Assignment 05: Pair

Date: 2024. 10. 10
Student ID: 201924451
Name: 김태훈

1. pair.c 설명

```
Data parse_pair(int *i)
{
    (*i)++;
    Data d1 =parse_data(i);
    while(buf[*i] ==' ')
        (*i)++;
    if(buf[*i] =='.')
    {
        (*i)++;
        while(buf[*i] ==' ')
               (*i)++;
        Data d2 =parse_data(i);
        while(buf[*i] ==' ')
               (*i)++;
        if(buf[*i] ==')')
        {
               (*i)++;
               return cons(d1, d2);
        }
        else
        {
            fputs("Error: expected )\n", stderr);
            exit(1);
        }
     }
    else
    {

        return nill;
    }
}
```

pair 데이터를 제외한 나머지 데이터를 처리하는 parse_data 함수가, 괄호를 만나면 parse_pair에 괄호가 나타난 위치를 전달하며 호출한다. parse_pair함수는 괄호 다음부터 나타나는 데이터를 다

시 parse_data 함수에 넘기고, parse_data 함수는 온점(‘.’) 이전까지 나타나는 데이터를 파싱하여 반환한다. 이렇게 되면 변수 i는 온점 위치이고, 다시 온점 이후의 데이터를 parse_data 함수에 넘겨 parse_data가 괄호 이전까지 파싱한 것을 반환한다. 이제 i는 버퍼에서 2번째 데이터 이후를 가리키고 있고, 오른쪽으로 이동하면서 괄호가 나타나면 온점 이전 데이터와 이후 데이터를 pair로 만들어서 반환하고, 괄호가 나타나지 않는다면 에러를 발생시킨다.

2. 정수, 문자열, 심볼, 짝 데이터 타입을 나타내는 BNF[1]

```
<data> ::= <integer> | <string> | <symbol> |<pair> | <nil>

<integer> ::= <digit> | <digit> <integer>

<string> ::= ‘“’ <character> ‘“’

<symbol> ::= <letter><symbol_character> | <underbar><symbol_character>

<symbol_character> = <letter><symbol_character> | <digit><symbol_character>
                        |<underbar><symbol_character> | ε

<character> ::= <letter><character> | <digit><character>|<white_space><character>| ε

<digit> ::= “0” | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9“

<letter> ::= "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" | "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“|

<nil> ::=“nil”

<pair> ::= ‘(’ <data> ‘ . ’ <data> ‘)’

<white_space>::=“ ”

<underbar> ::=“_”
```

위 BNF는 정수, 문자열, 심볼, 짝 데이터를 나타내는 BNF이다. 간단하게 설명하면 다음과 같다.

(1) <data> 데이터는 <integer>, <string>, <symbol>, <pair> 혹은 <nil>이다.

(2) <integer>는 <digit>로만 구성되어 있다.

(3) <string>은 쌍따옴표 사이에 <character>로 구성된다.

(4) <symbol>은 <underbar>나 <letter>로만 시작하며, 그 뒤에는 <letter>, <underbar>, <digit>가 들

어갈 수 있다.
(5) <character>는 <letter>, <digit>, <white_space>으로만 구성되며, 비어있을 수 있다.
(6) <digit>은 0부터 9까지의 터미널 기호로 바꿀 수 있다.
(7) <letter>는 A부터 z까지의 터미널 기호로 바꿀 수 있다.
(8) <nil>은 “nil”이라는 터미널 기호로 바꿀 수 있다(NULL을 나타냄),
(9) <pair>는 괄호(‘(’,‘)’) 사이에 ‘.’로 구분된 2개의 <data>이다.
(10) <white_space>는 “ ”(즉, 공백) 터미널 기호로 바꿀 수 있다.
(11) <underbar>는 “_” 터미널 기호로 바꿀 수 있다.

즉, 위의 BNF는 데이터 타입에는 정수, 문자열, 심볼, 짝 데이터, NULL을 표현할 수 있으며, 정수는 0-9의 숫자로, 문자열은 문자, 공백, 언더바의 조합이 쌍따옴표로 묶어진 것, 심볼은 쌍따옴표가 없는, 언더바나 문자로 시작하는 (문자, 숫자, 언더바)의 열, 짝 데이터는 괄호로 묶여진 두 개의 데이터 타입을 갖는 데이터이다.

3. 짝 데이터를 사용하여 리스트를 나타내는 방법[2]

빈 리스트는

nil

로 나타낼 수 있다.
데이터가 하나만 있는 리스트는
(data . nil)
짝 데이터로 표현할 수 있다.
여러 개의 데이터가 있는 리스트는,
(data1 . (data2 . (data3 . nil)))
로 표현될 수 있다.
예를 들어 데이터 5개의 정수 리스트 [1,2,3,4,5]는 짝 데이터를 사용하여 아래와 같이 나타낼 수 있다.
(1 . (2 . (3 . (4 . (5 . nil)))))

이 짝 데이터를 트리를 사용하여 구성하면, 왼쪽 노드를 탐색하여 리스트에 저장된 데이터를 가져올 수 있다.

4. 결론

BNF를 사용하여 짝 데이터를 구성할 수 있다는 것을 알았고, 짝 데이터를 사용하면, 리스트도 표현할 수 있음을 알게 되었다. 괄호로 구성된 짝 데이터를 읽는 과정에서, 짝 데이터 안에 있는 짝 데이터를 처리하는 것이 어려웠으며, 이는 데이터를 처리하는 함수가 짝 데이터를 만났을 때 짝 데이터를 처리하는 함수를 호출하는 형식으로 하여 해결하였다.

References

[1] 박두순 저. “컴파일러의 이해”. 한빛미디어. 2020