```
           CYCLOPS Check List
           ------------------
   Dictionary data names  = 2244
   New data names in text =    4
    [1]  Dictionary cif_core.dic 2.0.1 data names =  624
    [2]  Dictionary cif_mm.dic 0.9.0 data names = 1620

Data names NOT in Dictionary           Line Numbers

_blat1 . . . . . .                     9   11   94  96
                                      181  183  290  296
_blat2 . . . . . .                    13   15   98  100
                                      185  187  287  293
_dummy_test  . . . . . .               5    7   90  92
                                      177  179  201
_rubbish_here. . . . . .             431

[1]  Dictionary cif_core_2.0.1.dic
[2]  Dictionary cif_mm.dic
                                      Line Numbers

[2] _atom_site.calc_attached_atom     413
[1] = _atom_site_calc_attached_atom   412
[2] _atom_site.calc_flag  . . . . .   410
[1] = _atom_site_calc_flag            409
[2] _atom_site.fract_x  . . . . . .    38   44   50  390
[1] = _atom_site_fract_x              389
[2] _atom_site.fract_y  . . . . . .    39   45   51  394
[1] = _atom_site_fract_y              393
[2] _atom_site.fract_z  . . . . . .    40   46   52  398
[1] = _atom_site_fract_z              397
[2] _atom_site.id   . . . . . .        37   43   49  386
[1] = _atom_site_label                385
[2] _atom_site.thermal_displace_type  406
[1] = _atom_site_thermal_displace_type 405
[2] _atom_site.type_symbol  . . . .   416  420  424  428
                                      434  438  442  450
[1] = _atom_site_type_symbol          415  419  423  427
                                      433  437  441  449
```

[later in the validation output file, showing the transition to unreferenced data names . . .]

```
[1] _symmetry_cell_setting  . . . .   319
[2] = _symmetry.cell_setting          320
[1] _symmetry_space_group_name_H-M    323
[2] = _symmetry.space_group_name_H-M  324
[1] _symmetry_space_group_name_Hall   327   445
[2] = _symmetry.space_group_name_Hall 328   446

[1]  Dictionary cif_core_2.0.1.dic
[2]  Dictionary cif_mm.dic
                                  Names Not Referenced

[2] _atom_site.aniso_B[1][1]
[2] _atom_site.aniso_B[1][1]_esd
[2] _atom_site.aniso_B[1][2]
```

[. . . portion of output omitted . . .]

```
[2] _atom_site.aniso_U[3][3]_esd
[2] _atom_site.attached_hydrogens
[1] = _atom_site_attached_hydrogens
[2] _atom_site.auth_asym_id
[2] _atom_site.auth_atom_id
[2] _atom_site.auth_comp_id
[2] _atom_site.auth_seq_id
[2] _atom_site.B_equiv_geom_mean
[1] = _atom_site_B_equiv_geom_mean
[2] _atom_site.B_equiv_geom_mean_esd
[2] _atom_site.B_iso_or_equiv
[1] = _atom_site_B_iso_or_equiv
[2] _atom_site.B_iso_or_equiv_esd
```

[. . . remainder of output omitted . . .]

Fig. 5.3.4.1. Sample output from *CYCLOPS*. The output has been edited and reformatted slightly to fit into the present column width.

respectively. The special character hyphen ('-') may also be supplied as an argument to '*-i*' or '*-o*' to indicate standard input or standard output.

Finally, if the operating system supports the passing of environment variables to a program, the names of the input file, output file and dictionary file may be passed through the values of `$CYCLOPS_INPUT_TEXT`, `$CYCLOPS_VALIDATION_OUT` or `$CYCLOPS_CHECK_DICTIONARY`, respectively.

5.3.5. File transformation software

This section describes a number of applications that transform an input CIF either to another CIF that contains a subset of the original contents or to other formats suitable for use with general processing tools. (Conversion to other crystallographic data formats is not discussed here.)

