プログラミング言語実験・Scheme 課題レポート

1510151 柳 裕太

2017 年 8 月 9 日

1　課題 1

1.1　map-tree 関数

1.1.1　設計上の留意点

基本的には、ヒントページ (https://www.ied.inf.uec.ac.jp/text/laboratory/scheme/ex-2017/ex-1-hint1.html) のその 1 の雛形に沿って作成した。ヒントページの空欄には、ドット対でない場合にのみ呼び出される処理が入る。ここで、木の枝 (car，cdr で指定) を対象にもう一度 map-tree 関数を再帰呼出し、結果を cons で繋いでドット対を形成している。

Listing 1　map-tree 関数実行例 (対象関数:even?)

```
1  (#f (#t (#f #t)) #t (#f #t #f))
2  ((#f (#t (#f #t)) #t) (#f #t #f))
```

1.1.2　考察

課題の本意通り、map 関数と同等の処理を、S 式によって作成された木構造にも適用することができた。car，cdr で枝を指定することが、処理を実現する最大の決め手になったと考えている。

また、各要素に対する引数に指定された関数の結果を基に、木をまた再構成しているが、こちらも再帰呼出しで結果を問い合わせ、それを基に cons で繋いで引数の木構造を再現できている。これもまた、再帰呼出しを活用するメリットなのではないかと考えている。

1.2　map-tree2 関数

1.2.1　設計上の留意点

基本的には、ヒントページ (https://www.ied.inf.uec.ac.jp/text/laboratory/scheme/ex-2017/ex-1-hint1.html) のその 1 の雛形に沿って作成したものの、空欄部分を変更した。まず map 関数を指定し、使用する関数には lambda 式を使い、「map-tree2 関数に引数として受け取った関数と木の結果を出力する」無名関数を指定し、無名関数への引数としても当該関数が受け取った木を指定している。

Listing 2　map-tree2 関数実行例 (対象関数:even?)

```
1  (#f (#t (#f #t)) #t (#f #t #f))
```

```
2   ((#f (#t (#f #t)) #t) (#f #t #f))
```

1.2.2 考察

map-tree 関数では、わざわざ枝を分割して再帰呼出を行っていた。しかしながら、今回の map-tree2 関数では、単純に map 関数で全ての枝の要素を対象に map-tree2 関数再帰呼出を、lambda 式による無名関数を活用して行っている。これは、再帰呼出を行うケースは、受け取った第二引数が部分木である場合に限定されるからであることが考えられる。

2 課題 2

2.1 get-depth 関数

2.1.1 設計上の留意点

2.1.2 実行例

2.1.3 考察

2.2 get-cousin 関数

2.2.1 設計上の留意点

2.2.2 実行例

2.2.3 考察

3 課題 3

3.1 diff 関数

3.1.1 設計上の留意点

3.1.2 実行例

3.1.3 考察

3.2 tangent 関数

3.2.1 設計上の留意点

3.2.2 実行例

3.2.3 考察

3.3 diff2 関数

3.3.1 設計上の留意点

3.3.2 実行例

3.3.3 考察