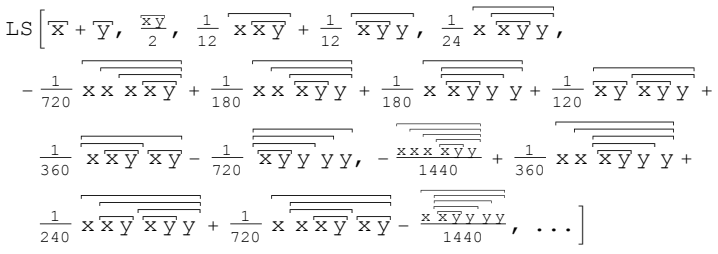
Loading, initializing variables, setting default degree to 6.

(The *Mathematica* packages `FreeLie‘` and `AwCalculus‘` are at ωεβ/WKO4).

```
path = "C:/drorbn/AcademicPensieve/";
SetDirectory[path <> "2015-08/LesDiablerets-1508"];
Get[path <> "Projects/WKO4/FreeLie.m"];
Get[path <> "Projects/WKO4/AwCalculus.m"];
x = LW@"x"; y = LW@"y"; u = LW@"u";
$SeriesShowDegree = 6;
```

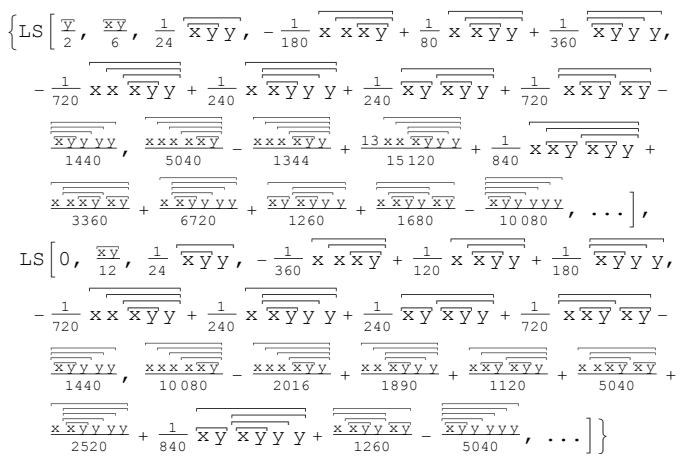
```
FreeLie` implements / extends
 {*, +, **, $SeriesShowDegree, ⟨⟩, ∫, ≡, ad, Ad, adSeries, AllCyclicWords,
  AllLyndonWords, AllWords, Arbitrator, ASeries, AW, b, BCH, BooleanSequence,
  BracketForm, BS, CC, Crop, cw, CW, CWS, CWSeries, D, Deg, DegreeScale,
  DerivationSeries, div, DK, DKS, DKSeries, EulerE, Exp, Inverse, j, J, JA,
  LieDerivation, LieMorphism, LieSeries, LS, LW, LyndonFactorization, Morphism,
  New, RandomCWSeries, Randomizer, RandomLieSeries, RC, SeriesSolve, Support, t,
  tb, TopBracketForm, tr, UndeterminedCoefficients, αMap, Γ, ι, Λ, σ, ħ, ⁀, ⁀}.

FreeLie` is in the public domain. Dror Bar-Natan is committed to
  support it within reason until July 15, 2022. This is version 150814.

AwCalculus` implements / extends
 {*, **, ≡, dA, dc, deg, dm, dS, dΔ, dη, dσ, El, Es, hA, hm, hS, hΔ, hη,
  hσ, RandomElSeries, RandomEsSeries, tA, tha, tm, tS, tΔ, tη, tσ, Γ, Λ}.

AwCalculus` is in the public domain. Dror Bar-Natan is committed to
  support it within reason until July 15, 2022. This is version 150814.
```

BCH[x, y] (* Can raise degree to 22 *)



