Input Format Design and Translator Development for NJOY

C. Emil Hessman
Abstract

Input Format Design and Translator Development for NJOY

C. Emil Hessman

The NJOY Nuclear Data Processing System is a software system used for nuclear data management [1]. In particular, it is used to convert evaluated nuclear data for materials stored in Evaluated Nuclear Data Files (ENDF) [2] into different formats, as well as performing operations on the data.

NJOY is widely used within nuclear data research, and as such, it is important that the system has a user friendly interface. The NJOY input instructions [4] is a non-interactive user interface used for specifying jobs to be run by NJOY. The input instructions are complex and hard to read compared to e.g. a high-level programming language. Working with a large and complex job easily becomes a daunting and error-prone task. Accordingly, there is a need for an improved input format.

In this thesis, a new input format has been designed. In order to make the new input format useable with NJOY, a translator which is able to translate the new input format into the original NJOY input instructions has also been implemented. The results have been verified by a small set of tests.
## Contents

1 Introduction .................................................. 1
  1.1 Background ................................................. 1
  1.2 Problem Description ....................................... 1
  1.3 Objective .................................................. 2

2 NJOY Input Instructions ...................................... 3
  2.1 General Structure .......................................... 3

3 Methodology .................................................. 5
  3.1 Introduction ................................................ 5
  3.2 Designing the New Input Format ............................ 6
  3.3 Building the Translator ..................................... 6
    3.3.1 Lexical Analysis ...................................... 6
    3.3.2 Syntax Analysis ........................................ 7
    3.3.3 Semantic Analysis ...................................... 7
  3.4 Testing ..................................................... 7

4 Implementation ............................................... 8
  4.1 NJOY Input Format (NIF) ................................... 8
  4.1.1 Grammar Definition ..................................... 8
  4.2 NJOY Input Format Translator (nifty) ...................... 10
    4.2.1 Structure of the Translator ............................ 10
    4.2.2 Reserved Keywords ..................................... 10
    4.2.3 The Modules ............................................ 11
  4.3 Translation Verification .................................... 12
  4.4 Translation Efficiency ..................................... 13

5 Results ....................................................... 14
  5.1 NJOY Input Format (NIF) ................................... 14
  5.2 NJOY Input Format Translator (nifty) ...................... 17
  5.3 Translation Verification .................................... 18
  5.4 Translation Efficiency ..................................... 19

6 Discussion .................................................... 20
  6.1 NJOY Input Format (NIF) ................................... 20
  6.2 NJOY Input Format Translator (nifty) ...................... 20
  6.3 Translation Verification .................................... 21
  6.4 Translation Efficiency ..................................... 21

7 Conclusions .................................................. 22

8 Future Work .................................................. 23

References ..................................................... 24
# User Manual

A.1 Structure of nifty ........................................ 26
A.2 Installation ................................................ 27
A.3 Running the Translator ................................. 27
A.4 Settings .................................................... 28

# Test Problems

B.1 Test Problem 01 (tp01) ................................. 29
B.2 Test Problem 02 (tp02) ................................. 37
B.3 Test Problem 03 (tp03) ................................. 46
B.4 Test Problem 04 (tp04) ................................. 51
B.5 Test Problem 05 (tp05) ................................. 55
B.6 Test Problem 06 (tp06) ................................. 57
B.7 Test Problem 07 (tp07) ................................. 66
B.8 Test Problem 08 (tp08) ................................. 70
B.9 Test Problem 10 (tp10) ................................. 75
B.10 Test Problem 11 (tp11) ............................... 79
B.11 Test Problem 12 (tp12) ............................... 88
B.12 Test Problem 13 (tp13) ............................... 93
B.13 Test Problem 14 (tp14) ............................... 97
B.14 Test Problem 17 (tp17) ............................... 99
1 Introduction

1.1 Background

Usability of software systems is important. The usability of a software system is determined, among other things, by its user interface. The user interface of a software system should provide means of interaction between the users and the system such that the desired result can be produced in an easy, elegant, and efficient fashion.

The NJOY Nuclear Data Processing System [1] is a software system used for nuclear data management. In particular, it is used to convert evaluated nuclear data for materials stored in Evaluated Nuclear Data Files (ENDF) [2] into different formats, as well as performing operations on the data. ENDF is a file format used for storing nuclear data which has been produced through an evaluation process.

NJOY is widely used within nuclear data research, and as such, it is important that the system has a user friendly interface. NJOY is currently being used within the MACRO project [3] at the Division of Applied Nuclear Physics, at the Department of Physics and Astronomy at Uppsala University. MACRO is a project that aims at linking reactor parameter uncertainties to uncertainties in nuclear cross sections and nuclear model parameters. This will be done using Monte Carlo-methods, and will require nuclear data processing on a large scale. It has been apparent to the researchers within the project that the non-interactive user interface used for specifying NJOY jobs is not as user friendly as one would like.

1.2 Problem Description

The NJOY input instructions [1, 4] are used for specifying jobs to be run by NJOY. It is a non-interactive user interface in the sense that the entire job needs to be specified prior to feeding the job to NJOY. No further input will be given when the processing of the job has started.

The NJOY input instructions are complex and hard to read compared to e.g. a high-level programming language. For example, algorithm 1 on the following page is a short and simple NJOY job which illustrates what the input instructions look like.
Without consulting the documentation, one might guess that line 4 and 11 are some kind of descriptive titles, which is correct. One might also guess that line 14 terminates the program. However, it is not obvious that line 2 denotes input and output files (each number indicates a specific file) that the system will operate on. It is also hard to deduce that the first number on line 5 denotes the material to be processed, and that the second number denotes the desired temperature in kelvin.

The input instructions can be annotated with descriptive comments, but even then, working with a large and complex job easily becomes a daunting and error-prone task.

1.3 Objective

The NJOY input instructions is not an optimal input format. Therefore, the scope of this thesis has been to design and implement a more user friendly, and readable input format. The design of the new input format could be based on some commonly known existing format that is fitting to the task. The basis could for example be a programming language.

In order to make the new input format useable with NJOY, it has to be translated into the original NJOY input instructions. As such, the scope of this work also included developing an accompanying translator for the new input format.
2 NJOY Input Instructions

The NJOY input instructions is described in reference [1, 4]. A brief summary of the general structure of the input instructions is provided in section 2.1 as a convenience for the reader.

2.1 General Structure

NJOY is composed by a set of modules where each module performs a specific task. Each module has its separate input specification, which defines the expected input instructions for the module.

An NJOY job is an ordered sequence of modules, where the order denotes the execution order. The name of a module is used to denote the start of the corresponding module and its specific input instructions. The input instructions for a module is composed by an ordered sequence of cards. A card is a line with an ordered sequence of values separated by spaces (or commas). A card may be terminated with a slash character to denote the end of the card, but it is not required. Algorithm 2 illustrates the general structure of an NJOY job.

Algorithm 2 General structure of an NJOY job

```
1 module_name
2 value value ... value
3 value value ... value
4 ...
5 value value ... value
6 module_name
7 value value ... value
8 value value/
9 ...
10 value value ... value
11 ...
12 ...
13 ...
14 module_name
15 value value ... value
16 /
17 ...
18 value value ... value
19 stop
```

Line 1 denotes the start of the first module and its specific input instructions. Line 2 denotes the first card for the module declared on line 1. Line 3 through 5 denotes input instructions for successive cards that also belongs to the module declared on line 1. Line 6 through 10 denotes the declaration of another module and its corresponding cards. Successive module definitions, lines 11 through 18, may follow. Line 8 and 16 shows cards that have been terminated with the
slash character, denoting that no more values has been defined for the cards. An NJOY job is terminated by the **stop** instruction as indicated by line 19.

The expected cards in a module and the expected values within a card depends on the specific input specification for the module. The expected type of the values also depends on the input specification for the specific card and module. There are three kinds of types that the values in a card may be defined as: floating-point numbers, natural numbers (integers), and character strings. Character strings are generally required to be terminated by a slash character.

A card may have default values. A default value is a value that does not have to be defined in an NJOY job. If the value is not defined, then a default value will be set by NJOY. Values that may be defaulted are always defined at the end of the card. Hence, the values in a card are organized such that values that must be defined are always defined prior to values that may be defaulted. Default values will be used when a card is terminated by a slash character. For example, according to reference [4], card 3 in the **reconr** module is composed by three values. The last two values have default values. Declaring the card as

```
value/
```

will set the first value to **value** while the last two values will be set to their default values internally by NJOY since no more values were defined in the card.

To conclude, the NJOY input instructions are powerful. It is possible to construct a complex NJOY job by declaring a long chain of modules with their specific input instructions. Or, an NJOY job may be as simple as a single **stop** instruction, which just terminates the job.
3 Methodology

3.1 Introduction

The NJOY input instructions had to be understood in order to design the new input format. Each module in the NJOY software system, as described in reference [1, 4], was analyzed separately such that a general structure and common language features could be extracted and used for further analysis.

The NJOY modules are listed in table 1. The modules have been prioritized with a number. The number indicates the importance level of the module to the MACRO project [3]. A low number indicates high priority, e.g. a module with a low number had to be implemented before a module with a higher number.

