IAEA-NDS-0244
Rev. 2024/07

INTERNATIONAL ATOMIC ENERGY AGENCY

NUCLEAR DATA SERVICES

DOCUMENTATION SERIES OF THE IAEA NUCLEAR DATA SECTION

EXFOR Utility Codes

Naohiko Otuka
IAEA Nuclear Data Section, Vienna, Austria

July 2024

IAEA Nuclear Data Section, Vienna International Centre, A-1400 Vienna, Austria

Citation guideline:

When quoting the EXFOR Utility Codes in a publication this should be done in the following way:

N. Otuka, “EXFOR Utility Codes”, report IAEA-NDS-0244 Rev. 2024/07, International Atomic Energy Agency, 2024.

IAEA-NDS-0244
Rev. 2024/07

EXFOR Utility Codes

Naohiko Otuka
IAEA Nuclear Data Section, Vienna, Austria

Abstract

Descriptions are given for a package of utility codes operating on the experimental nuclear reaction data files in the EXFOR format. This program package is written in Python and may be downloaded from the NRDC website (http://nds.iaea.org/nrdc/).

July 2024

Introduction

The EXFOR Utility Codes are written to process EXFOR Entry files and EXFOR/CINDA Dictionary files. Currently, the following 10 codes (Python scripts) are included in this package.

- DIC227: Produce Archive Dictionary 227 from a NUBASE file.
- DICA2J: Convert Archive dictionaries to a JSON Dictionary.
- DICDIS: Prepare Archive and Backup dictionaries for distribution.
- DICJ2A: Convert a JSON Dictionary to Archive dictionaries.
- DICJ2T: Convert a JSON Dictionary to a Transmission dictionary.
- DIRINI: Split an EXFOR library tape into EXFOR entry files.
- DIRUPD: Update the EXFOR entry files with an EXFOR transmission tape.
- MAKLIB: Merge EXFOR entry files into a single library tape.
- SEQADD: Add record identification and bookkeeping information to an EXFOR file.
- SPELLS: Check English spell in free text in EXFOR format.

This document explains how to use these codes. Users need to install Python3 in their environments prior to run these utility codes. Any comments on the use of the codes, including difficulties encountered or any suggestions are welcome.

Option available in all codes

- `-h` Display help information
- `-v` Display the version
- `-f` Never prompt

Acknowledgements

The developer would like to thank Oscar Cabellos and Nicolas Soppera for their comments and proposals.

History (major revisions only)

2023-10-23:

- First release of 4 scripts (DIRINI, DIRUPD, MAKLIB, SEQADD)

2023-11-02:

- First release of IAEA-NDS-0244.

2024-05-03:

- First release of 6 scripts (DIC227, DICA2J, DICDIS, DICJ2A, DICJ2T, SPELLS)

2024-06-25:

- Addition of -c option to MAKLIB.
- Update of DICA2J to implement format changes of Dictionaries 25, 209 and 227 concluded in the NRDC 2024 meeting (C12 and C13).

DIC227

This code reads a NUBASE file and a supplemental input file (compilation of properties of elementary particles and natural elements in the Archive dictionary format), and convert them to Archive Dictionary227.

Input

- NUBASE file (can be an argument following `-i`)
- supplemental input file (can be an argument following `-s`)

Output

- Archive Dictionary 227 file (can be an argument following `-o`)

Option

• `-i` *`NUBASE_file`*	Specify the NUBASE file to read
• `-s` *`supplemental_file`*	Specify the supplemental input file to read
• `-o` *`dictionary_file`*	Specify the archive dictionary file for output

Example

Convert a NUBASE file *nubase.txt* to *dict_arc_new.227* with a supplemental input file *sup.txt*:

```
python3 x4_dic227.py -i nubase.txt -s sup.txt -o dict_arc_new.227
```

DICA2J

This code reads Archive Dictionaries and convert them to a JSON Dictionary.

