Tricks

1. Use `==` to test for equality. In effect, `A==B` is equivalent to `simplify(A-B)==0`.

2. In a script, line breaking is allowed where the scanner needs something to complete an expression. For example, the scanner will automatically go to the next line after an operator.

3. Setting `trace=1` in a script causes each line to be printed just before it is evaluated. Useful for debugging.

4. The last result is stored in symbol `last`.

5. Use `contract(A)` to get the mathematical trace of matrix A.

6. Use `binding(s)` to get the unevaluated binding of symbol s.

7. Use `s=quote(s)` to clear symbol s.

8. Use `float(pi)` to get the floating point value of π. Set `pi=float(pi)` to evaluate expressions with a numerical value for π. Set `pi=quote(pi)` to make π symbolic again.

9. Assign strings to unit names so they are printed normally. For example, setting `meter="meter"` causes the symbol `meter` to be printed as meter instead of m_{eter}.

10. Use `expsin` and `expcos` instead of `sin` and `cos`. Trigonometric simplifications occur automatically when exponentials are used.

11. The following exercise[1] demonstrates some `eval` tricks. Let

$$\psi = \frac{\phi_1 + \phi_2}{2} \exp\left(-\frac{iE_1 t}{\hbar}\right) + \frac{\phi_1 - \phi_2}{2} \exp\left(-\frac{iE_2 t}{\hbar}\right)$$

where ϕ_1 and ϕ_2 are orthogonal and

$$\begin{aligned} A\phi_1 &= a_1\phi_1 \\ A\phi_2 &= a_2\phi_2 \end{aligned}$$

Verify that

$$\langle A \rangle = \int \psi^* A \psi \, dx = \frac{a_1 + a_2}{2} + \frac{a_1 - a_2}{2} \cos\left(\frac{(E_1 - E_2)t}{\hbar}\right)$$

[1] See exercise 4-10 of *Quantum Mechanics* by Richard Fitzpatrick.

```
psi = (phi1 + phi2) / 2 exp(-i E1 t / hbar) +
      (phi1 - phi2) / 2 exp(-i E2 t / hbar)

Apsi = eval(psi, phi1, a1 phi1, phi2, a2 phi2)

A = conj(psi) Apsi

-- delta function trick
A = eval(A, phi1^2, 1, phi2^2, 1)
A = eval(A, phi1, 0, phi2, 0)

A == (a1 + a2) / 2 + (a1 - a2) / 2 cos((E1 - E2) t / hbar)
```

Note: `eval` uses exact pattern matching, not arithmetic matching. For example, `eval(a b c, b c, 1)` does not match, returns *abc*.