<table>
<thead>
<tr>
<th>NJOY Module</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>acer</td>
<td>1</td>
</tr>
<tr>
<td>broadr</td>
<td>1</td>
</tr>
<tr>
<td>ccccr</td>
<td>3</td>
</tr>
<tr>
<td>covr</td>
<td>1</td>
</tr>
<tr>
<td>dtfr</td>
<td>3</td>
</tr>
<tr>
<td>errorr</td>
<td>1</td>
</tr>
<tr>
<td>gaminr</td>
<td>3</td>
</tr>
<tr>
<td>gaspr</td>
<td>3</td>
</tr>
<tr>
<td>groupr</td>
<td>1</td>
</tr>
<tr>
<td>heatr</td>
<td>1</td>
</tr>
<tr>
<td>leapr</td>
<td>3</td>
</tr>
<tr>
<td>matxsr</td>
<td>3</td>
</tr>
<tr>
<td>mixr</td>
<td>3</td>
</tr>
<tr>
<td>moder</td>
<td>1</td>
</tr>
<tr>
<td>plotr</td>
<td>2</td>
</tr>
<tr>
<td>powr</td>
<td>3</td>
</tr>
<tr>
<td>purr</td>
<td>3</td>
</tr>
<tr>
<td>reconr</td>
<td>1</td>
</tr>
<tr>
<td>resxsr</td>
<td>3</td>
</tr>
<tr>
<td>thermr</td>
<td>1</td>
</tr>
<tr>
<td>unresr</td>
<td>3</td>
</tr>
<tr>
<td>viewr</td>
<td>2</td>
</tr>
<tr>
<td>wimsr</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 1: Implementation priority of the NJOY modules

As stated in reference [5], a translator (compiler) is a program that can read a program in one language and translate it into an equivalent program in another language. In the following subsections, principles and techniques for constructing a translator presented in reference [5], is described.
3.2 Designing the New Input Format

The syntax definition of the new input format was specified in a notation called context-free grammar [6]. A context-free grammar is a convenient method of specifying the syntax of a programming language. For instance, the assignment (declaration) of an identifier can have the form

\[
\text{material} = 9237
\]

which can be expressed in a context-free grammar as the production

\[
\text{assignment ::= l_value} \text{ "=} \text{ r_value}
\]

where l_value and r_value are other productions expressing the structure of the left and right hand side of the assignment, respectively.

3.3 Building the Translator

In reference [5], the translation process is described as a sequence of phases. Each phase inspects and transforms a representation of the source program to another. Phases such as lexical analysis, syntax analysis, and semantic analysis has been used throughout this work and is described in section 3.3.1, 3.3.2, and 3.3.3 respectively.

The translator, which is supposed to translate the input format into NJOY input instructions, was partly constructed using a lexical-analyzer generator [7] and a parser generator [8]. The translator was written in the Python programming language [9], in a Unix-like environment.

3.3.1 Lexical Analysis

Lexical analysis is the process of dividing the source program into sequences of characters, called tokens [10]. Each token describes a group of characters in the source program as an abstract type.

For example, the identifier material, the assignment character, =, and the integer 9237 could be represented as tokens of the form

\[
\langle\text{IDENTIFIER, material}\rangle,
\langle\text{ASSIGNMENT, =}\rangle, \text{ and }
\langle\text{INTEGER, 9237}\rangle
\]

PLY Lex [11] was used to generate a lexical analyzer (lexer) for the input format. The method of identifying the tokens was implemented by using the notation of regular expressions [12] in PLY Lex.
3.3.2 Syntax Analysis

Syntax analysis is the process of creating a tree-like representation, an abstract syntax tree, composed of the tokens generated by the lexical analyzer [13]. The syntax tree is used to describe the grammatical structure of the source program.

PLY Yacc [11] was used to generate a syntax analyzer (parser) for the grammar definition of the input format. The method of building the syntax tree was implemented by using the facilities provided by the PLY tools.

3.3.3 Semantic Analysis

Semantic analysis is the process of checking the syntax tree for errors that have to do with the meaning of the program [14].

For example, according to reference [4], card 1, 2 and 3 in the acer module must always be defined, and they must be defined in sequential order. The translator should report an error if these rules are violated; such as when card 1 has not been defined or when card 3 has been defined prior to card 2.

Type checking is another important part of the semantic analysis where the translator checks that each operator has valid operands.

For example, the identifier hk, in card 3 module acer, is used to denote a descriptive character string. According to reference [4], hk must be declared as a character string and must not exceed 70 characters in length. The translator should report an error if these rules are violated; such as when hk has been declared as an integer, or when the character string contains more than 70 characters.

3.4 Testing

Testing was carried out continuously during the design and implementation of the input format and the translator. The NJOY test problems\(^1\) [1] was used to test the functionality of both the input format and the translator.

The NJOY test problems was manually translated into equivalent NJOY jobs in the new input format, which were run through the translator. The resulting output was compared with the expected output, to verify that the translator was working appropriately.

The Python unit testing framework [9] was utilized to set up the testing environment.

\(^1\)The NJOY Test Problems are test runs which are used to test the functionality of the NJOY software system. See http://t2.lanl.gov/codes/njoy99/
4 Implementation

4.1 NJOY Input Format (NIF)

The new input format, NJOY Input Format (NIF), is basically the original NJOY input instructions which have been annotated with a syntax to make it easier to read and express. NIF has been designed to appear more like a high-level programming language.

4.1.1 Grammar Definition

The proposed NJOY Input Format (NIF) is illustrated as a context-free grammar definition in algorithm 3 on the next page. The structure of the grammar is simple. Just like in reference [4], a NIF program is an ordered sequence of modules. Each module is composed by an ordered sequence of cards. A card is an ordered sequence of value definitions.

In NIF, the start symbol is program. The capitalized terminals, such as MODULE and CARD, are token classes specified by the lexer. Special symbols are denoted within double quotes. empty denotes the empty string.

An assignment denotes that a left hand side is assigned to hold the values of a right hand side. A left hand side is an ordered list of elements, where the elements can be an array or identifier. A right hand side is an ordered list of elements, where the elements can be a float, integer, null or a string. As such, a value definition is an array or identifier that has been declared to hold the value of either a floating-point number, natural number, empty string or a character string.

As indicated by the grammar, NIF supports multiple assignment. That is, multiple identifiers can be assigned in the same expression. For example, the expression

\[
\text{material, temp} = 9237, 300.0;
\]

denotes that the identifier material holds the integer 9237, and the identifier temp holds the float 300.0. The syntax analysis in the parser enforces that the number of elements on both sides of an assignment are the same.
Algorithm 3 NJOY Input Format (NIF) Grammar Definition

program ::= module_list

module_list ::= module module_list  
               | empty
module ::= MODULE "{" card_list "}"

card_list ::= card card_list  
              | empty
card ::= CARD "{" stmt_list "}"

stmt_list ::= statement stmt_list  
             | empty
statement ::= expression ";"

expression ::= assignment

assignment ::= l_value_list "=" r_value_list

l_value_list ::= l_value  
               | l_value "," l_value_list
r_value_list ::= r_value  
                | r_value "," r_value_list

l_value ::= array  
          | ident
array ::= IDENTIFIER "][ INTEGER "]"
ident ::= IDENTIFIER

r_value ::= FLOAT  
          | INTEGER  
          | NULL  
          | STRING
4.2 NJOY Input Format Translator (nifty)

4.2.1 Structure of the Translator

The translator, NJOY Input Format Translator (nifty), was constructed as a set of modules where each module implements a specific phase in the translation process. Five phases have been implemented as part of the translation process and are shown in figure 1.

The first phase is the lexical analysis which is implemented by the lexer module. The second phase, syntax analysis, is implemented by the parser module.

The third phase, implemented by the organizer module, is a special phase where the order of the statements in a card are analyzed and possible rearranged.

The fourth phase is the semantic analysis which is implemented by the module named analyzer. The fifth, and final, phase of the translator is the emitter module which implements a NJOY input instructions generator.

A basic user manual for the translator is available in appendix A on page 26.

4.2.2 Reserved Keywords

An important design choice is that the translator will enforce the use of reserved keywords to specify NIF programs. It will not only consider card and module names as reserved keywords, but also identifier names. As such, it is not possible to use an identifier name until it has been defined as an identifier in the translator. Similarly, it is not possible to use a card or module name which has not been defined in the translator. This restricts the expressiveness of the input format, but allows detailed analysis of the semantics in the organizer and analyzer modules. As a consequence, it also forces the user to write consistent and readable input files – which has been the objective of this work.
4.2.3 The Modules

**Lexer**  The lexer is responsible for recognizing character patterns and generating the appropriate NIF tokens. As input, the lexer expects a NIF program and will generate a token stream as its output unless the lexer detects a lexical error. If a lexical error is detected, an error message will be reported and the translation process will stop at this phase. The lexer will only recognize card and module names which are specified in reference [4], thus enforcing the use of a specific set of cards and modules as mentioned previously. The lexer also recognizes comments in the input program. The comments will be discarded during the lexical analysis and thus won’t be passed on to the next phase in the translation process.

**Parser**  The parser is responsible for enforcing the structure of the NIF grammar and constructing the syntax tree. As input, the parser expects a stream of tokens generated by the lexer. The parser will produce a syntax tree as its output, which represents the structure of the NIF program. If the parser detects a syntax error, an error message will be reported and the translation process will stop at this phase.

**Organizer**  The organizer analyzes the syntax tree produced by the parser. Its purpose is to rearrange the statements in a card such that they appear in the expected, working order. As such, it should be possible to write a NIF program without having to list the statements in a card in the expected order as indicated by reference [4].

The NJOY modules and the cards within the modules still needs to be given in the correct order though. This is due to the fact that the number of possible NJOY jobs is infinite (all may not be functional in the NJOY software system, though). An infinite number of NJOY jobs can simply be created by just appending another module specification to an existing NJOY job in order to create a new one. Simply stated, the translator can not guess the intention of the job due to the number of possible combinations the modules may be listed in. Hence, the modules must be provided in the expected order by the user. Cards are not arrangeable either, since they also are prone to be repetitive. It is not possible to determine which card should go first from a set of cards (with the same name) which e.g. only contains a descriptive title. The cards must also be provided in the expected order by the user.

Each NJOY module requires its own organizer implementation since each module has its specific set of rules as described in reference [4]. Since the identifier names are hardwired in the translator, the organizer is able to do a detailed analysis of the syntax tree and easily detect if a specific identifier has been defined out of order.

If any statements have been provided out of order in a card, and the organizer is able to arrange the statements, a new syntax tree is returned where the statements have been ordered in the expected sequence. If the organizer somehow fails to organize the syntax tree, it will return the original syntax tree
as produced by the parser and pass it on to the next phase in the translation process.

**Analyzer** The analyzer expects a syntax tree as its input. Like in the organizer phase, the NJOY modules needs to be analyzed separately since each module has its specific set of rules. As such, each module also requires its own analyzer implementation.

