```
# Sample CIF with syntax errors

_date              'Monday 12 April 1999

_cell_length_a     7.514 (3)
_cell_length_b     9.467 (2)

    loop_
     _geom_bond_atom_site_label_1
     _geom_bond_atom_site_label_2
     _geom_bond_distance
                  O1   C2   1.342(4)
                  O1   C5   1.439 (3)

_example_comment
; The purpose of this example is to indicate how vcif
  describes some of the syntax errors it finds.
```

(*a*)

```
ERROR: No data block code before dataname at line 3
ERROR: Single-quoted character string does not
     terminate at line 4
ERROR: Unexpected string ((3)) at line 5
ERROR: Unexpected string ((2)) at line 6
ERROR: Number of loop elements not multiple of
     packetsize at line 15
ERROR: Text field at end of file does not terminate
```

(*b*)

Fig. 5.3.2.1. (*a*) An example CIF with a number of syntax errors and (*b*) the report of the errors produced by *vcif*.

file. Hence the same logical error (the detachment of a standard uncertainty in parentheses from its parent value) is indicated variously as an unexpected text string or as an extraneous loop item, depending on where it occurs in the file. Indeed, in the case of the incorrect number of loop elements, the program makes no attempt to identify which data value or values in the loop might be in error: it simply counts the number of values in a loop and complains when this is not a multiple of the number of data names declared in the loop header.

5.3.2.1.2. *Options to vcif*

A number of options may be supplied as command-line arguments to modify the output from *vcif*.

A more complete account is given of each error on its first occurrence when the program is invoked with the '*-v*' option. The output listing explains in more detail what the breach of syntax is and sometimes suggests how misunderstandings of the file structure result in such breaches (Fig. 5.3.2.2).

Each error message is prefaced by the word 'ERROR' (or occasionally another phrase such as 'WARNING' or 'STAR ERROR'). Three chevrons preface a printout of the beginning of the troublesome line. Then an expanded description of the error is given, prefaced by three asterisks, *on the first occurrence of each distinct error*. In this mode, only the first 20 errors are listed (the assumption is that this mode is best suited to novices, who should identify and correct each error in turn and would not want to be swamped by large numbers of error messages arising from a single error). More errors may be reported by using the '*-e*' command-line option.

The *quiet* option (*vcif -q*) outputs no error messages but instead returns to the calling environment an integer giving the total number of errors found. This option allows scripts or external programs to use *vcif* as a silent test of whether a file has any syntax errors.

```
ERROR: No data block code before dataname at line 3
>>> "_date"
*** A data block MUST begin with a data_something
      declaration.
ERROR: Single-quoted character string does not
      terminate at line 4
>>> "_cell_length_a"
*** The indicated line appears to contain some word
      or words introduced by a single quote, but not
      terminated with a matching single quote.
ERROR: Unexpected string ((3)) at line 5
*** There is an unexpected word or number as
      indicated. This may be because a loop is
      intended but the loop_ keyword has been missed
      out; or a phrase with several words is not
      enclosed in matching delimiting quote marks; or
      a text field (extending over several lines) is
      not properly closed with a final semicolon; or
      a data_ block header has not yet been seen.
ERROR: Unexpected string ((2)) at line 6
>>> "_cell_length_b"
ERROR: Number of loop elements not multiple of
      packetsize at line 15
>>> "_example_comment"
*** A loop_ header defines a list of datanames. The
      values following this header are assigned in
      sequence with the datanames in the header, so
      each packet of information (or row in the table
      of values defined by the loop structure) must
      have the same number of values as there are
      datanames declared in the loop header. Common
      reasons for this error include: omission of a
      value where the associated data are absent
      (insert . or ? as placeholders); numeric values
      where the standard uncertainty (or e.s.d) has
      come adrift from its associated value (e.g.
      10.925 (2)); multi-word phrases or text entries
      that are not properly delimited with quote
      marks or initial semicolons.
ERROR: Text field at end of file does not terminate
>>> ""
*** Is the CIF complete?
```

Fig. 5.3.2.2. Verbose error listing from *vcif* when run with the '*-v*' option on the example of Fig. 5.3.2.1.

A related option, *vcif -b*, counts errors and returns the result as an integer to the calling environment, as in the previous case; but additionally outputs a list of all the data-block codes in the file. While adding nothing to the syntax-checking function of the program, this provides a useful small utility for simply listing data-block names.

Although intended for use with the restricted STAR File syntax permitted for CIF (Chapter 2.2), *vcif* may also be used with the '*-s*' option to check the syntax of CIF dictionary files, which may include save frames. The program does not, however, handle nested loop structures.

The program will flag as an error any line of greater than 80 characters length (the original limit in the CIF version 1.0 specification; see Chapter 2.2), but this behaviour may be overridden with the '*-l*' option. If used, only lines longer than the specified number of characters will be reported and the reports of such lines will be prefaced with the word 'WARNING'. Likewise, the '*-w*' option may be used to override the CIF version 1.0 restriction of data names and data-block codes to 32 characters.

Other options allow the program to write extensive debugging information to a user-specified file, indicating its internal state upon processing each token of input, and to list either a brief summary of how it may be used or its current version number.

Copyright © 2006 International Union of Crystallography