Input

- dictionary version (can be an argument following `-n`)
- Archive Dictionaries (their directory can be an argument following `-i`)

Output

- JSON Dictionary (its directory can be an argument following `-o`)

Option

• `-n` *`trans_ID`*	Specify the dictionary version (transmission ID)
• `-i` *`directory_inp`*	Specify the directory of Archive Dictionaries to read
• `-o` *`dictionary_out`*	Specify the directory of JSON Dictionary for output

Example

Convert Archive Dictionaries *input/dict_arc.top, input/dict_arc_new.001, input/dict_arc_new.002 etc.* to a JSON Dictionary *output/dict.9128.json*: with a transmission ID *9128*.

```
python3 x4_dica2j.py -n 9128 -i input -o output
```

DICDIS

This code reads Archive and JSON Dictionaries and process them for distribution after removal of records with the alteration flag D (deletion) and updating the year+month field (YYYYMM). At the same time, this code also produces the Backup Dictionary.

Input

- dictionary version (can be an argument following `-n`)
- Archive and JSON Dictionaries (its directory can be an argument following `-i`)

Output

- Archive, Backup and JSON Dictionaries for distribution (its directory can be an argument following `-o`)

Option

• `-n` *`trans_ID`*	Specify the dictionary version (transmission ID)
• `-a` *`directory_archive`*	Specify the directory of Archive dictionaries to read
• `-j` *`directory_json`*	Specify the directory of JSON dictionary to read
• `-o` *`dictionary_out`*	Specify the directory of dictionaries for output

Example

Read Archive and JSON Dictionaries *input/dict_arc.top, input/dict_arc_new.001, input/dict_arc_new.002 etc.* and *json/dict.9128.json*, process them for distribution, and output them under the same names but under the directory *output* with a transmission ID *9128*. At the same time, it also produces the Backup Dictionary *output/dan_back_new.9128.*

```
python3 x4_dicdis.py -n 9128 -a input -j json -o output
```

DICJ2A

This code reads JSON Dictionary and convert it to Archive Dictionaries.

Input

- dictionary version (can be an argument following `-n`)
- JSON Dictionary (its directory can be an argument following `-i`)

Output

- Archive Dictionaries (its directory can be an argument following `-o`)

Option

`-n` *`trans_ID`*	Specify the dictionary version (transmission ID)
`-i` *`directory_inp`*	Specify the directory of JSON Dictionary to read
`-o` *`dictionary_out`*	Specify the directory of Archive Dictionaries for output

Example

Convert JSON Dictionary *input/dict.9128.json* to Archive Dictionaries *output/dict_arc.top, output/dict_arc_new.001, output/dict_arc_new.002 etc.* with a transmission ID *9128*.

```
python3 x4_dicj2a.py -n 9128 -i input -o output
```

DICJ2T

This code reads JSON Dictionary and convert it to a Transmission (TRANS) Dictionary.

Input

- dictionary version (can be an argument following `-n`)
- JSON Dictionary (its directory can be an argument following `-i`)

Output

- TRANS Dictionary (its directory can be an argument following `-o`)

Option

`-n` *`trans_ID`*	Specify the dictionary version (transmission ID)
`-i` *`directory_inp`*	Specify the directory of JSON Dictionary to read
`-o` *`dictionary_out`*	Specify the directory of TRANS Dictionaryfor output

Example

Convert JSON Dictionary *input/dict.9128.json* to TRANS Dictionary *output/trans.9128* with a transmission ID *9128*.

```
python3 x4_dicj2t.py -n 9128 -i input -o output
```

DIRINI

This code reads an EXFOR library tape (e.g., a master) in EXFOR formats with `MASTER`, `LIB` or `REQUEST` record as the first record, splits it into entries, and saves each entry file in an entry storage. It initialises the storage (i.e, namely delete the files in the storage directory) at the beginning of processing.

Input

- library tape (can be an argument following `-l`)

Output

- entry files (e.g., ./entry/a/a0510.txt)
- log file (dirupd.log)

Option

- `-c` Delete (1) trailing blanks in col. 12-66, (2) line sequential number (col.67-80) and (3) N2 of ENDBIB/ENDCOMMON/ENDSUBENT/ENDENTRY.
- `-l` *`library_tape`* Specify the library tape to read
- `-d` *`storage_directory`* Specify the entry storage directory for outputs

Example

Initialise the entry storage directory *entry* by loading the input library tape *lib/library.txt* (without elimination of the record identification and bookkeeping if the input library has them):

```
python3 x4_dirini.py -l lib/library.txt -d entry
```

At the end of processing, one obtains entry files under *entry*/1/, *entry*/2/, etc. and a log file with a new line like

```
Seq. Update date/time           Trans(N1) Trans(N2)  Centre Tape
   0 2023-10-04 00:48:30.849567 0001      20231004          lib/library.txt
```

DIRUPD

This code reads a trans tape (starting from the `TRANS` record), and adds or updates the entry files in the local storage.