The analyzer basically visits every node in the order they appear in the syntax tree and checks if it is the expected one. The analysis can be made very detailed since the translator can, to some extent, predict the next card or identifier due to the ordered nature described in reference [4]. Since the cards and the identifiers have reserved names, the analyzer is able to easily determine whether a card or an identifier is the expected one. Using reserved names also makes type checking easy, since a reserved identifier in a specific card may be associated with a specific type, range, size, length, et cetera.

The analyzer does not alter the syntax tree, it just analyzes it. The input syntax tree will be the output of the analyzer if the syntax tree is semantically correct according to the translator. If the analyzer detects a semantic error in the syntax tree, an error message will be reported and the translation process will stop at this phase.

**Emitter** The emitter expects a syntax tree as its input and it is responsible for generating NJOY input instructions from the syntax tree. The emitter simply flattens the tree structure and formats the instructions to their corresponding counterparts in the NJOY input instructions format. The emitter returns a string with the resulting NJOY input instructions. Each card in the resulting output has been annotated with a descriptive comment, indicating which card it is, to make it easier to find errors.

### 4.3 Translation Verification

As previously described in section 3.4 on page 7, the NJOY test problems [1] were manually translated into NIF programs expressing the equivalent NJOY jobs. The resulting NIF programs were used for verifying that the implementation was working appropriately by setting up a test suite using the Python unit testing framework [9]. Each test problem was set up to be run through each individual phase in the translation process, and each run were expected to be successful since the NIF programs should be an equivalent and functional version of the original test problems.

The resulting output, as produced by the emitter, was compared with the expected output. That is, each NIF version of the test problems were compared with its corresponding original NJOY test problem.

Note that modified versions of the original NJOY test problems had to be used as the expected output when comparing the resulting output from the translator. The comments that the emitter appends to every card had to be appended to the expected output such that the comparison could be performed.
4.4 Translation Efficiency

A simple Python script was written to check the resulting translation efficiency of the translator. Two different notions of timing were used in the script, namely *process time* and *wall time*.

Process time is the time that the entire task spent executing on the processor, measured by `time.clock()` which should be used for timing algorithms [9]. Wall time is the time that elapsed from when the task was started to when the task finished, measured by checking the difference in time using `time.time()` [9]. The main difference between process time and wall time is that wall time is the time it takes until the system delivers the computed result, whereas process time is the time that it took to compute the result.
5 Results

5.1 NJOY Input Format (NIF)

The result of the proposed grammar described in section 4.1 on page 8 is best illustrated with examples. Algorithm 4 illustrates NJOY input instructions (slightly modified to make it shorter for illustrational purposes) from NJOY Test Problem 2 [1]. In algorithm 5 on the following page, lines 1 through 9 from algorithm 4 are expressed in NIF.

Algorithm 4 Modified subset of NJOY Test Problem 2

```
1  moder
2   20 -21/
3  reconr
4   -21 -22/
5 'pendf tape for pu-238 from endf/b-iv tape 404'/
6   1050 1/
7   0.005/
8 '94-pu-238 from endf/b tape t404'/
9   0/
10  broadr
11  -21 -22 -23/
12 (1050 3 0 1/
13   0.005/
14  300.0 900.0 2100.0/
15   0/
16  stop
```
Algorithm 5 NIF version of Algorithm 4 on the preceding page, lines 1 through 9

moder {
    card_1 {
        pendf_input = 20;
        pendf_output = -21;
    }
}

reconr {
    card_1 {
        nendf = -21;
        npend = -22;
    }
    card_2 {
        tlabel = "pendf tape for pu-238 from endf/b-iv tape 404";
    }
    card_3 {
        mat = 1050;
        ncards = 1;
    }
    card_4 {
        err = 0.005;
    }
    card_5 {
        cards = "94-pu-238 from endf/b tape t404";
    }
    /* Card 6 not defined since 'ngrid' defaults to 0 in first card 3. */
    card_3 { mat = 0; } // Terminate reconr.
}

Descriptive names for the identifiers on line 3 and 4 have been specified in the translator. The other identifier names have been chosen to reflect the documentation in reference [4], but the identifier names are interchangeable in the translator (the identifier names are hardwired in the translator). Line 30 and 31 shows how comments are expressed in NIF. Line 30 illustrates the structure of multiline comments while line 31 illustrates the structure of single line comments.
Algorithm 6 is a NIF version of the lines 10 through 16 from algorithm 4 on page 14. It shows how arrays are expressed in NIF (lines 24 through 26). The stop instruction on line 16 in algorithm 4 on page 14 does not have to be specified in NIF, the translator will automatically append it in the translation process.

When combined, algorithm 5 on the preceding page and algorithm 6 forms the complete NJOY job as listed in algorithm 4 on page 14.

Algorithm 6 NIF version of Algorithm 4 on page 14, lines 10 through 16

```c
broadr {
    card_1
    {
        nendf = -21;
        nin = -22;
        nout = -23;
    }
    card_2
    {
        mat1 = 1050;
        ntemp2 = 3;
        istart = 0;
        istrap = 1;
    }
    card_3
    {
        errthn = 0.005;
    }
    card_4
    {
        temp2[0] = 300.0;
        temp2[1] = 900.0;
        temp2[2] = 2100.0;
    }
    /* Terminate execution of broadr with mat1 = 0 as usual. */
    card_5
    {
        mat1 = 0;
    }
}
```
5.2 NJOY Input Format Translator (nifty)

Table 2 shows the implementation status for the NJOY modules. Each column entry indicates the completeness of a translator phase for a given NJOY module.

<table>
<thead>
<tr>
<th>NJOY Module</th>
<th>Lexer</th>
<th>Parser</th>
<th>Organizer</th>
<th>Analyzer</th>
<th>Emitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>acer</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>broadr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ccccr</td>
<td></td>
<td>0%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>covr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dtfr</td>
<td></td>
<td>0%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>errorr</td>
<td></td>
<td>70%</td>
<td>20%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gamnr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gaspr</td>
<td></td>
<td>100%</td>
<td>99%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>groupr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heatr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>leapr</td>
<td>100%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>matxsr</td>
<td></td>
<td>0%</td>
<td></td>
<td></td>
<td>100%</td>
</tr>
<tr>
<td>mixr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>moder</td>
<td></td>
<td>100%</td>
<td>95%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>plotr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>powr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>purr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>reconr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rexsxsr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thermr</td>
<td></td>
<td>100%</td>
<td>90%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>unresr</td>
<td></td>
<td>100%</td>
<td>95%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>viewr</td>
<td></td>
<td>100%</td>
<td>10%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wimsr</td>
<td></td>
<td>0%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Implementation status for the NJOY modules

The completeness of the implementation has been rated in a grading scale with percentage. The grades have been set with respect to whether the functionality of the phases presented in section 4.2.3 on page 11 (also see section 3.3 on page 6) has been fulfilled or not. 100% indicates that the functionality has been finished. 0% indicates that the implementation of the functionality has not been started. The other percentages are rough approximations of how much functionality that has been implemented.
5.3 Translation Verification

The result of the translation verification is summarized in Table 3. All test problems listed in Appendix B on page 29 passed all the phases in the translation process. That is, the test problems were successfully translated\(^2\); no lexical, syntax, nor semantic errors were found. No differences between the expected output and the resulting output were detected for the test problems.

<table>
<thead>
<tr>
<th>Test Problem</th>
<th>Translator Phases</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>tp01</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp02</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp03</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp04</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp05</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp06</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp07</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp08</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp10</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp11</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp12</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp13</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp14</td>
<td>Passed</td>
<td>Expected</td>
</tr>
<tr>
<td>tp17</td>
<td>Passed</td>
<td>Expected</td>
</tr>
</tbody>
</table>

Table 3: Translation verification results for the test problems

\(^2\)Note that the organizer's ability to arrange statements in the correct order has not been tested for the test problems, since the instructions in the test problems have been provided in the expected order.
5.4 Translation Efficiency

The efficiency of the translator was tested by running the entire translation process for each test problem, listed in Appendix B on page 29, 10,000 times. Table 4 shows the resulting runtimes, both process time and wall time, in seconds. The resulting runtimes denotes the aggregate of 10,000 repeated runs for a given test problem.

<table>
<thead>
<tr>
<th>Test Problem</th>
<th>Process Time</th>
<th>Wall Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>tp01</td>
<td>2.87</td>
<td>345.22</td>
</tr>
<tr>
<td>tp02</td>
<td>2.87</td>
<td>374.29</td>
</tr>
<tr>
<td>tp03</td>
<td>2.81</td>
<td>292.49</td>
</tr>
<tr>
<td>tp04</td>
<td>2.79</td>
<td>281.89</td>
</tr>
<tr>
<td>tp05</td>
<td>3.10</td>
<td>255.14</td>
</tr>
<tr>
<td>tp06</td>
<td>3.05</td>
<td>346.87</td>
</tr>
<tr>
<td>tp07</td>
<td>3.07</td>
<td>278.47</td>
</tr>
<tr>
<td>tp08</td>
<td>2.75</td>
<td>294.17</td>
</tr>
<tr>
<td>tp10</td>
<td>3.05</td>
<td>288.98</td>
</tr>
<tr>
<td>tp11</td>
<td>2.84</td>
<td>373.76</td>
</tr>
<tr>
<td>tp12</td>
<td>2.72</td>
<td>301.82</td>
</tr>
<tr>
<td>tp13</td>
<td>3.19</td>
<td>280.76</td>
</tr>
<tr>
<td>tp14</td>
<td>3.25</td>
<td>251.40</td>
</tr>
<tr>
<td>tp17</td>
<td>2.95</td>
<td>350.35</td>
</tr>
<tr>
<td><strong>Average Time:</strong></td>
<td><strong>2.95</strong></td>
<td><strong>308.26</strong></td>
</tr>
</tbody>
</table>

Table 4: Aggregated runtimes (in seconds) for 10,000 runs

The average wall time for a single run for the test problems is $\frac{308.26}{10000} \approx 0.031$ seconds.

The repeated runs were conducted on a multi-user system equipped with three Dual Core AMD Opteron Processor 280 at 2.4GHz each, and a total of 3.6GB RAM. The system was running Linux 2.6.18 and Python 2.4.3.

