微分問題

1. 関数 d

式 E を変数 x で微分する。

$\frac{dE}{dx}$ は関数 d を使い、(d x E)と記述する。式 E は前置記法で表す。

例

式	関数 d による記述
$\frac{d}{dv}(u+v+w)$	(d 'v '(+ u v w))
$\frac{d}{dv}v(u+v+w)$	(d 'v '(* v (+ u v w)))

2. 関数 d の定義

```
(define  (d  x  E)
      (cond  ((const?  E) (diff-constant  x  E))
                          式Eが定数であるとき(diff-constant  x  E)を評価する
             ((variable?  E)     (diff-variable  x  E))
                          式Eが変数であるとき(diff- variable  x  E)を評価する
             ((sum?  E)   (diff- sum  x  E))
                          式Eが加算式であるとき(diff-sum  x  E)を評価する
             ((product?  E)      (diff-product  x  E))
                          式Eが乗算式であるとき(diff-product  x  E)を評価する
             (else  (error  "d:cannt parse"  E)) ))
                          以上のどれでもないとき「d は式 E を理解できない」と表示する
```

ここで以下の述語や関数が登場したが、その詳細は後に定義する。

分類	述語名、関数名	説明
述語	(constant? E)	式 E が定数であるとき#t
	(variable? E)	式 E が変数であるとき#t
	(sum? E)	式 E が加算式であるとき#t
	(product? E)	式 E が乗算式であるとき#t
関数	(diff-constant x E)	式 E(定数)を x で微分する
	(diff-variable x E)	式 E(変数)を x で微分する
	(diff-sum x E)	式 E(加算式)を x で微分する
	(diff-product x E)	式 E(乗算式)を x で微分する
	(error X)	画面に X を表示する

3. 式Eが定数であるとき
「式Eが定数である」とは、「式Eが数値である」と同値であるので、述語constant?と関数diff-constantはつぎのように定義する。

```
(define  constant?  number?)
```
constant?を述語number?に束縛し、新しい述語とする。
(nubmer? x)はxが値であるとき#tとなる。

```
(define  (diff-constant  x  E)  0)
```
式Eが定数であるとき $\frac{dE}{dx}=0$

4. 式Eが変数であるとき
「式Eが変数である」とは、「式Eが記号(シンボル)である」と同値であるので、述語variable?と関数diff-variableはつぎのように定義する。

```
(define  variable?  symbol?)
```
variable?を述語symbol?に束縛し、新しい述語とする。
(symbol? x)はxが記号(シンボル)であるとき#tとなる。

```
(define  (diff-variable  x  E)
   (if (equal?  x  E) 1  0))
```
変数xと式Eが等しいとき $\frac{dE}{dx}=1$ それ以外は0

例

$\frac{dx}{dx}=1,\quad \frac{dv}{dv}=1,\quad \cdots$ ただし、aはxの関数ではない

$\frac{da}{dx}=0,\quad \frac{dx}{dv}=0,\quad \cdots$ xはvの関数ではない

5. 式Eが加算式であるとき
加算式は(+ □ □)の形をしている。述語sum?と関数diff-sumはつぎのように定義する。

```
(define  (sum?  E)
   (and  (pair?  E)
         (equal?  '+  (car  E)) ))
```
式Eが対(pair)であり、かつ、式Eの第一要素が+であるとき#t
つまり式Eが(+ □ □)の形のとき#t

関数diff-sumおよびこれに関係する関数はつぎのように定義する。

```
(define  (args  E) (cdr  E))
```
加算式から+を除去する
加算式(+ A B)から(A B)を得る

```
(define  (make-sum  x) (cons '+  x))
```
リストxの前に+をつけ、加算式をつくる
リスト(A B)から(+ A B)を得る

```
(define  (diff-sum  x  E)
  (make-sum
    (map (lambda (expr) (d x expr))
      (args E))))
```
(lambda ...)は(d x □)を表す
map関数は(args E)の各要素にラムダ関数を適用し、リストを生成する
make-sum関数はリストの先頭に+をつけ、加算式にする

以上のことからdiff-sumは

$\frac{d}{dx}\{f(x)+g(x)\}=\frac{d}{dx}f(x)+\frac{d}{dx}g(x)$ に相当する処理を行っていることがわかる。