EXAMPLE FILE FOR MERGETEX

PAUL ZINN-JUSTIN

1. Introduction

some basic examples:

```
i1 : R=QQ[x,y]; factor(x^3-y^3)
```

$$o2 = (x-y)\,(x^2+x\,y+y^2)$$

```
o2 : Expression of class Product
i3 : res coker vars R
```

$$o3 = R^1 \xleftarrow{\begin{pmatrix} x & y \end{pmatrix}} R^2 \xleftarrow{\begin{pmatrix} -y \\ x \end{pmatrix}} R^1 \xleftarrow{0} 0$$
$$\qquad\quad 0 \qquad\qquad 1 \qquad\qquad 2 \qquad 3$$

```
o3 : ChainComplex
i4 : OO_(Proj(R/(x^3-y^3)))^{1,2}
```

$$o4 = \mathcal{O}^1_{\mathtt{Proj}\left(\frac{R}{x^3-y^3}\right)}(1) \oplus \mathcal{O}^1_{\mathtt{Proj}\left(\frac{R}{x^3-y^3}\right)}(2)$$

```
o4 : coherent sheaf on Proj (R/(x^3-y^3)), free
i5 : matrix {{1,2},{3,4}}
```

$$o5 = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

```
o5 : Matrix ZZ^2 <-- ZZ^2
```

The code can also be inline: `gcd(1300,75)` but isn't parsed. More:

```
i6 : 318/46
```

$$o6 = \frac{159}{23}$$

```
o6 : QQ
i7 : exp 3.73767
```

$$o7 = 42.0000160321016$$

```
o7 : RR (of precision 53)
```

strings and nets:

```
i8 : "hehe"
o8 = hehe
i9 : ( "haha123456789"
     ||"hoho!@#$%^&*(")
o9 =  haha123456789
      hoho!@#$%^&*(
i10 : {oo,ooo}
```

```
o10 = { haha123456789 , hehe }
      { hoho!@#$%^&*(  }

o10 : List
```

printing:

```
i11 : for i from 1 to 3 do print(i^2+ii)
1 + i
4 + i
9 + i
```

2. Help

```
i12 : help cohomology
o12 =
```

cohomology – general cohomology functor

Synopsis

- Optional inputs:
  - Degree => ..., default value 0,

Description

cohomology – a method name available for computing expressions of the forms HH^i(X) and HH^i(M,N).

If it is intended that i be of class ZZ, M be of class A, and N be of class B, then the method can be installed with

```
cohomology(ZZ, A, B) := opts -> (i,M,N) -> ...
```

See also

- homology – general homology functor
- HH – general homology and cohomology functor
- ScriptedFunctor – the class of all scripted functors

Ways to use cohomology :

- HH^ZZ ChainComplex – cohomology of a chain complex
- HH^ZZ ChainComplexMap – cohomology of a chain complex map
- HH^ZZ Module – local cohomology of a module
- HH^ZZ SheafOfRings – cohomology of a sheaf of rings on a projective variety
- HH^ZZ SumOfTwists – coherent sheaf cohomology module
- "HH^ZZ CoherentSheaf" – see HH^ZZ(ProjectiveVariety,CoherentSheaf) – cohomology of a coherent sheaf on a projective variety
- HH^ZZ(ProjectiveVariety,CoherentSheaf) – cohomology of a coherent sheaf on a projective variety

For the programmer

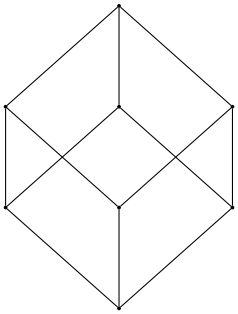
The object `cohomology` is a `method function with options`.

```
o12 : DIV
```

3. Packages

packages that have a `tex` output will work:

```
i13 : needsPackage "Posets";
i14 : booleanLattice 3
```



```
o14 =
o14 : Poset
```

4. Tricky examples

```
i15 : -- some tricky examples
```

A bunch of complicated cases: a multi-line example

```
      f = i -> (
      -- that's dumb
      i+1
      )
o15 = f
o15 : FunctionClosure
```

and another weirder one:

```
i16 : I=ideal 0; f = i -> (
o16 : Ideal of ℤ
      i+1)
o17 = f
o17 : FunctionClosure
```

finally:

```
i18 : a=1;b=2;
i20 : c=3;
```

That last one has no output.

5. Reusing output

The output `o5` is $\left(\begin{smallmatrix}1 & 2\\ 3 & 4\end{smallmatrix}\right)$. The nonexistent output `o18` is .

6. Inputting from external file

Some more code:

```
i21 : -- a test file
      R=QQ[x,y,z]
o21 = R
o21 : PolynomialRing
i22 : poincare ideal(x^2+y^2,x^3+z^3)
o22 = 1 - T^2 - T^3 + T^5
o22 : ZZ[T]
```

7. Changing key/values

```
i23 : "some␣weird␣spacing␣␣and␣string␣style"
o23 = some weird spacing and string style
```