The Python library functions `time.clock()` and `time.time()` [9] were used to measure the process time and wall time, respectively.
6 Discussion

6.1 NJOY Input Format (NIF)

As indicated by the examples listed in section 5.1 on page 14, a typical NIF program is vertically long compared to the compact notation of the NJOY input instructions. NIF programs can of course be specified in a compact form as well, e.g. on a single line, but this is not the intended usage of NIF. The purpose of NIF is to make NJOY jobs readable. The readability would be limited if the jobs were expressed on a single line.

The proposed grammar does not differ much from the original NJOY input instructions since it basically is an annotated version of them. The NIF grammar could have been expanded to include more complex programming idioms, such as an `if` expression to allow flow control in a NIF program. Although, the structure of NIF was designed to be simple and to closely resemble the original input instructions such that a user does not need to learn a completely new programming language to specify NJOY jobs. Another intention of this design choice is that the NJOY input instructions documented in reference [4] can be used to specify NJOY jobs in NIF.

6.2 NJOY Input Format Translator (nifty)

The translator is able to translate NIF programs into functional NJOY input instructions, but an organizer and analyzer has not been provided for all modules in the NJOY software system due to time constraints of this thesis. As such, the important semantic analysis of the translator is incomplete. However, much of the needed functionality and structure is provided by the existing implementation such that both the organizer and the analyzer should be easy to complete.

The implementation of the analyzer module has been the most time consuming task when designing the translator. It requires detailed analysis of what kind of input the NJOY modules expect and how they operate on it. The documentation in reference [4] was the main resource used while implementing the semantic analysis in the analyzer. It was evident that this was not a sufficient resource for the task at hand. It does not clearly indicate the expected type for all identifiers, nor the expected integer ranges or length of the character strings. In some cases, it has also been hard to deduce which cards that must be supplied by just reading the documentation in reference [4]. To fully check the semantics of a NIF program, the source code for the NJOY software system must be studied in greater detail. The ENDF formats must also be studied in greater detail in order to understand the semantics and what kind of values that the NJOY modules accept.
6.3 Translation Verification

The testing that was conducted within this work is not rigorous enough due to time constraints of this thesis. NJOY is a large and complex program\(^3\) with many possible combinations of input within each NJOY module and card. The NJOY test problems \([1]\) which were used to test the translation functionality is a very small set of possible NJOY jobs. Hence, there is a lot of scenarios within each NJOY module that has not been tested.

6.4 Translation Efficiency

The efficiency testing of the translator as described in section 5.4 on page 19 was conducted in a simple fashion. The resulting process runtimes revealed that the performance of the translator implementation is not a huge bottleneck, compared to the much greater wall times. The performance appeared to be good enough to fit the purpose of the translator. Therefore, more elaborate testing of the efficiency was not conducted.

\(^3\)The source files for the NJOY software system consists of more than 100 000 lines.
7 Conclusions

In this thesis, a new input format, NJOY Input Format (NIF), has been designed. A translator which is able to translate NIF into NJOY input instructions has been implemented.

It is possible to specify basic NJOY jobs in NIF. The resulting NIF programs can be translated into NJOY input instructions, which can be run by the NJOY software system. Production use is although not advisable, since it has been challenging to conduct rigorous and complete testing.

It has also been evident that analyzing the NJOY input instructions is not enough to design a new input format for the NJOY software system. Analyzing the ENDF libraries and the source code for the NJOY software system is required in order to build a translator which can conduct a complete semantic analysis for an NJOY job.
8 Future Work

Future work includes completing the semantic analysis and the organizer feature for all modules in the NJOY software system. The NJOY Input Format and the translator also needs to be systematically evaluated and verified by a complete software quality assurance process as described in reference [15]. The project may also be expanded to include default scenarios, which uses normal mode of operation per default, such that the user does not have to specify exhaustive NJOY jobs just to convert a library into another.

A spin-off project, that is related to developing a user friendly and readable input format, is to construct a graphical user interface editor which can display and produce NJOY input instructions in a user friendly fashion.
References


A User Manual

A.1 Structure of nifty

The nifty directory structure is organized as shown in figure A.1.

nifty/
  bin/
    analyzer
    bench
    emitter
    lexer
    nifty
    organizer
    parser
    test
  data/
   ...
   test_problems/
  nifty/
    analyzer/
    emitter/
    environment/
    lexer/
    organizer/
    parser/
    settings/
    tests/
  [ply/]

Figure A.1: Directory Structure of nifty

The nifty/bin/ directory includes all executable Python scripts which are used for running and testing the translator. The nifty executable in the nifty/bin/ directory runs the complete translation process on an input NIF program. The test executable runs the test suite. The bench executable is a script used for testing the efficiency of the translator. The other executable scripts run their corresponding named phase in the translation process (and all the successive phases that they depend on).

The test problems are located in the nifty/data/test_problems/ directory. The nifty/nifty/ directory contains the source code for the translator. The optional directory ply/ indicates where PLY can be placed such that the translator is able to locate it.


A.2 Installation

Python version 2.2 or greater is required to use nifty. Python version 2.4.3 and 2.6.1 has been tested with nifty and are known to work. nifty itself does not require any special installation methods, although PLY [11] is required to run the translator. It is sufficient to download PLY and put the ply/ directory from PLY in the nifty/ top directory as indicated by figure A.1 on the previous page. (Note the non-restrictive license of PLY generously provided by its author.)

A.3 Running the Translator

The translator has been implemented as a command-line based interface for a Unix-like environment. To run the entire translation process, the nifty executable in the nifty/bin/ directory should be used. Issuing the command

```
bin/nifty -h
```

in the nifty/ top directory, will print the usage message shown in figure A.2.

```
usage: nifty [options] [input_file] [output_file]
options:
  -h, --help       show this help message and exit
  -a               don’t analyze the input
  -o               don’t organize the input
```

Figure A.2: bin/nifty usage

The options flag(s) are optional. The input_file and output_file are also optional. If no input file is given, standard input (stdin) will be used as the input source. If no output file is given, the result will be redirected to standard output (stdout).

As an example, the command

```
bin/nifty input.nif output
```

will simply run the translator on a file named input.nif and output the resulting NJOY input instructions on a file named output. The analyzer and organizer phase can be skipped by giving the -a and -o flag

```
bin/nifty -a input.nif output, to skip the analyzer phase

bin/nifty -o input.nif output, to skip the organizer phase
```

To skip both the organizer and analyzer phase, run nifty with both flags specified

```
bin/nifty -ao input.nif output
```

27
A.4 Settings

The `nifty/nifty/settings/` directory as shown in figure Figure A.1 on page 26 contains module specific settings. Each module has its own settings file. The analyzer and organizer phase utilizes the settings files when processing a syntax tree. The settings describes the expected identifiers and their expected order within the cards in a module.
B Test Problems

In this section, the test problems that were used for testing the functionality of the translator is listed. Both the NIF versions and the expected NJOY input instructions are provided. The test problems listed in this section are also available in the `nifty/data/test_problems/` directory, as described in section A.1 on page 26.

B.1 Test Problem 01 (tp01)

NIF Version of Test Problem 01

```c
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
/* Card 6 skipped since ngrid defaults to 0 in first card 3 */
```
mat = 0;
}
}
 srand
{
   card_1
   {
      nendf = -21;
      nin = -22;
      nout = -23;
   }
   card_2
   {
      mat1 = 1306;
      ntemp2 = 1;
   }
   card_3
   {
      errthn = 0.005; // Use C-style floats.
   }
   card_4
   {
      temp2[0] = 300.0; // Use C-style floats.
   }
   card_5
   {
      mat1 = 0;
   }
}
 heatr
{
   card_1
   {
      nendf = -21;
      nin = -23;
      nout = -22;
   }
   card_2
   {
      matd = 1306;
      npk = 1;
   }
   card_3
   {
      mtk[0] = 444; // Note that mtk has to be defined as an array.
   }
   /* Card 4, 5, and 5a are skipped since nqa defaults to 0 in card 2. */
}
 thermr
{
   card_1
   {
      nendf = 0;
      nin = -22;
      nout = -24;
   }
   card_2
{  
  matde = 0;
  matdp = 1306;
  nbin = 8;
  ntemp = 1;
  iinc = 1;
  icoh = 0;
  natom = 1;
  mtref = 221;
  iprint = 0;
}

card_3
{
  tempr[0] = 300.0; // Use C-style floats.
}

card_4
{
  tol = 0.05; // Use C-style floats.
  emax = 1.2;
}

thermr
{
  card_1
  {
    nendf = 26;
    nin = -24;
    nout = -23;
  }
  card_2
  {
    matde = 1065;
    matdp = 1306;
    nbin = 8;
    ntemp = 1;
    iinc = 4;
    icoh = 1;
    natom = 1;
    mtref = 229;
    iprint = 0;
  }
  card_3
  {
    tempr[0] = 300.0; // Use C-style floats.
  }
  card_4
  {
    tol = 0.05; // Use C-style floats.
    emax = 1.2;
  }
}

groupr
{
  card_1
  {
    nendf = -21;
    npend = -23;
    ngout1 = 0;
    ngout2 = -24;
  }
}
card_2
{
  nmatb = 1306;
  ign = 3;
  igg = 3;
  iw = 3;
  lorder = 3;
  ntemp = 1;
  nsigz = 1;
  iprint = 1;
}

