Pensieve header: Implementing U(g0).

Implementing g_0

```
f[1]
```
```
f[1]
```

```
f@1
```
```
f[1]
```

```
1 // f
```
```
f[1]
```

```
huan = 7
```
```
7
```

```
PBWRule = {e → 1, l → 2, f → 3};
B[U@l, U@e] = -(B[U@e, U@l] = U@e);
B[U@f, U@l] = -(B[U@l, U@f] = U@f);
B[U@e, U@f] = -(B[U@f, U@e] = h U[]);
```

```
B[U[e], f // U]
```
```
-h U[]
```

```
Uᵢ_[ℰ_] := ℰ /. {h → hᵢ, t → tᵢ, u_U :> Replace[u, x_ :> xᵢ, 1]};
B[x_, x_] = 0;
B[U[(x_)ᵢ_], U[(y_)ᵢ_]] := B[U[xᵢ], U[yᵢ]] = Uᵢ[B[U@x, U@y]];
B[U[(x_)ᵢ_], U[(y_)ⱼ_]] /; i =!= j := 0;
B[x_, y_] := x ** y - y ** x;
```

```
U₃[h U[r, e, i, n]]
```
```
h₃ U[r₃, e₃, i₃, n₃]
```

```
B[U@e₂, U@f₂]
```
```
-h₂ U[]
```

```
B[U@e₂, U@f₃]
```
```
0
```

```
PrecedesEqual[x, y]
```
```
x ≼ y
```

```
? OrderedQ
```

OrderedQ[h[e_1, e_2, …]] gives True if the e_i are in canonical order, and False otherwise. ≫

```
OrderedQ[{1, 2}]
```

```
True
```

```
OrderedQ[{2, 1}]
```

```
False
```

```
{e, f} /. PBWRule
```

```
{1, 3}
```

```
x_ ≼ y_ := OrderedQ[{x, y} /. PBWRule];
x_ ≺ y_ := ! OrderedQ[{y, x} /. PBWRule];
Simp[ℰ_] := Collect[ℰ, _U, Expand];
```

```
Unprotect[NonCommutativeMultiply];
NonCommutativeMultiply[x_] := x;
0 ** _ = _ ** 0 = 0;
x_ ** U[] := x; U[] ** x_ := x;
(a_ * x_U) ** (b_ * y_U) := If[a b === 0, 0, Simp[a b (x ** y)]];
(a_ * x_U) ** y_ := Simp[a (x ** y)]; x_ ** (a_ * y_U) := Simp[a (x ** y)];
(x_Plus) ** y_ := (# ** y) & /@ x; x_ ** (y_Plus) := (x ** #) & /@ y;
U[x_] ** U[y_] := If[x ≺ y, U[x, y], U[y, x] + B[U@x, U@y]];
U[x_] ** U[y1_, yy__] := If[x ≼ y1, U[x, y1, yy], (U@x ** U@y1) ** U@yy];
U[xx__, xn_] ** U[yy__] := U@xx ** (U@xn ** U@yy);
```

```
UU[l___, x_^n_, r___] := UU[l, Sequence @@ Table[x, {n}], r];
UU[l___, 1, r___] := UU[l, r];
UU[] = U[];
UU[l_, r___] := U[l] ** UU[r];
```

Testing g_0

```
UProducts[{}, 0] = {UU[]};
UProducts[{}, n_Integer] /; n > 0 = {};
UProducts[{x_, xs___}, n_Integer] :=
  Sort@Flatten@Table[UU[x^k] ** u, {k, 0, n}, {u, UProducts[{xs}, n - k]}];
UProducts[xs_List, k_Integer, n_Integer] :=
  UProducts[Flatten@Table[x_j, {x, xs}, {j, k}], n];
UProducts[any__, {n_}] := Flatten@Table[UProducts[any, k], {k, 0, n}];
```

```
B[U@f_1, U@e_1]
```

```
UProducts[{e, l, f}, 2, {3}]
```

```
bas = UProducts[{e, l, f}, 2, {3}];
Table[B[x, y] + B[y, x], {x, bas}, {y, bas}] // Flatten // Union
```

```
bas = UProducts[{e, l, f}, 2, {2}];
Table[
   {x, y, z} = xyz;
   Simp[B[B[x, y], z] + B[B[y, z], x] + B[B[z, x], y]],
   {xyz, Subsets[bas, {3}]}
  ] // Flatten // Union
```

```
bas = UProducts[{e, l, f}, 2, {2}];
Table[
   {x, y, z} = xyz;
   Simp[x ** (y ** z) - (x ** y) ** z],
   {xyz, Subsets[bas, {3}]}
  ] // Flatten // Union
```

