Test

April 11, 2025

Copied from `github.com/agda/agda`

This test file currently lacks module-related stuff.

And interesting uses of shadowing.

```
module Test where

infix  12 _!
infixl 7 _+_ _-_
infixr 2 -_

data ℕ : Set where
  zero : ℕ
  suc : ℕ -> ℕ


_+_ : ℕ -> ℕ -> ℕ
zero + n = n
suc m + n = suc (m + n)

postulate _-_ : ℕ -> ℕ -> ℕ

-_ : ℕ -> ℕ
- n = n

_! : ℕ -> ℕ
zero ! = suc zero
suc n ! = n - n !

record Equiv {a : Set} (_≈_ : a -> a -> Set) : Set where
  field
    refl  : forall x     -> x ≈ x
    sym : {x y : a} -> x ≈ y -> y ≈ x
    _`trans`_ : forall {x y z} -> x ≈ y -> y ≈ z -> x ≈ z

data _≡_ {a : Set} (x : a) : a -> Set where
  refl : x ≡ x

subst : forall {a x y} ->
  (P : a -> Set) -> x ≡ y -> P x -> P y
subst {x = x} .{y = x} _ refl p = p
```

```
Equiv-≡ : forall {a} -> Equiv {a} _≡_
Equiv-≡ {a} =
  record { refl = \_ -> refl
         ; sym = sym
         ; _`trans`_ = _`trans`_
         }
  where
  sym : {x y : a} -> x ≡ y -> y ≡ x
  sym refl = refl

  _`trans`_ : {x y z : a} -> x ≡ y -> y ≡ z -> x ≡ z
  refl `trans` refl = refl


postulate
  String : Set
  Char : Set
  Float : Set

{-# BUILTIN STRING String #-}
{-# BUILTIN CHAR   Char #-}
{-# BUILTIN FLOAT  Float #-}

{-# BUILTIN NATURAL ℕ #-}

data [_] (a : Set) : Set where
  [] : [ a ]
  _::_ : a -> [ a ] -> [ a ]

{-# BUILTIN LIST [_] #-}
-- {-# BUILTIN NIL  []  #-}
-- {-# BUILTIN CONS _::_ #-}

primitive
  primStringToList : String -> [ Char ]

string : [ Char ]
string = primStringToList "∃ apa"

char : Char
char = '∀'

anotherString : String
anotherString = "¬ be\
  \pa"

nat : ℕ
nat = 45

float : Float
float = 45.0e-37
```