5.3.5.1. *QUASAR*: a data extractor

The oldest CIF manipulation program is *QUASAR* (Hall & Sievers, 1993), which was described as the prototype CIF application in the original standard specification paper (Hall *et al.*, 1991). Much of the functionality of *QUASAR* has now been included in the *cif2cif* program (Section 5.3.5.2). However, it remains useful as an application in its own right, and so is briefly described here.

5.3.5.1.1. *Purpose*

The program was designed to read a *request list* of data names, to locate the associated data in an input CIF and to output the data in the order of the request list. The output retains local conformance to CIF syntax rules, but the output file may not be strictly CIF conformant. For example, the same data can be requested multiple times and will be reproduced as often as requested in the output stream, a feature forbidden within a legal CIF.

5.3.5.1.2. *Mode of operation*

Written as a pure Fortran77 application, *QUASAR* requires three data streams: a file containing the request list, an input CIF and an output file. In an operating system such as Unix, it is convenient to attach the request list to the standard input channel; the first two lines of the input stream then take the form `star_arc_`*`infile`* and `star_out_`*`outfile`*, where *infile* and *outfile* are the file names of the input and output files, respectively.

The assignment of an output file may be replaced by a line containing `star_log`. When this is done, the program will test the syntactic validity of the input CIF and write any error messages to the standard output channel. In this mode the program may be used as a syntactic validator, although it is more tolerant of certain syntactic errors than *vcif* (Section 5.3.2.1).

5.3.5.1.3. *The request list*

Fig. 5.3.5.1 is an example request list, intended to highlight some of the special features of the way the program operates. Fig. 5.3.5.2 shows an example CIF against which this request list will be tested; Fig. 5.3.5.3 shows the output. Both figures have been modified slightly to fit on the printed page; they are derived from the sample files distributed with the program.

The request list begins with directives specifying the input and output file names (qtest.cif and qtest.out, respectively). The file may contain comments prefaced by a hash character `#`; this is a useful feature for annotating a request list. Another use for such comments is seen in the standard request list distributed to authors for papers published in *Acta Crystallographica*. Here, data names that are *not* normally published are hidden within the request list as comments and may be activated if they occur in a `publ_manuscript_incl_extra_item` loop within a CIF (see Section 5.7.2.3).

Copyright © 2006 International Union of Crystallography

```
star_arc_qtest.cif
star_out_qtest.out

data_    #<< wild-card block name - accepts first

# request all fractional coord items
   _atom_site_fract_
   _atom_site_label
# capitals to test case insensitivity
   _atom_site_aniso_LABEL
# request something that is not in the CIF
   _dummy
   _atom_site_aniso_U_11

data_P6122
_        #<< this requests all data in this block
```

Fig. 5.3.5.1. An example request list for *QUASAR*.

```
data_P6122

loop_
_atom_type_symbol
_atom_type_oxidation_number
_atom_type_number_in_cell
   # capitals to test case insensitivity
_atom_type_scat_dispersion_REAL
_atom_type_scat_dispersion_imag
_atom_type_scat_source
  S   0  6   .319  .557
        Int_Tab_Vol_III_p202_Tab._3.3.1a
  O   0  6   .047  .032
        Cromer,D.T._&_Mann,J.B._1968_AC_A24,321.
  C   0  20  .017  .009
        Cromer,D.T._&_Mann,J.B._1968_AC_A24,321.
  RU  0  1  -.105   3.296
        Cromer,D.T._&_Mann,J.B._1968_AC_A24,321.

loop_
_atom_site_label
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_U_iso_or_equiv
_atom_site_thermal_displace_type
_atom_site_calc_flag
_atom_site_calc_attached_atom
_atom_site_type_symbol
  s   .20200  .79800  .91667  .030(3)  Uij  ?  ?  s
  o   .49800  .49800  .66667  .02520  Uiso  ?  ?  o
  c1  .48800  .09600  .03800  .03170  Uiso  ?  ?  c

loop_
_atom_site_aniso_label
_atom_site_aniso_U_11
_atom_site_aniso_U_22
_atom_site_aniso_U_33
_atom_site_aniso_U_12
_atom_site_aniso_U_13
_atom_site_aniso_U_23
_atom_site_aniso_type_symbol
  s .035(4) .025(3) .025(3) .013(1) .000 .000 s
```