card_3
{
  title = "carbon in graphite";
}

card_4
{
  temp[0] = 300;
}

card_5
{
  sigz[0] = 1.0e10; // No trailing dots. Use C-style floats.
}

card_9
{
  nfd = 3;
  mt = 1;
  mname = "total";
}

card_9
{
  nfd = 3;
  mt = 2;
  mname = "elastic";
}

card_9
{
  nfd = 3;
  mt = 4;
  mname = "inelastic";
}

card_9
{
  nfd = 3;
  mt = 51;
  mname = "discrete inelastic";
}

card_9
{
  nfd = 3;
  mt = -68;
  mname = "continued";
}

card_9
{
  nfd = 3;
  mt = 91;
  mname = "continuum inelastic";
}
{
    mfd = 3;
    mtd = 102;
    mtname = "n,g";
}
{
    mfd = 3;
    mtd = 103;
    mtname = "(n,p)";
}
{
    mfd = 3;
    mtd = 104;
    mtname = "(n,d)";
}
{
    mfd = 3;
    mtd = 107;
    mtname = "(n,a)";
}
{
    mfd = 3;
    mtd = 221;
    mtname = "free thermal scattering";
}
{
    mfd = 3;
    mtd = 229;
    mtname = "graphite inelastic thermal scattering";
}
{
    mfd = 3;
    mtd = 230;
    mtname = "graphite elastic thermal scattering";
}
{
    mfd = 3;
    mtd = 251;
    mtname = "mubar";
}
{
    mfd = 3;
    mtd = 252;
    mtname = "x1";
}
{
    mfd = 3;
    mtd = 253;
mtname = "gamma";
}
card_9
{
    mfd = 3;
    mtd = 301;
    mtname = "total heat production";
}
card_9
{
    mfd = 3;
    mtd = 444;
    mtname = "total damage energy production";
}
card_9
{
    mfd = 6;
    mtd = 2;
    mtname = "elastic";
}
card_9
{
    mfd = 6;
    mtd = 51;
    mtname = "discrete inelastic";
}
card_9
{
    mfd = 6;
    mtd = -68;
    mtname = "continued";
}
card_9
{
    mfd = 6;
    mtd = 91;
    mtname = "continuum inelastic";
}
card_9
{
    mfd = 6;
    mtd = 221;
    mtname = "free thermal scattering";
}
card_9
{
    mfd = 6;
    mtd = 229;
    mtname = "graphite inelastic thermal scattering";
}
card_9
{
    mfd = 6;
    mtd = 230;
    mtname = "graphite elastic thermal scattering";
}
Expected NJOY Input Instructions for Test Problem 01

1  moder
2   20 -21/ ### card_1
3  reconr
4    -21 -22/ ### card_1
5    'pendf tape for c-nat from endf/b tape 511'/ ### card_2
6   1306 3/ ### card_3
7    0.005/ ### card_4
8    'c-nat from tape 511'/ ### card_5
9    'processed by the njoy nuclear data processing system'/ ### card_5
10    'see original endf/b-v tape for details of evaluation'/ ### card_5
11   0/ ### card_3
12  broads
13     -21 -22 -23/ ### card_1
14    1306 1/ ### card_2
15    0.005/ ### card_3
16    300.0/ ### card_4
17    0/ ### card_5
18  heatr
19     -21 -23 -22/ ### card_1
20    1306 1/ ### card_2
21    444/ ### card_3
22  therer
23     0 -22 -24/ ### card_1
24    0.1306 0 1 1 0 1 221 0/ ### card_2
25    300.0/ ### card_3
26    0.05 1.2/ ### card_4
27  therer
28     26 -24 -23/ ### card_1
29    1006 1306 8 1 4 1 1 229 0/ ### card_2
30    300.0/ ### card_3
31    0.05 1.2/ ### card_4
32 groupr
33     -21 -23 0 -24/ ### card_1
B.2 Test Problem 02 (tp02)

NIF Version of Test Problem 02

```c

t moder

{ 
   card_1
   { 
      nin = 20;
      nout = -21;
   }
}

reconr

{ 
   card_1
   { 
      nendf = -21;
      npend = -22;
   }
   card_2
   { 
      tlabel = "pendf tape for pu-238 from endf/b-iv tape 404";
   }
   card_3
   { 
      mat = 1050;
      ncards = 3;
   }
   card_4
   { 
      err = 0.005; // Use C-style floats instead of ".005".
   }
   card_5
   { 
      cards = "94-pu-238 from endf/b tape t404";
   }
   card_5
   { 
      cards = "processed by the njoy nuclear data processing system";
   }
   card_5
   { 
      cards = "see original endf/b-iv tape for details of evaluation";
   }
   /* Card 6 skipped since ngrid defaults to 0 in first card 3. */
   card_3
   { 
      mat = 0;
   }
}

broadr

{ 
   card_1
   { 
      nendf = -21;
      nin = -22;
      nout = -23;
   }
```

37
card_2
{
  mat1 = 1050;
  ntemp2 = 3;
  istart = 0;
  istrap = 1;
  temp1 = 0;
}
card_3
{
  errthn = 0.005; // Use C-style floats instead of ".005".
}
card_4
{
  /* In this example, Each temperature is declared as an element in an
     array.
     ntemp2 in card_2 denotes the number of expected temperatures.
   */
  temp2[0] = 300.0;
  temp2[1] = 900.0;
  temp2[2] = 2100.0;
}
card_5
{
  mat1 = 0;
}
moder
{
card_1
{
  nin = -23;
  nout = 33;
}
}
unresr
{
  card_1
  {
    nendf = -21;
    nin = -23;
    nout = -24;
  }
  card_2
  {
    natd = 1050;
    ntemp = 3;
    nsigz = 7;
    iprint = 1;
  }
  card_3
  {
    temp[0] = 300;
    temp[1] = 900;
    temp[2] = 2100;
  }
  card_4
  {
    sigz[0] = 1.0e10;
sigz[1] = 1.0e5;
sigz[2] = 1.0e4;
sigz[3] = 1000.0;
sigz[4] = 100.0;
sigz[5] = 10.0;
sigz[6] = 1;
}

card_2
{
  matd = 0;
}
}
groupr
{
card_1
{
  nendf = -21;
  npend = -24;
  ngout1 = 0;
  ngout2 = -25;
}

card_2
{
  matb = 1050;
  ign = 5;
  igg = 0;
  iwt = 4;
  lord = 3;
  ntemp = 3;
  nsigz = 7;
  iprint = 1;
}

card_3
{
  title = "94-pu-238";
}

card_4
{
  /* ntemp in card_2 denotes the number of expected temperatures. */
  temp[0] = 300.0;
  temp[1] = 900.0;
  temp[2] = 2100.0;
}

card_5
{
  /* nsigz in card_2 denotes the number of expected sigma zeroes. */
  sigz[0] = 1.0e10;
  sigz[1] = 1.0e5;
  sigz[2] = 1.0e4;
  sigz[3] = 1000.0;
  sigz[4] = 100.0;
  sigz[5] = 10.0;
  sigz[6] = 1;
}

card_8c
{
  eb = 0.1;
  tb = 0.025;
  ec = 0.8208e06;
  tc = 1.4e06;
}
/* Reactions for temperature 300.0. */
card_9
{
  mfd = 3;
  std = 1;
  mtname = "total";
}
card_9
{
  mfd = 3;
  std = 2;
  mtname = "elastic";
}
card_9
{
  mfd = 3;
  std = 16;
  mtname = "n2n";
}
card_9
{
  mfd = 3;
  std = 17;
  mtname = "n3n";
}
card_9
{
  mfd = 3;
  std = 18;
  mtname = "fission";
}
card_9
{
  mfd = 3;
  std = 102;
  mtname = "capture";
}
card_9
{
  mfd = 3;
  std = 251;
  mtname = "mubar";
}
card_9
{
  mfd = 3;
  std = 252;
  mtname = "xi";
}
card_9
{
  mfd = 3;
  std = 253;
  mtname = "gamma";
}
card_9
{
  mfd = 3;
mtd = 259;
mtname = "1/v";
}

card_9
{
  mfd = 6;
  mtd = 2;
  mtname = "elastic";
}

card_9
{
  mfd = 6;
  mtd = 16;
  mtname = "n2n";
}

card_9
{
  mfd = 6;
  mtd = 17;
  mtname = "n,3n";
}

card_9
{
  mfd = 6;
  mtd = 18;
  mtname = "fission";
}

card_9
{
  mfd = 6;
  mtd = 51;
  mtname = "discrete inelastic";
}

card_9
{
  mfd = 6;
  mtd = -59;
  mtname = "continued";
}

card_9
{
  mfd = 6;
  mtd = 51;
  mtname = "continuum inelastic";
}

/* Terminate temperature 300.0. */
card_9
{
  mfd = 0;
}

/* Reactions for temperature 900.0. */
card_9
{
  mfd = 3;
  mtd = 1;
  mtname = "total";
}

card_9


```c
337  
338  {  
339     mfd = 3;  
340     mtd = 2;  
341     mtname = "elastic";  
342  }
343  
344  card_9  
345  {  
346     mfd = 3;  
347     mtd = 18;  
348     mtname = "fission";  
349  }
350  
351  card_9  
352  {  
353     mfd = 3;  
354     mtd = 102;  
355     mtname = "capture";  
356  }
357  
358  /* Terminate temperature 900.0. */  
359  card_9  
360  {  
361     mfd = 0;  
362  }
363  
364  /* Reactions for temperature 2100.0. */  
365  card_9  
366  {  
367     mfd = 3;  
368     mtd = 1;  
369     mtname = "total";  
370  }
371  
372  card_9  
373  {  
374     mfd = 3;  
375     mtd = 2;  
376     mtname = "elastic";  
377  }
378  
379  card_9  
380  {  
381     mfd = 3;  
382     mtd = 18;  
383     mtname = "fission";  
384  }
385  
386  card_9  
387  {  
388     mfd = 3;  
389     mtd = 102;  
390     mtname = "capture";  
391  }
392  
393  card_9  
394  {  
395     mfd = 6;  
396     mtd = 2;  
397     mtname = "elastic";  
398  }
```

42
/* Terminate temperature 2100.0. */
card_9
{
    mfd = 0;
}
/* Terminate groupr. */
card_10
{
    matd = 0;
}
}
cccc
{
card_1
{
    nin = -25;
    nisot = 26;
    nbrks = 27;
    ndlay = 0; // dlayxs not wanted
}
card_2
{
    lprint = 1;
    ivers = 1;
    huse = "t2lanl njoy";
}
card_3
{
    /* hsetid does not have to be 12 chars? */
    hsetid = "ccccr tests for njoy87";
}
card_4
{
    ngroup = 50;
    nggrup = 0; // Denotes number of card_5's.
    niso = 1; // Blocking by reaction order.
    maxord = 4;
    ifopt = 1; // Blocking by reaction order.
}
card_5
{
    /* Note that the original input does not denote the first four
       variables as strings.
       What does the two '" denote? Seems a bit irregular.
    */
    hisnm = "pu238";
    habsid = "pu238";
    hident = "endfb4";
    hmat = "1050";
    imat = 1050;
    xspo = 10.89;
}
card_1
{
    nsblok = 1;
    maxup = 0; // Always zero (?).
    maxdn = 50;
    ichix = -1; // Vector (using groupr flux).
Expected NJOY Input Instructions for Test Problem 02

