weyl-tests1

February 25, 2025

[1]:
```
(ql:quickload :weyl)
```

```
(WEYL)
```

[2]:
```
(in-package :weyl)
```

```
#<PACKAGE "WEYL">
```

[3]:
```
;;; defines one general variable
(defmacro ge-var (v)
  `(defvar ,v (coerce ',v *general*)))
```

```
GE-VAR
```

[4]:
```
(macroexpand '(ge-var r))
```

```
(PROGN
 (EVAL-WHEN (COMPILE-TOPLEVEL) (%COMPILER-DEFVAR 'R))
 (%DEFVAR 'R (SOURCE-LOCATION) (UNLESS (%BOUNDP 'R) (COERCE 'R *GENERAL*))))
```

[5]:
```
(defun eval-str (s)
  (eval (read-from-string s)))
```

```
EVAL-STR
```

[6]:
```
;;; defines general variables from a list
(defun ge-vars (vl)
  (loop for i in vl
    do (eval-str (format nil "(ge-var ~a)" i))))
```

```
GE-VARS
```

[7]:
```
(defun wtype (obj) (cl::type-of obj))
```

```
WTYPE
```

[8]:
```
(weyli::ge-variables *general*)
```

```
(v.1 x)
```

[9]:
```
(ge-vars '(p q r x y z x_0 x_1 x_2 x_3))
```

```
NIL
```

[10]:
```
(weyli::ge-variables *general*)
```

```
(x_3 x_2 x_1 x_0 z y x r q p v.1 x)
```

[11]:
```
p
```

```
p
```

[12]:
```
(wtype p)
```

```
GE-VARIABLE
```

Describe output on console ...

[13]:
```
(describe (wtype p))
```

```
NIL
```

Inspect goes to console as well (:todo) – end with `q` + 2*ENTER

[16]:
```
(inspect (wtype (* p q)))
```

```
 The variable GE1 is unbound.)
```

[19]:
```
(defvar ge1 (expt p (* p q)))
```

```
GE1
```

[20]:
```
ge1
```

```
p^(q p)
```

[22]:
```
(defvar dge1/dp (deriv ge1 p))
```

```
DGE1/DP
```

[23]:
```
dge1/dp
```

```
(log(p)) q p^(q p) + q p^(q p)
```

[24]:
```
(substitute p q dge1/dp)
```

```
(log(p)) p^(1 + p^2) + p^(1 + p^2)
```

[26]:
```
(deriv (substitute p q dge1/dp) p)
```

```
p^p^2 + (2 (log(p)) p^(2 + p^2) + (1 + p^2) p^p^2) (log(p)) + 2 (log(p)) p^(2 +
p^2) + (1 + p^2) p^p^2
```

[27]:
```
(ge-variable? p)
```

```
T
```

[28]:
```
(ge-variable? u)
```

```
 The variable U is unbound.)
```

[31]:
```
(ge-vars '(u v))
```

```
NIL
```

[32]:
```
;;; make-app-function (todo: wrong in manual: make-applicable-function)
(defvar f1 (weyli::make-app-function '(u v) (+ (* 'u 'v) (* 'u 'u 'u))))
```

```
F1
```

[33]:
```
f1
```

```
(lambda (v.1 v.2) v.1^3 + v.2 v.1)
```

[35]:
```
(defvar df10 (deriv f1 0))
```

```
DF10
```

[36]:
```
(defvar df11(deriv f1 1))
```

```
DF11
```

[37]:
```
(wtype f1)
```

```
APPLICABLE-FUNCTION
```

[38]:
```
(wtype df10)
```

```
APPLICABLE-FUNCTION
```

[39]:
```
(apply f1 '(p q))
```

```
p^3 + q p
```

[40]:
```
(apply (deriv f1 0) '(p q))
```

```
q + 3 p^2
```

[41]:
```
(documentation 'weyli::make-ge-variable 'function)
```

```
Create a variable in a domain.
```

[42]:
```
(documentation 'weyli::coerce 'function)
```

```
Coerce the element into the domain.
```

[43]:
```
(documentation 'weyli::expand 'function)
```

```
Replaces all products of sums in exp by sums of products.
```

[44]:
```
(defun show (out)
  (sb-ext:run-program "/usr/local/bin/aamath"
                      (list  (format nil "~A" out))
```

```
                    :output *standard-output*))
```

```
SHOW
```

Goes to console too :-((TODO)

[45]:
```
(show "a/b+c^x-2")
```

```
#<PROCESS :EXITED 0>
```

[ ]:
```

```