Assignment 06: List

Date: 2024. 10. 15
Student ID:
Name: 김태훈

1. 여러 자릿수의정수, 길이가 2이상인 형태의 식별자를 인식하도록 yylex를 개선하는 방법[1]

```
void yylex(){
    while(input[pos] ==' ') pos++;
    char ch =input[pos];

    if(ch =='\0'){
        current_token.type =T_EOF;
        return;
}

    if(isdigit(ch)){
        int value =0;

        while(isdigit(input[pos])){
            value =value *10 +(input[pos] -'0');
            pos++;
}
        current_token.type =T_INT;
        current_token.ival =value;
        return;
}

    if(isalpha(ch)){
        int symbol_pos =0;

        while(isalpha(input[pos]) &&symbol_pos <100 -1){
            current_token.sval[symbol_pos++] =input[pos++];
}
        current_token.sval[symbol_pos] ='\0';
        current_token.type =T_SYM;
        return;
}
...
```

한글자만 인식하는 yylex는 input string의 인덱스를 1씩 증가시키면서 숫자가 발견되면 그 글자만

숫자 Data로 파싱하고, 문자 Data가 발견되면 그 문자만 문자 data로 파싱하면 되었다. 그러나 여려 글자를 인식하기 위해서는, 다른 방법이 필요하다.

(1) 두 자릿 수 이상의 숫자 인식

두 자릿 수 이상의 숫자 인식은, 먼저 숫자가 발견되면, 해당 글자가 숫자일 때 동안 인덱스를 1씩 증가 시키면서 발견된 새로운 숫자 C를 아래와 같이 m에 더한다.

$m = 0$
$m = m \times 10 + C$

그리고 더 이상 숫자가 나오지 않으면 current token 데이터에 타입은 int, value에는 m을 대입하여 두 자릿 수 이상의 숫자를 토큰화 한다.

(2) 두 자릿 수 이상의 문자 인식

먼저 문자가 발견되면, 해당 문자가 알파벳일 때 동안 인덱스를 1씩 증가시키면서 current token value에 해당 문자를 대입한다. sval의 크기는 100이여서, 최대 99 길이의 문자를 인식할 수 있다. 더 이상 문자가 나오지 않으면 sval에 저장되어있는 문자열 끝에 null 문자('\0')을 삽입한다.

2. 데이터를 출력하기 위한 기록자(writer)를 별도의 함수로 작성하는 방법과 이를 이용하는 장점

```
void print_data(Data data)
{
    switch(data.tag)
{
    case TINT:
        printf("%d", data.ival);
        break;
    case TSTR:
        printf("%s", data.sval);
        break;
    case TDUO:
        printf("(");
        print_data(data.pval->d[0]);
        printf(" . ");
        print_data(data.pval->d[1]);

        printf(")");
        break;
    case TNIL:
        printf("NIL");
        break;
}
}
```

파싱한 데이터를 바로 출력하지 않고 Data 구조체에 저장하였다가 출력한다. 타입이 정수, 심볼, NIL인 경우에는 각각 정수 값, 심볼 문자열, NIL을 출력하고, pair인 경우에는, pair는 (Data Data)로 구성되어 있으므로, '('를 출력하고 왼쪽 데이터 출력, '.' 출력, 오른쪽 데이터 출력 ,')' 출력

으로 이어진다. 왼쪽 데이터와 오른쪽 데이터 출력 시 다시 writer 함수(여기서는 print_data)를 호출하여 재귀함수로 이루어진다. 데이터 파싱 부분과 출력 부분을 분리하면, 향후 새로운 타입의 데이터를 추가할 때 추가하기 쉬워지고, 출력 방식 변경 시 변경이 쉽다.

3. 결론

숫자, 문자, 리스트를 파싱하고, 리스트는 pair 데이터 타입을 사용하여 저장하였고, 여기서 여러 자릿수의정수, 길이가 2이상인 형태의 식별자를 인식하도록 yylex를 개선하는 방법과 데이터를 출력하기 위한 기록자(writer)를 별도의 함수로 작성하는 방법과 이를 이용하는 장점을 알아보았다. Pair 타입을 사용하면 리스트를 만들 수 있다는 것을 알게 되었고, 데이터 파싱과 출력 부분을 분리하는 것이 유지보수면에서 더 좋다는 것을 알게 되었다.

References

[1] 박두순 저. “컴파일러의 이해”. 한빛미디어. 2020