1 noder
2  20 -21/ ### card_1
3 reconr
4  -21 -22/ ### card_1
5 'pendf tape for pu-238 from endf/b-iv tape 404'/ ### card_2
6 1050 3/ ### card_3
7 0.005/ ### card_4
8 '94-pu-238 from endf/b tape t404'/ ### card_5
9 'processed by the njoy nuclear data processing system'/ ### card_5
10 'see original endf/b-iv tape for details of evaluation'/ ### card_5
11 0/ ### card_3
12 broadr
13  -21 -22 -23/ ### card_1
14 1050 3 0 1 0/ ### card_2
15 0.005/ ### card_3
16 300.0 900.0 2100.0/ ### card_4
17 0/ ### card_5
18 noder
19  -23 33/ ### card_1
unres
-21 -23 -24/ ### card_1
1050 3 7 1/ ### card_2
300 900 2100/ ### card_3
1.0e10 1.0e6 1.0e4 1000.0 100.0 10.0 1/ ### card_4
0/ ### card_2

group
-21 -24 0 -25/ ### card_1
1050 5 0 4 3 7 1/ ### card_2
'94-pu-238'/ ### card_3
300.0 900.0 2100.0/ ### card_4
1.0e10 1.0e6 1.0e4 1000.0 100.0 10.0 1/ ### card_5
0.1 0.025 0.8208 e06 1.4 e06/ ### card_8c
3 1 'total'/ ### card_9
3 2 'elastic'/ ### card_9
3 16 'n2n'/ ### card_9
3 17 'n3n'/ ### card_9
3 18 'fission'/ ### card_9
3 102 'capture'/ ### card_9
3 251 'nubar'/ ### card_9
3 252 'xi'/ ### card_9
3 253 'gamma'/ ### card_9
3 259 '1/n'/ ### card_9
3 51 'discrete inelastic'/ ### card_9
6 2 'elastic'/ ### card_9
6 16 'n2n'/ ### card_9
6 17 'n3e'/ ### card_9
6 18 'fission'/ ### card_9
6 51 'discrete inelastic'/ ### card_9
6 59 'continued'/ ### card_9
6 91 'continuum inelastic'/ ### card_9
0/ ### card_9
5 1 'total'/ ### card_9
5 2 'elastic'/ ### card_9
5 18 'fission'/ ### card_9
5 102 'capture'/ ### card_9
6 2 'elastic'/ ### card_9
6 0/ ### card_9
6 0/ ### card_10
6 ccccr
ccccr tests for njoy87'/ ### card_3
50 0 1 4 1/ ### card_4
'pu238' 'pu238' 'endfb4' '1050' 1050 10.89/ ### card_5
1 0 50 -1/ ### card_1
0 2.3821 e02 3.3003 e-11 1.7461 e-12 0.0 1.0e10 0.0/ ### card_4
3 6/ ### card_1
3 0 0 0 2100/ ### card_2
1.0e6 1.0e4 1000.0 100.0 10.0 1/ ### card_3
moder
-24 28/ ### card_1
stop
B.3 Test Problem 03 (tp03)

NIF Version of Test Problem 03

```c
1 reconr
2 {
3   card_1
4   {
5       nendf = 30;
6       npend = 31;
7   }
8
9   card_2
10   {
11       tlabel = "pendf tape for photon interaction cross sections from dlc7e ";
12   }
13
14   card_3
15   {
16       nat = 1;
17       ncards = 1;
18       ngrid = 0;
19   }
20
21   card_4
22   {
23       err = 0.001; // Note the C-style float format with preceding 0.
24   }
25
26   card_5
27   {
28       cards = "1-hydrogen";
29   }
30
31   card_3
32   {
33       nat = 92;
34       ncards = 1;
35       ngrid = 0;
36   }
37
38   card_4
39   {
40       err = 0.001; // Note the C-style float format with preceding 0.
41   }
42
43   card_5
44   {
45       cards = "92-uranium";
46   }
47
48   card_3
49   {
50       nat = 0;
51   }
52 }
53
gaminr
54 {
55   card_1
56   {
57       nendf = 32;
58       npend = 31;
59       ngam1 = 0;
60       ngam2 = 33;
61   }
62 }
```
```c
{  
  nath = 1;
  igg = 3;
  iwt = 3;
  lord = 4;
  iprint = 1;
}

card_3
{  
  title = "12 group photon interaction library";
}

card_6
{  
  mfd = -1;
  mtd = 0;
}

card_7
{  
  matd = 92;
}

card_6
{  
  mfd = -1;
  mtd = 0;
}

card_7
{  
  matd = 0;
}

dtfr
{  
  card_1
  {  
    nin = 33;
    nout = 34;
    npend = 31;
    npplot = 36;
  }
  
card_2
  {  
    iprint = 1;
    ifilm = 1;
    iedit = 0;
  }
  
card_3
  {  
    nlmax = 5;
    ng = 12;
    iptotl = 4;
    ipingp = 5;
    itabl = 16;
    ned = 1;
    ntherm = 0;
  }
  
card_4
  {  
    /* iptotl-3 names will be read, i.e. 4-3 = 1 in this case. */
```
edits[0] = "pheat";
}  
card_5
{  
  /* ned triplets, i.e. 1 triplet in this case. */
  jpos[0] = 1;
  mt[0] = 621;
  mult[0] = 1;
}

card_7
{
  nptabl = 0;
}
/* One card_8 for each table set desired. Empty card denotes termination
of dtfr. */
card_8
{
  hisnam = "h";
  mat = 1;
  jsigz = 1;
  dtemp = 0.0;
}
card_8
{
  hisnam = "u";
  mat = 92;
  jsigz = 1;
  dtemp = 0.0;
}
  card_8 {} // Terminate dtfr.
}
  
nmatx
{
  card_1
  {
    ngen1 = 0;
    ngen2 = 33;
    nmatx = 35;
  }
  card_2
  {
    ivers = 1;
    huse = "t2lanl njoy";
  }
  card_3
  {
    npart = 1;
    ntype = 1;
    nholl = 1;
    nmat = 2;
  }
  card_4
  {
    hsetid = "12-group photon interaction library";
  }
  
card_5

hpart = "g";
}
card_6
{
  ngrp = 12;
}
card_7
{
  htype = "gscat";
}
card_8
{
  jinp = 1;
}
card_9
{
  joutp = 1;
}
card_10
{
  hmat = "h";
  matno = 1;
  matgg = 1;
}
card_10
{
  hmat = "u";
  matno = 92;
  matgg = 92;
}
viewr
{
  /* Documentation names the first two cards as card 1. Use card 0 to denote 
   the first card, just like in plotr. */
  card_0
  {
    infile = 36;
    nps = 37;
  }
}

**Expected NJOY Input Instructions for Test Problem 03**

1 reconr
2 30 31/ ### card_1
3 'pendf tape for photon interaction cross sections from dlc7e'/ ### card_2
4 1 1 0/ ### card_3
5 0.001/ ### card_4
6 '1-hydrogen'/ ### card_5
7 92 1 0/ ### card_3
8 0.001/ ### card_4
9 '92-uranium'/ ### card_5
10 0/ ### card_3
11 gaminr
12 32 31 0 33/ ### card_1
13 1 3 3 4 1/ ### card_2
'12 group photon interaction library'/

-1 0/

92/

-1 0/

0/

dtr

33 34 31 36/

1 1 0/

5 12 4 5 16 1 0/

'pheat'/

1 621 1/

0/

'h' 1 1 0.0/

'u' 92 1 0.0/

matxsr

0 33 35/

1 't2lanl_mjoy'/

't2lanl_mjoy'/

'12-group photon interaction library'/

'g'/

'gscat'/

1/

'h' 1 1/

'u' 92 92/

viewr

36 37/

stop
B.4 Test Problem 04 (tp04)

