸ **출력**
- 문법에 맞는 코드가 입력된 경우에는 다음 코드를 입력 받음. 다음 코드를 입력 받기 전에 출력할 결과가 있다면 출력을 수행함. 문법에 맞지 않는 코드가 입력된 경우에는 "Syntax Error!"를 출력한 후, 다음 코드를 입력 받음
- 아무것도 입력되지 않은 경우에는 프로그램 수행을 종료함

▸ **실행예**

>> x = (12 + 3 ;
>> Syntax Error!

```
>> x = 10 + 5 / 2 ;
>> Syntax Error!
>> k = 3 ; j = 20 ; print k + j ;
>> Syntax Error!
>> a = 5 ; b = 3 ;
>> ab = - ( - ( - ( - ( - 12 ) ) ) ) ;
>> y = - ( 12 - 3 ) ;  print a ;
>> 0
>> b = ( 12 * 2 ) + 30    *  5 ; print b ;
>> 270
>> x = - 12 + 3 ;  print x ;
>> -9
>> x = 10 + 5 - - 2 ; print x ;
>> 17
>> z = 100 * 3 ; y = 42 - 7 ; abc = z - 10 * y + 50 ;  print y ; print abc ;
>> 35 10200
>> x = - 12 + 3 - ( 2 + 8 ) ; y = 10 + 5 * 3 ; print y ; print x ;
>> 45 -19
>> y = 10 * 5 - 3 ; xyz = 5 - 2 + - 8 - 2 ; print y ; print xyz ;
>> 20 -7
```

▸ 제출 요구사항

- 구현 및 테스트를 완료한 소스 코드 파일 2개는 (C/C++, Python 각각 1개씩) 하나의 파일로 압축하여 제출해야 함
- 보고서는 별도의 PDF 파일로 제출해야 함
- 보고서에는 본인이 구현한 코드를 개략적으로 설명해야 하고, 실행 결과 스냅샷은 반드시 포함하여야 함. 보고서에 소스 코드 전체를 포함할 필요는 없음
- 제출 요구사항 미준수 시에는 10% 감점 처리함