from piling up and to make the code clearer, we are better off computing `TIME` and `THETA` from `I` on each iteration. Putting everything together gives:

```
      REAL L
      R = 12.0
      L = 24.0
      X = 0.0
      DO 20 I = 1, 100
         TIME = FLOAT(I)/100.0
         THETA = 2.0 * 3.141593 * TIME
         XNEW = R * (1.0 - COS(THETA)) + L -
     $           L * SQRT(1.0 - (R*SIN(THETA)/L)**2)
         VEL = (XNEW - X)/0.01
         WRITE(2,10) TIME, THETA, XNEW, VEL
   10    FORMAT(4F9.2)
         X = XNEW
   20 CONTINUE
      STOP
      END
```

(We have used `$` as the continuation character, because it is the only standard Fortran character without any other syntactic meaning. It minimizes the chance for confusion, and is likely to cause a visible error if used in the wrong column.)

Since we have saved a hundred function evaluations, we will not worry about computing 2π inside the loop. We also decided to stick with the identifier `L`, instead of changing all occurrences to `AL`. The original problem was stated in terms of `R` and `L`; it is usually safer to remain consistent with this notation than to try to remember the translation all the time. This is one of those unfortunate occasions when standard Fortran notation is at odds with the usage desired. You can argue it either way, but we decided in this case that adding the statement

```
      REAL L
```

is better than renaming the variable. If you get into the habit of declaring *all* variables, the problem doesn't arise at all.

Arithmetic expressions in Fortran and PL/I also differ sometimes from the way we intuitively tend to write them. We are accustomed, in writing algebra, to bind multiplication tighter than division. That is, we assume that if we write

```
A*B/2.0*C
```

it means

```
(A*B)/(2.0*C)   /* WRONG */
```

But in Fortran and PL/I the interpretation is

```
((A*B)/2.0)*C
```

Only by using parentheses or rearranging the computation can we avoid potential confusion.

A more insidious operator ambiguity occurs in this expression from an arctangent routine:

```
TERM = TERM*(-X**2)/DENOM
```