컴파일러 과제 1

2022. 3.

과제: 예측 파서 구현

다음은 한 자리 숫자의 사칙연산을 할 수 있는 문법을 해당 문법 번호를 출력하는 action을 포함하도록 확장한 것이다. (단 마지막의 factor 문법은 제외) 이에 대한 예측 파서를 C 언어로 구현하시오. 필요한 경우, 왼쪽 순환 문법을 오른쪽 순환 문법으로 변환하시오. 번역계획으로 추가된 action은 문법 번호를 출력하는 코드를 뜻하므로, 예를 들어 print(1)은 숫자 1을 출력해야 한다. 입력 문자열은 항상 $로 끝난다.

문법:

expr	→ expr + term {print(1)}
expr	→ expr – term {print(2)}
expr	→ term {print(3)}
term	→ term * factor {print(4)}
term	→ term / factor {print(5)}
term	→ factor {print(6)}
factor	→ (expr) {print(7)}
factor	→ 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

제출 내용:

1. 구현을 설명한 보고서 (프로그램을 실행해 본 화면 (3 가지 이상의 경우를 테스트한 결과) 포함) – hwp나 word로 작성
2. 프로그램 소스 - *.c와 *.h 등 꼭 필요한 소스만 포함시킬 것

- 1, 2를 zip으로 묶어 "이름-과제1.zip" 형태로 제출할 것

제출 방법: e-Campus 컴파일러 강좌 과제 제출란에 업로드

제출일: 4월 2일(토) PM 11:50까지

평가방법: 몇가지 예제를 입력하여 수행한 후, 통과한 프로그램에 한 해 채점을 하며, 프로그램 정확도, 가독성, 보고서의 명확성 등으로 평가함.

주의: 인터넷 게시 내용 혹은 다른 사람의 프로그램이나 보고서를 복사 혹은 일부 수정해서 제출할 경우, 과목을 F학점으로 처리함

실행 예제:

입력: 1+2*3$ 출력: 63641	입력: 5*–2$ 출력: 6 error (연산자 *와 -가 연속 입력되어 오류)
입력: 4*(9–2)$ 출력: 66362743	입력: 12+3$ 출력: 63 error (1과 2가 숫자로 2번 연속 입력되어 오류)
입력: 4/2*5$ 출력: 6543	

주의: 아래 정의된 main, match, nexttoken 함수를 활용하여 구현할 것

```
char lookahead;

void main() {
        lookahead = nexttoken();
        exp();
        if (lookahead == '$')
                printf("₩n");
        else
                printf(" error₩n");
}

void match(char token) {
        if (lookahead == token)
                lookahead = nexttoken();
        else {
                printf(" error₩n");
                exit(1);
        }
}

char nexttoken() {
        char c;

        while (1) {
                c = getchar();
                if (c == ' ' || c == '₩t' || c == '₩n' || c == '₩0') continue;
                return(c);
        }
}
```