Input

- trans tape (can be an argument following `-t`)
- entry storage (can be an argument following `-d`)

Output

- entry files (e.g., ./entry/a/a0510.txt)
- log file (dirupd.log)

Option

- `-c` (Same as DIRINI. Use this option if you use it for DIRINI.)
- `-t` *`trans_tape`* Specify the trans tape to read
- `-d` *`storage_name`* (Same as DIRINI)

Example

Update of the entry storage *entry* by a tape *trans/trans.1234*:

```
python3 x4_dirupd.py -t trans/trans.1234 -d entry
```

At the end of processing, one obtains new and/or updated entry files under *entry*/1/ etc. and a log file with a new line like

```
Seq. Update date/time           Trans(N1) Trans(N2)  Centre Tape
   0 2023-10-04 00:48:30.849567 0001      20231004          lib/library.txt
   1 2023-10-04 00:48:31.725707 1234      20160121   NNDC   trans.1234
```

MAKLIB

This reads and combines the entry files in the entry storage and create a single library tape.

Input

- entry storage (can be an argument following `-d`)

Output

- library tape with `LIB` and `ENDLIB` records as the first and last records.

Option

- `-a` addition of “19” to two-digit year in N2 of ENTRY/SUBENT/NOSUBENT.
- `-c` (Same as DIRINI.)
- `-n` exclusion of dictionaries
- `-d` *`storage_name`* (Same as DIRINI)
- `-l` *`library_tape`* Specify the library tape to create
- `-i` *`tape_ID`* An integer printed at cols 12-22 of the first record

Example

Create a library tape *lib/library.txt* by merging entry files in the storage *entry* with the tape ID 001:

```
python3 x4_maklib.py -d entry -l lib/library.txt -i 0001
```

This operation does not add a new line in the log file.

Note: The format of the output library tape depends on the format of the files in the entry storage. If user maintains the entry files in the storage without record identifications etc. (e.g., with -c option of DIRINI and DIRUPD), then the produced library tape also does not have them. The record identifications can be added by processing the output library tape by SEQADD.

SEQADD

This code adds and/or updates record identifications (cols.67-79) and bookkeeping information such as N1 and N2 of `BIB` and `ENDBIB`. records. (Similar to ORDER developed at NNDC and ZORDER developed at NDS).

Input

- entry, trans or library tape (can be an argument following `-i`)

Output

- entry, trans or library tape with updated record identifications and bookkeeping

Option:

- `-m` do not add "19" to two-digit year in N2 of ENTRY/SUBENT/NOSUBENT, and do not alter N2 of ENDBIB/ENDCOMMON/ENDDATA/ENDSUBENT/ENDENTRY/ENDSUBDICT records.
- `-i` *`input_file`* Specify the input
- `-o` *`input_file`* Specify the output

Example

Create a new trans tape *trans/trans.ord* by adding and updating the record identification and bookkeeping information in *trans/trans.txt*.

```
python3 x4_seqadd.py -i trans/trans.txt -o trans/trans.ord
```

SPELLS

This code checks English spells in the free text field in the EXFOR format. It checks each set of lower characters in free text (i.e., the first word of a sentence is not checked). The default dictionary does not know nuclear physics technical terms, and the user should add the to the dictionary to minimize the output.

Execution requires a Python module **spellchecker**, which may be installed by the following command if pip (a standard Python package manager) is installed in your computer:

```
pip install pyspellchecker
```

Input

- entry, trans or library tape (can be an argument following `-i`)
- dictionary collecting known words (can be an argument following `-d`)

  Example of the dictionary file (a plain text file to be updated by the user by adding more technical terms etc.)

```
atm
deadtime
decoupler
epithermal
fluence
linac
nonuniformity
subentry
```

Output

- log file summarizing typos (can be an argument following `-l`)

Option:

• `-i` *`input_file`*	Specify the EXFOR file to read
• `-d` *`input_file`*	Specify the known word dictionary to read
• `-l log_`*`file`*	Specify the log file name for output

Example

Check spells in an EXFOR entry file *exfor.txt* with a dictionary *x4_spells.dic* and record the checking result in *x4_spells.log*.

```
python3 x4_spells.py -i exfor.txt -d x4_spells.dic -l x4_spells.log
```