Fig. 5.3.5.2. Example CIF for demonstrating the use of *QUASAR*.

```
data_P6122

loop_
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_label
  .20200 .79800 .91667 s
  .49800 .49800 .66667 o
  .48800 .09600 .03800 c1

loop_
_atom_site_aniso_label
_dummy           # requested item not present
_atom_site_aniso_U_11
  s ? .035(4)

#                -----end-of-data-block-----

data_P6122

loop_
_atom_type_symbol
_atom_type_oxidation_number
_atom_type_number_in_cell
_atom_type_scat_dispersion_REAL
_atom_type_scat_dispersion_imag
_atom_type_scat_source
  S  0  6  .319  .557
        Int_Tab_Vol_III_p202_Tab._3.3.1a
  O  0  6  .047  .032
        Cromer,D.T._&_Mann,J.B._1968_AC_A24,321.
  C  0 20  .017  .009
        Cromer,D.T._&_Mann,J.B._1968_AC_A24,321.
  RU 0  1 -.105 3.296
        Cromer,D.T._&_Mann,J.B._1968_AC_A24,321.

loop_
_atom_site_label
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_U_iso_or_equiv
_atom_site_thermal_displace_type
_atom_site_calc_flag
_atom_site_calc_attached_atom
_atom_site_type_symbol
  s  .20200 .79800 .91667 .030(3) Uij  ? ? s
  o  .49800 .49800 .66667  .02520 Uiso ? ? o
  c1 .48800 .09600 .03800  .03170 Uiso ? ? c

loop_
_atom_site_aniso_label
_atom_site_aniso_U_11
_atom_site_aniso_U_22
_atom_site_aniso_U_33
_atom_site_aniso_U_12
_atom_site_aniso_U_13
_atom_site_aniso_U_23
_atom_site_aniso_type_symbol
  s .035(4) .025(3) .025(3) .013(1) .000 .000 s

#                -----end-of-data-block-----
```

Fig. 5.3.5.3. Result of running *QUASAR* with the example request list of Fig. 5.3.5.1 on the CIF listed in Fig. 5.3.5.2.

The request list must specify the data block from which the requested data are to be extracted. Multiple data blocks may be requested in the same file. An entry '`data_`' operates as a wild card and indicates that requests should be served from the next data block encountered. In the example above, the first group of requests will be met from the first data block in the CIF; the second set from the data block named 'P6122' (if present).

5.3.5.1.4. *Output from QUASAR*

The body of the request list is a series of data names. Where a data name appears in the CIF, it will be extracted with its associated data value or values. The user need not have prior knowledge of whether a data item occurs in a looped list or not: *QUASAR* will automatically retrieve the matching values and construct a loop header if necessary. However, because the requests are served in the exact order in which they occur in the file, data items in the same list in the input CIF may be extracted into different lists upon output. Although this breaks the semantic association between items grouped in the same list (especially for CIFs described by the DDL2 relational scheme), it is a syntactically valid construction and may be a valuable feature for some processes.

5.3.5.1.4.1. *Treatment of missing data*

When a requested data item is absent from the CIF, *QUASAR* will nevertheless emit a data name with a corresponding value of '?', the conventional CIF value of null type for 'unknown quantity'. A CIF comment is also generated by *QUASAR* to indicate that the entry was missing from the input CIF. If the missing data name is found between data names that have multiple values and that occur in the same looped list, it is assumed that the missing data name should be associated with the same looped list, and it will be emitted in the loop header; the integrity of the list is then satisfied by emitting a column of unknown values. Note how this behaviour differs from that of the generic STAR File extraction utility *Star_Base* (Spadaccini & Hall, 1994), which silently ignores missing data items. However, it is a useful behaviour for applications that depend on finding a specific data item in their processing stream, even where its value is unknown.