NIF Version of Test Problem 04

```plaintext
1 node
2 {
3    card_1
4    {
5        nin = 20;
6        nout = -21;
7    }
8 }
9 reconr
10 {
11    card_1
12    {
13        nendf = -21;
14        npend = -22;
15    }
16    card_2
17    {
18        tlabel = "u-235 10\% pendf for errorr test problem from t511";
19    }
20    card_3
21    {
22        mat = 1395;
23    }
24    card_4
25    {
26        err = 0.10; // Use C-style floats.
27    }
28    card_3
29    {
30        nat = 0;
31    }
32 }
33 errorr
34 {
35    card_1
36    {
37        nendf = -21;
38        npend = -22;
39        ngout = 0;
40        nout = 23;
41        nin = 0;
42    }
43    card_2
44    {
45        natd = 1395;
46        ign = 19;
47        iwt = 3;
48        iprint = 1;
49        irelco = 1;
50    }
51    card_3
52    {
53        mprint = 0;
54        tempin = 0;
55    }
56 }
57 }
58}
59```

51
/* Test problem 04 is using a file of the endf-5 format (iverf = 5) */
card_7
{
  iread = 0;
  mfcov = 33;
}
card_12a
{
  ngn = 1;
}
card_12b
{
  egn[0] = 1.0e0;
  egn[1] = 1.0e3;
}
groupr
{
card_1
{
  nendf = -21;
  npend = -22;
  ngout1 = 0;
  ngout2 = 24;
}
card_2
{
  matb = 1395;
  ign = 3;
  igg = 0;
  iwt = 3;
  lord = 0;
  ntemp = 1;
  nsigz = 1;
  iprint = 1;
}
card_3
{
  title = "u-235 multigroup nubar calculation";
}
card_4
{
  temp[0] = 0.0;
}
card_5
{
  sigz[0] = 1.0e10;
}
card_9
{
  mfd = 3;
  std = 452;
  mtnam = "total nubar";
}
/* Terminate temperature/material with mfd = 0 as usual. */
card_9
{
  mfd = 0;
133 }
134 } /* Terminate group run with matd = 0 as usual. */
136 card_10
137 {
138    matd = 0;
139 } }  
141 { errorr
143 {
144    card_1
145 {
146       nendf = -21;
147       npend = 0;
148       ngout = 24;
149       nout = 25;
150       nin = 23;
151 } }  
153 card_2
154 {
155    matd = 1395;
156    ign = 1;
157    iwt = 2;
158    iprint = 1;
159    irelco = 1;
160 } } /* Card 3 omitted since ngout != 0. */
165 /* Test problem 04 is using a file of the endf-5 format (iverf = 5) */
166 card_7
167 {
168    iread = 0;
169    mfcov = 31;
170 } }  
172 card_12a
173 {
174    ngn = 7;
175 } }  
177 card_12b
178 {
179    egn[0] = 1.0e0;
180    egn[1] = 1.0e1;
181    egn[2] = 1.0e2;
182    egn[3] = 1.0e3;
183    egn[4] = 1.0e4;
184    egn[5] = 1.0e5;
185    egn[6] = 1.0e6;
186    egn[7] = 1.0e7;
187 } }  

Expected NJOY Input Instructions for Test Problem 04

1 moder
2 20 -21/ ### card_1
3 reconr  
4 -21 -22/ ### card_1
5 'u-235 10% pendf for errorr test problem from t511'/ ### card_2
6 1395/ ### card_3
7 0.10/ ### card_4
8 0/ ### card_3
error
-21 -22 0 23 0/ ### card_1
1395 19 3 1 1/ ### card_2
0 0/ ### card_3
0 33/ ### card_7
1/ ### card_12a
1.0e0 1.0e3/ ### card_12b

group
-21 -22 0 24/ ### card_1
1395 3 0 3 0 1 1 1/ ### card_2
'\textit{u-235 multigroup nubar calculation}'/ ### card_3
0.0/ ### card_4
1.0e10/ ### card_5
3 462 'total nubar'/ ### card_9
0/ ### card_9
0/ ### card_10
error
-21 0 24 25 23/ ### card_1
1395 1 2 1 1/ ### card_2
0 31/ ### card_7
7/ ### card_12a
1.0e0 1.0e1 1.0e2 1.0e3 1.0e4 1.0e5 1.0e6 1.0e7/ ### card_12b
stop
B.5 Test Problem 05 (tp05)

NIF Version of Test Problem 05

```plaintext
1 node
2 {
3   card_1
4   {
5     nin = 30;
6     nout = -31;
7   }
8 }
9 node
10 {
11   card_1
12   {
13     nin = -31;
14     nout = -32;
15   }
16 }
17 errorr
18 {
19   card_1
20   {
21     nendf = -31;
22     npend = -32;
23     ngout = 0;
24     nout = -33;
25   }
26   card_2
27   {
28     mats = 1306;
29     ings = 19;
30     imt = 2;
31     iprint = 1;
32   }
33   card_3
34   {
35     mprint = 0;
36     tempin = 0;
37   }
38 /* Test problem 05 is using a file of the endf-5 format (IVERF=5) */
39 card_7
40 {
41     iread = 0;
42     mfcov = 33;
43 }
44 card_12a
45 {
46     ngn = 1;
47 }
48 card_12b
49 {
50     ings = 1e-5;
51     ings = 2e7;
52 }
53 covr
54 {
55 }
```
Expected NJOY Input Instructions for Test Problem 05

1 noder  
2 30 -31/ ### card_1  
3 noder  
4 -31 -32/ ### card_1  
5 errorr  
6 -31 -32 0 -33/ ### card_1  
7 1306 19 2 1/ ### card_2  
8 0 0/ ### card_3  
9 0 33/ ### card_7  
10 1/ ### card_12a  
11 1e-5 2e7/ ### card_12b  
12 covr  
13 -33 0 34/ ### card_1  
14 1/ ### card_2  
15 / ### card_2a  
16 / ### card_3a  
17 1306/ ### card_4  
18 viewr  
19 34 35/ ### card_0  
20 stop
B.6 Test Problem 06 (tp06)

NIF Version of Test Problem 06

```c
plotr
{
  card_0
  {
    nplt = 31;
  }
  card_1 {}
  /* New axes, new page. */
  card_2
  {
    iplot = 1;
  }
  card_3
  {
    /* e should be delimited by < >? Oh well. */
    t1 = "<endf/b-v carbon";
  }
  card_3a
  {
    t2 = "<t>otal <c>ross <s>ection";
  }
  card_4
  {
    itype = 4;
  }
  card_5
  {
    el = 1e3;
    eh = 2e7;
  }
  card_5a {}
  card_6
  {
    yl = 0.5;
    yh = 10;
  }
  card_6a {}
  /* card_7 and card_7a skipped since jtype = 0. */
  card_8
  {
    ierf = 5;
    nin = 30;
    mard = 1306;
    mfd = 3;
    mtd = 1;
  }
  /* card_9 since it's a 2d plot (indicated by sign of itype in card_4) */
  card_9 {}
  /* New axes, new page. */
  card_2
  {
```

iplot = 1;
}
card_3
{ /* e should be delimited by < >? Oh well. */
   t1 = "<endf/b-v carbon";
}
card_3a
{ t2 = "(n,a) with fake data";
}
card_4
{ itype = 1;
  jtype = 0;
  igrid = 2;
  ileg = 1;
  xtag = 1.3e7;
  ytag = 0.32;
}
card_5 {}
card_5a {}
card_6 {}
card_6a {}
/* card_7 and card_7a skipped since jtype = 0 */
card_8
{ iverf = 5;
  nin = 30;
  matd = 1306;
  mfd = 3;
  mtd = 107;
}
card_9 {}
card_10
{ aleg = "<endf/b-v mat1306";
}
/* Add plot on existing axes. */
card_2
{ iplot = 2;
}
/* card 3-7 skipped since iplot = 2. */
card_8
{ iverf = 0; // Ignore rest of parameters on card.
}
card_9
{ icon = -1;
  isym = 0;
}
/* card_10 since ileg = 1. */
card_10
{
alez = "mith & mith 1914";
}
/* card_12 since iverf = 0. */
card_12{
nform = 0;
}
/* card_13 since nform = 0. */
card_13{
  xdata = 1.1e7;
ydata = 0.08;
yerr1 = 0.05;
yerr2 = 0.05;
}
card_13{
  xdata = 1.2e7;
ydata = 0.10;
yerr1 = 0.05;
yerr2 = 0.05;
}
card_13{
  xdata = 1.3e7;
ydata = 0.09;
yerr1 = 0.04;
yerr2 = 0.04;
}
card_13{
  xdata = 1.4e7;
ydata = 0.08;
yerr1 = 0.03;
yerr2 = 0.03;
}
/* Terminate card_13 with empty card. */
card_13{
/* Add plot on existing axes. */
card_2{
  iplot = 3;
}
/* Card 3-7 skipped since iplot = 3. */
card_8{
  iverf = 0; // Ignore rest of parameters on card.
}
card_9{
  icon = -1;
isym = 2;
}
/* card_10 since ileg = 1. */
card_10{
  aleg = "lack & lue 2008";
" card_12 since iverf = 0. */
card_12
{
  nform = 0;
}

" card_13 since nform = 0. */
card_13
{
xdata = 1.15e7;
ydata = 0.07;
yerr1 = 0.02;
yerr2 = 0.0;
xerr1 = 0.2e6;
xerr2 = 0.0;
}
card_13
{
xdata = 1.25e7;
ydata = 0.11;
yerr1 = 0.02;
yerr2 = 0.0;
xerr1 = 0.2e6;
xerr2 = 0.0;
}
card_13
{
xdata = 1.35e7;
ydata = 0.08;
yerr1 = 0.015;
yerr2 = 0.0;
xerr1 = 0.2e6;
xerr2 = 0.0;
}
card_13
{
xdata = 1.45e7;
ydata = 0.075;
yerr1 = 0.01;
yerr2 = 0.0;
xerr1 = 0.2e6;
xerr2 = 0.0;
}

" Terminate card_13 with empty card. */
card_13()

" New axes, new page. */
card_2
{
iplot = 1;
}
card_3
{
  /* e should be delimited by < >? Oh well. */
t1 = "<endf/b-v carbon";
}
card_3a
{
t2 = "<e>lastic <mf4>";
}
card_4
{
    itype = -1; // 3d axes.
    jtype = 2;
}
card_5 {}  
card_6 {}  
card_6a {}  
card_7 {}  
card_7a {}  
card_8 {}  
{
    iverf = 5;
    nin = 30;
    matd = 1306;
    mfd = 4;
    mtd = 2;
}
card_11 {}  
/* New axes, new page. */
card_2
{
    iplot = 1;
}
card_3
{
    t1 = "<endf/b-v l>1-6";
}
card_3a {}  
{
    t2 = "(n,2n)]a >neutron distribution";
}
card_4
{
    itype = -1;
    jtype = 2;
}
card_5 {}  
card_6 {}  
card_6a {}  
card_7 {}  
card_7a {}  
card_8 {}  
{
    iverf = 5;
    nin = 30;
    matd = 1303;
    mfd = 5;
    mtd = 24;
/* 3D plot. */
card_11 {}

/* New axes, new page. */
card_2
{
  iplot = 1;
}
card_3
{
  t1 = "<endf/b-v 1>i-6";
}
card_3a
{
  t2 = "(n,2n)]a >neutron spectra vs <E>";
}
card_4
{
  iotype = 4;
  jtype = 0;
  igrid = 2;
  ileg = 2;
}
card_5
{
  el = 10.0;
  eh = 2.0e7;
}
card_5a {}
card_6
{
  yl = 1e-11;
  yh = 1e-6;
}
card_6a
{
  ylabl = "<c>ross <s>ection (barns/e<v>)";
}
card_8
{
  iverf = 5;
  nin = 30;
  matd = 1303;
  mfd = 5;
  mtd = 24;
  temper = 0.0;
  nth = 12;
}
card_9 {}
card_10
{
  aleg = "10 <m>e<v";
}
card_10a
xtag = 1e3;
ytag = 2e-11;
xpoint = 1e2;
}

/* 2th additional plot on existing axes. */
card_2
{
  iplot = 2;
}

card_8
{
  iplot = 2;
  iplot = 2;
}

/* 3rd additional plot on existing axes. */
card_8
{
  iplot = 3;
}

/* card 10, 10a since ileg = 2 for the current axes. */
card_10
{
  aleg = "14 <m>e<v";
}

card_10a
{
  xtag = 1e4;
ytag = 2e-10;
xpoint = 1e2;
}

/* 3rd additional plot on existing axes. */
card_2
{
  iplot = 3;
}

card_8
{
  iplot = 3;
  iplot = 3;
}

/* card 10, 10a since ileg = 2 for the current axes. */
card_10
{
  aleg = "20 <m>e<v";
}

card_10a
{
  xtag = 1e5;
ytag = 2e-9;
xpoint = 4e4;
}
/* Terminate plotting job. */
card_2
{
  iplot = 99;
}
}
vieur
{
  /* Documentation names the first two cards as card 1. Use card 0 to
denote
  the first card, just like in plotr.
*/
card_0
{
infile = 31;
nps = 32;
}
}

Expected NJOY Input Instructions for Test Problem 06

plotr
1 31/ ### card_0
2 / ### card_1
3 1/ ### card_2
4 'endf/b-v carbon' / ### card_3
5 '<t>total <t>ross <t>cction' / ### card_3a
6 4/ ### card_4
7 le3 2e7/ ### card_5
8 / ### card_5a
9 0.5 10/ ### card_6
10 / ### card_6a
11 5 30 1306 3 1/ ### card_8
12 / ### card_9
13 1/ ### card_2
14 'endf/b-v carbon' / ### card_3
15 '(n,ja) with fake data' / ### card_3a
16 1 0 2 1 1.3e7 0.32/ ### card_4
17 / ### card_5
18 / ### card_5a
19 / ### card_6
20 / ### card_6a
21 5 30 1306 3 107/ ### card_8
22 / ### card_9
23 'endf/b-v mat1306' / ### card_10
24 2/ ### card_2
25 0/ ### card_8
26 -1 0/ ### card_9
27 '<s>mith & <s>mith 1914' / ### card_10
28 0/ ### card_12
29 1.1e7 0.08 0.05 0.0 0.2 e6 0.0/ ### card_13
30 1.2e7 0.10 0.05 0.0/ ### card_13
31 1.3e7 0.09 0.04 0.04/ ### card_13
32 1.4e7 0.08 0.03 0.03/ ### card_13
33 / ### card_13
34 3/ ### card_2
35 0/ ### card_8
36 -1 2/ ### card_9
37 '<b>lack & <b>lue 2008' / ### card_10
38 0/ ### card_12
39 1.15e7 0.07 0.02 0.0 0.2e6 0.0/ ### card_13
40 1.25e7 0.11 0.02 0.0 0.2e6 0.0/ ### card_13
41 1.35e7 0.08 0.015 0.0 0.2e6 0.0/ ### card_13
42 1.45e7 0.075 0.01 0.0 0.2e6 0.0/ ### card_13
43 / ### card_13
44 1/ ### card_2
45

64
B.7 Test Problem 07 (tp07)

NIF Version of Test Problem 07