Testing CYBE

```
r_i_,j_ := h_i UU[l_j] + UU[f_i, e_j]
```

```
B[r_1,2, r_1,3]
```

```
B[r_1,3, r_2,3]
```

```
B[r_1,2, r_2,3]
```

```
B[r_1,2, r_1,3] + B[r_1,3, r_2,3] + B[r_1,2, r_2,3]
```

Testing YBE

```
UExp[n_Integer, u_] := Module[{t},
   t = U[];
   Simp[t + Sum[t = t ** u / k!, {k, n}]]
  ];
R_i_,j_[n_] := UExp[n, r_i,j];
```

```
UExp[5, U@e_1]
```

```
R_1,2[4]
```

```
With[{n = 2}, Simp[R_1,2[n] ** R_1,3[n] ** R_2,3[n] - R_2,3[n] ** R_1,3[n] ** R_1,2[n]]] // Short
```

```
ToDegree[n_][ℰ_] := Simp[ℰ /. {h_i_ :> ħ h_i, u_U :> ħ^Count[u,f_] u}] /.
   a_. x_U :> Normal[Series[a, {ħ, 0, n}]] * x /. ħ → 1
```

```
With[{n = 2},
 Simp[R_1,2[n] ** R_1,3[n] ** R_2,3[n] - R_2,3[n] ** R_1,3[n] ** R_1,2[n]] // ToDegree[n + 2]]
```

```
With[{n = 3}, Simp[R_1,2[n] ** R_1,3[n] ** R_2,3[n] - R_2,3[n] ** R_1,3[n] ** R_1,2[n]] // ToDegree[n]]
```

The "Internal Multiplication" and Meta-Associativity

```
m[i_, j_, k_][ℰ_] := Simp[ℰ /. {
    u_U :>
     UU @@ Join[DeleteCases[u, x_{i|j}], U @@ Cases[u, x_i :> x_k], U @@ Cases[u, x_j :> x_k]],
    h_{i|j} → h_k
   }]
```

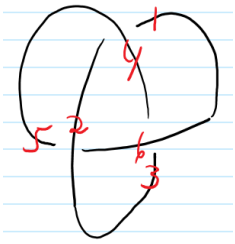
```
UU[e_1, l_4, f_2]
```

```
UU[e_1, l_4, f_2] // m[1, 2, 3]
```

```
UU[e_1, l_4, f_2] // m[2, 1, 3]
```

```
Union@Table[
  (u // m[1, 2, 1] // m[1, 3, 1]) - (u // m[2, 3, 2] // m[1, 2, 1]),
  {u, UProducts[{e, l, f}, 4, {3}]}
 ]
```

The Invariant of the Trefoil




```
With[{n = 2}, R_{4,1}[n] ** R_{2,5}[n] ** R_{6,3}[n] // ToDegree[n]]
```

```
With[{n = 2},
 R_{4,1}[n] ** R_{2,5}[n] ** R_{6,3}[n] // ToDegree[n] // m[1, 2, 1] // m[1, 3, 1] // m[1, 4, 1] //
   m[1, 5, 1] // m[1, 6, 1]]
```

```
With[{n = 3},
 R_{4,1}[n] ** R_{2,5}[n] ** R_{6,3}[n] // ToDegree[n] // m[1, 2, 1] // m[1, 3, 1] // m[1, 4, 1] //
   m[1, 5, 1] // m[1, 6, 1]]
```

Ordering Symbols

```
𝔒[n_, poly_, specs___] := Module[{vs, us},
  vs = Join @@ (First /@ {specs});
  us = Join @@ ({specs} /. (l_ → s_) :> (l /. x_{i_} :> x_s));
  Total[
   CoefficientRules[Normal@Series[poly, {ℏ, 0, n}], vs] /. (p_ → c_) :> c UU @@ (us^p)]]
```