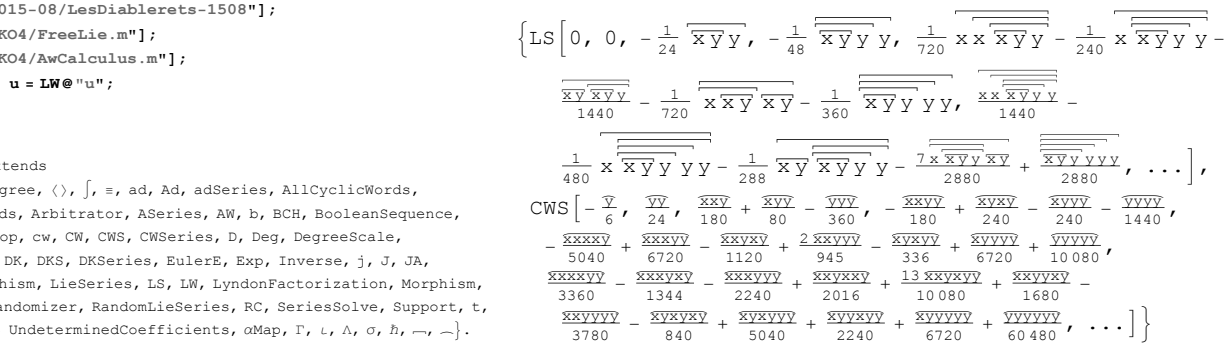
KV Direct.

```
{F = LS[{x, y}, Fs], G = LS[{x, y}, Gs]}; Fs["y"] = 1/2;
SeriesSolve[{F, G},
  ħ^-1 (LS[x + y] - BCH[y, x] ≡ F - G - Ad[-x][F] + Ad[y][G]) ⋀
    div_x[F] + div_y[G] ≡
      1/2 tr_u[adSeries[ad/e^ad-1, x][u] + adSeries[ad/e^ad-1, y][u] -
        adSeries[ad/e^ad-1, BCH[x, y]][u]]];
{F, G} (* Can raise degree to 13 *)
```



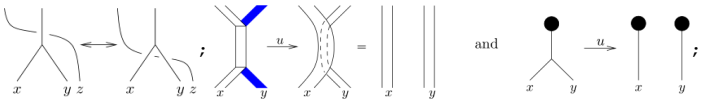
Meaningless calculations.

```
{b[F, G], tr_x[F]}
```



(Also implemented: ∂_λ and derivations in general, `tb`, e^{∂_λ} and morphisms in general, `div`, `j`, Drinfel'd-Kohno, etc.)

The [BND] "vertex" equations.



```
α = LS[{x, y}, αs]; β = LS[{x, y}, βs];
γ = CWS[{x, y}, γs];
V = Es[⟨x → α, y → β⟩, γ];
κ = CWS[{x}, κs]; Cap = Es[⟨x → LS[0]⟩, κ];
Rs[a_, b_] := Es[⟨a → LS[0], b → LS[LW@a]⟩, CWS[0]];
R4Eqn = V ** (Rs[x, z] // dΔ[x, x, y]) ≡ Rs[y, z] ** Rs[x, z] ** V;
UnitarityEqn =
  (V ** (V // dA) ≡ Es[⟨x → LS[0], y → LS[0]⟩, CWS[0]]);
CapEqn = ((V ** (Cap // dΔ[x, x, y]) // dc[x] // dc[y]) ≡
    (Cap (Cap // dσ[x, y]) // dc[x] // dc[y]));
βs["x"] = 1/2; βs["y"] = 0;
SeriesSolve[{α, β, γ, κ},
  (ħ^-1 R4Eqn) ⋀ UnitarityEqn ⋀ CapEqn];
{V, κ}
```

```
SeriesSolve::ArbitrarilySetting : In degree 1 arbitrarily setting {κs[x] → 0}.
SeriesSolve::ArbitrarilySetting : In degree 3 arbitrarily setting {αs[x, y, y] → 0}.
SeriesSolve::ArbitrarilySetting : In degree 5 arbitrarily setting {αs[x, x, x, y, y] → 0}.
General::stop :
  Further output of SeriesSolve::ArbitrarilySetting will be suppressed during this calculation. >>
```