5.3.5.1.4.2. *Matching data names*

As with the specification of data-block names, the data names in the request list may have a trailing underscore. Where this is the case, *QUASAR* will retrieve all data items where the data name starts with the specified string. For example, a request for '_atom_site_' will extract *all* data names starting with '_atom_site_'. The special case of an isolated underscore character '_' matches *all* data names present in the current data block.

5.3.5.1.4.3. *Case sensitivity*

The example demonstrates the way in which the application handles the case insensitivity of a requested data item. Data names are converted internally to a lower-case representation, both from the request list and the input CIF. Matches are therefore determined in a case-insensitive manner. However, if a data name is present in the CIF, its original case is retained on output. This permits the computationally irrelevant but cosmetically useful retention of capitalization as used in canonical CIF dictionary definitions. Where the requested data name is absent, the output is all lower-case.

5.3.5.2. *cif2cif*

cif2cif (Bernstein, 1998) is a program built with the *CIFtbx* toolkit (Chapter 5.4) to copy a CIF while checking data names against dictionaries, optionally reformatting numbers to maintain standard uncertainties within a specified range. The output CIF may contain a subset of the data in the original CIF according to a request list, in the manner of *QUASAR* (Hall & Sievers, 1993).

The program was built as a sample application using *CIFtbx* routines and grew out of requirements from several sources.

5.3.5.2.1. *Operation*

5.3.5.2.1.1. *Copying*

In its simplest application, the program copies a CIF from the standard input channel to standard output. The copy is not verbatim (standard utilities of the computer operating system should be used for that purpose), but the output CIF differs from the input only in the following respects: some comments are deleted; lines in the input longer than 80 characters are wrapped to 80 characters or less; white space between tokens may be altered, especially in an attempt to align entries in looped lists in a cosmetically pleasing manner. While none of these changes should affect robust CIF-parsing applications, they are nevertheless useful in imposing a uniform style of presentation for browsing in a text editor or other human-readable framework.

5.3.5.2.1.2. *Constraining standard uncertainties to specified ranges*

Some journals require that standard uncertainties in experimental values should be quoted within a specified range. Typically the standard uncertainty (s.u.) should be quoted as an integer in parentheses, modifying the last place or two of decimals in the experimental data, and with a value between 2 and 19. *cif2cif* permits s.u. values in the ranges 1–9, 2–19 or 3–29, selectable by a command-line switch. The effect of applying the 'rule of 19' would be to change a value of 1.458(1) in the input CIF to 1.4580(10) in the output.

5.3.5.2.1.3. *Dictionary validation*

cif2cif will open one or more CIF dictionary files as it copies the input CIF and identify certain classes of error against the dictionary definitions. The conditions that will raise an error are an unrecognized data name or a wrong data type. The program will also optionally indicate a warning if a data name has been assigned a category different from the leading portion of the data name – this may indicate an inconsistency within the dictionary itself.

5.3.5.2.1.4. *Serving a request list*

cif2cif will extract a subset of the data items contained in a CIF as specified by a request list, in the manner of *QUASAR*. The handling of data names specified in the request list is as described in Section 5.3.5.1.3 above, with the following additional feature. The special string `data_which_contains:` will extract the specified data items from the first data block in which at least one occurs; the block code need not be known in advance.

Some care must be exercised in attempting to extract data from data blocks by context without prior knowledge of the file contents. Consider the following simple example file:

```
data_A
   loop_    _A1
            _A2
             a1 a2 aa1 aa2


data_B
   loop_    _A1
            _B1
             a b aa bb
```

The loop containing **_A1** and **_B1** *cannot* be extracted with a request list of the form

```
data_which_contains:
_A1
_B1
```