Programming Languages Assignment6-B

2021312738 소프트웨어학과 김서환

Q1. y함수는 lis라는 list와 s라는 요소를 비교하여 s가 lis안에 존재하는지 확인하는 함수이다. lis가 null이라면 빈 리스트를 반환하고, null이 아니라면 lis의 맨 앞에 있는 요소(Head)와 s를 비교한다. 비교했을때 같다면 그 즉시 lis를 반환하고, 같지 않다면, lis의 Head 제외한 나머지 리스트와 s를 다시 한번 y함수에 집어넣어 재귀적으로 구현된다.

모든 과정을 마쳤을 때, s가 lis에 존재했다면 s와 lis의 Head를 비교했을 당시의 lis 리스트를 반환하고, 만약 존재하지 않는다면 결국에 빈리스트를 출력하게 되는 함수이다.

Q2. x함수는 lis라는 list에 존재하는 #f을 제외한 값의 개수를 반환하는 함수이다. lis가 null인 경우 0을 반환한다. lis의 Head가 list가 아니면 내부에 있는 cond 블록으로 들어가게된다. 해당 블록안에서 lis의 Head가 #f라면 Head를 제외한 나머지 부분을 x함수에 넣어 재귀적으로 구현된다. 만약 #f가 아니라면 (+ 1 (x (cdr lis)))))을 해줌을 통해서 재귀적으로 들어감과 동시에 +1씩 값을 센다. 이는 결국에 나중에 모든 lis의 요소를 돌게 되면 null이 돼서 0을 반환하게 되고, 재귀를 도는 과정에서 +1이 존재(#f가 아닌 값이 존재)한다면 (ex) (+ 1 (+ 1 0))) 이런식으로 #f가 아닌 값에 의해 재귀를 빠져나오면서 +1씩 계산되어 #f을 제외한 값의 개수를 반환할 것이다.

만약에 lis가 list였다면 Head와 Head가 아닌 부분으로 둘을 나눠 재귀적으로 실행하고 그 결과를 + 해준다.

즉, lis안에는 리스트, 요소 둘 다 존재하는 것과 상관없이 #f가 아닌 값의 수를 반환하는 함수이다.

Q3. 다음과 같이 변경하면 x라는 요소의 개수를 반환하는 함수를 만들 수 있다.

```
(define (countElem x aList)
  (cond ((null? aList) 0)
        ((eq? (car aList) x) (+ 1 (countElem x (cdr aList))))
        (#t (countElem x (cdr aList)))))
```

0의 개수를 검사하는 것과 비슷하게 x를 input으로 두고, eq?조건을 통해 x랑 aList의 Head와 비교해서 같다면 +1해주고 다음 요소를 재귀적으로 검사하도록 구현했고, 다르다면 아무것도 하지 않고 다음 요소를 재귀적으로 검사하도록 구현했다.

```
> (countElem 1 '(1 2 5 4 1 3 1))
3
> (countElem 4 '(4 2 4 4 4 3 4 6 7))
5
>
```

Q4. 다음과 같이 작성 List에서 가장 큰 요소와 가장 작은 요소를 반환하는 함수를 만들 수 있다.

```
(define (findMaxMin lst)
  (cond
    ((null? (cdr lst)) (list (car lst) (car lst)))
    (else
     (let ((rest (findMaxMin (cdr lst)))
           (first (car lst)))
       (let ((currentMax (car rest))
             (currentMin (cadr rest)))
         (list
          (if (> first currentMax) first currentMax)
          (if (< first currentMin) first currentMin)))))))
```

((null? (cdr lst)) (list (car lst) (car lst)))은 base case가 되고, rest(나머지)를 재귀적으로 설정하고, first(list의 Head)를 설정하여 입력받은 리스트의 맨 뒤에서부터 비교함을 통해서 가장 큰 값과 가장 작은 값을 찾는다. rest가 재귀적으로 구현되었기 때문에 맨 뒤의 요소부터 비교가 시작된다. (1 2 5 4 1 3 1)로 예를 들면 맨 뒤의 (1)은 base case이므로 (Max Min) = (1 1)을 반환하고, 그 다음 (3 1) -> (Max Min) = (3 1), 그 다음 (1 3 1) -> first : 1이고 (3 1)에서 currentMax가 3, currentMin이 1이므로 first와 비교하여 first가 currentMax

보다 더 크다면 currentMax값 갱신, first가 currentMin보다 더 작다면 currentMin값 갱신 해준다. 이런식으로 맨 앞의 요소까지 확인하여 List 전체에서 가장 큰 값과 가장 작은 값을 반환할 수 있다.

```
> (findMaxMin '(1 2 5 4 1 3 1))
(list 5 1)
```

Q5. 다음과 같이 작성 List에서 입력 받은 요소를 삭제한 결과의 List를 반환하는 함수를 만들 수 있다.

```
(define (deleteatom x lst)
  (cond
    ((null? lst) '())
    ((eq? (car lst) x) (deleteatom x (cdr lst)))
    (else (cons (car lst) (deleteatom x (cdr lst))))))
```

x와 lst를 입력받아서 lst의 Head가 x와 같은지 비교하여 같다면 cdr을 통해 x를 제외한 나머지를 재귀적으로 실행하고(x를 삭제하는 것과 같다.), x와 다르다면 cons를 통해 lst의 Head를 재귀적으로 실행된 결과와 합쳐서 반환해준다.

이렇게 되면 lst안에 있는 x를 전부 삭제한 리스트를 반환해줄 수 있다.