```plaintext
1  noder
2  {
3     card_1
4     {
5         nin = 20;
6         nout = -21;
7     }
8  }
9  }
10  reconr
11  {
12     card_1
13     {
14         nendf = -21;
15         npend = -22;
16     }
17  }
18  }
19  {  # label = "pendf tape for u-235 from endf/b-v tape 511";
20     }
21  }
22  }
23  card_3
24  {
25     mat = 1395;
26     ncards = 3;
27  }
28  }
29  card_4
30  {
31      /* Note C-style float compared to the original declaration above. */
32      err = 0.005;
33  }
34  }
35  card_5
36  {
37      cards = "92-u-235 from endf/b-v tape 511 ";
38  }
39  }
40  card_5
41  {
42      cards = "processed by the njoy nuclear data processing system";
43  }
44  }
45  card_5
46  {
47      cards = "see original endf/b-v tape for details of evaluation";
48  }
49  /* Terminate execution of reconr with mat = 0 as usual. */
50  card_3
51  {
52      mat = 0;
53  }
54  }
55  }
56  broadr
57  {
58     card_1
59     {
60         nendf = -21;
61         nin = -22;
62         nout = -23;
63     }
64  }
```
card_2
{
mat1 = 1395;
temp2 = 1;
istart = 0;
istrap = 1;
temp1 = 0;
}
card_3
{
errthn = 0.005;
}
card_4
{
temp2[0] = 300;
}
/* Terminate execution of broadr with mat1 = 0 as usual. */
card_5
{
mat1 = 0;
}
}

heatr
{

card_1
{
nendf = -21;
nin = -23;
nout = -24;
/* nplot not supplied, defaulted to 0 */
}
card_2
{
matd = 1395;
}
}

moder
{

card_1
{
nin = -24;
nout = 28;
}
}

groupr
{

card_1
{
nendf = -21;
npend = -24;
ngout1 = 0;
ngout2 = -25;
}

card_2
{
matb = 1395;
ign = 3;
igg = 2;
iwt = 9;
}
lord = 0;
ntemp = 1;
nsigz = 1;
  iprint = 1;
}

card_3
{
  title = "u-235 from tape 511";
}
card_4
{
  temp[0] = 300.0;
}
card_5
{
  sigz[0] = 1.0e10;
}
card_9
{
  mfd = 16;
  /* mtd and mtname does not have to be supplied */
}

/* Terminate temperature/material with mfd = 0 as usual. */
card_9
{
  mfd = 0;
}

/* Terminate group run with matd = 0 as usual. */
card_10
{
  matd = 0;
}

acer
{
card_1
{
  nendf = -21;
npend = -24;
ngend = -25;
nace = 26;
ndir = 27;
}
card_2
{
  inpt = 1;
}
card_3
{
  hk = "njoy test problem 7";
}
card_5
{
  matd = 1395;
tempd = 300.0;
}
card_6

68
 Expected NJOY Input Instructions for Test Problem 07

1  noder
2  20 -21/ ### card_1
3  reconr
4  -21 -22/ ### card_1
5  'pendf tape for u-235 from endf/b-v tape 511'/ ### card_2
6  1395 3/ ### card_3
7  0.005/ ### card_4
8  '92-u-235 from endf/b-v tape 511'/ ### card_5
9  'processed by the njoy nuclear data processing system'/ ### card_6
10 'see original endf/b-v tape for details of evaluation'/ ### card_6
11 0/ ### card_3
12  broadr
13  -21 -22 -23/ ### card_1
14  1395 1 0 1 0/ ### card_2
15  0.005/ ### card_3
16  300/ ### card_4
17  0/ ### card_5
18  heatr
19  -21 -23 -24/ ### card_1
20  1395/ ### card_2
21  noder
22  -24 28/ ### card_1
23  groupr
24  -21 -24 0 -25/ ### card_1
25  1395 3 2 9 0 1 1 1/ ### card_2
26  'u-235 from tape 511'/ ### card_3
27  300.0/ ### card_4
28  1.0e10/ ### card_5
29  16/ ### card_9
30  0/ ### card_9
31  0/ ### card_10
32  acer
33  -21 -24 -25 26 27/ ### card_1
34  1/ ### card_2
35  'njoy test problem 7'/ ### card_3
36  1395 300.0/ ### card_5
37  0/ ### card_6
38  / ### card_7
39  stop
B.8 Test Problem 08 (tp08)

NIF Version of Test Problem 08

```c
1
2 noder
3 {
4     card_1
5     {
6         nin = 20;
7         nout = -21;
8     }
9 }
10 reconr
11 {
12     card_1
13     {
14         nendf = -21;
15         npend = -22;
16     }
17     card_2
18     {
19         tlabel = "pendf tape for endf/b-vi.1 28-ni-61a";
20     }
21     card_3
22     {
23         mat = 2834;
24         ncards = 1;
25         ngrid = 0;
26     }
27     card_4
28     {
29         /* Note C-style float compared to the original declaration above. */
30         err = 0.01;
31     }
32     card_5
33     {
34         cards = "28-ni-61a from endf/b-vi.1 t124 (hetrick, fu; ornl)";
35     }
36     /* Terminate execution of reconr with mat = 0 as usual. */
37     card_3
38     {
39         mat = 0;
40     }
41 }
42 broadr
43 {
44     card_1
45     {
46         nendf = -21;
47         nin = -22;
48         nout = -23;
49     }
50     card_2
51     {
52         mat1 = 2834;
53         ntemp2 = 1;
54     }
55     card_3
56     {
57     }
58 }
```

70
errthn = 0.01;

{ temp2[0] = 300;
}

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;
}

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;

/* Terminate execution of broadr with mat1 = 0 as usual. */
{ mat1 = 0;
ign = 3;
igg = 3;
iwt = 9;
lord = 4;
ntemp = 1;
nsigz = 1;
iprint = 1;
}

card_3
{
  title = "ni61a endf/b-vi.1 30x12";
}
card_4
{
temp[0] = 300;
}
card_5
{
sigz[0] = 1e10; // No trailing dots. Use C-style floats.
}
card_9
{
  mfd = 3;
  /* mtd and mtname does not have to be supplied */
}
card_9
{
  mfd = 3;
  mtd = 251;
  mtname = "mubar";
}
card_9
{
  mfd = 3;
  mtd = 252;
  mtname = "xi";
}
card_9
{
  mfd = 3;
  mtd = 253;
  mtname = "gamma";
}
card_9
{
  mfd = 6;
  /* mtd and mtname does not have to be supplied */
}
card_9
{
  mfd = 16;
  /* mtd and mtname does not have to be supplied */
/* Terminate temperature/material with mfd = 0 as usual. */
card_9
{
  mfd = 0;
}
/* Terminate group run with matd = 0 as usual. */
card_10
{
  matd = 0;
}
acer
{
card_1
{
nendf = -21;
npend = -24;
ngend = 0;
nace = 25;
ndir = 26;
}
card_2
{
iopt = 1;
iprint = 1;
ntype = 1;
}
card_3
{
hk = "28-ni-61a from endf-vi.1";
}
card_5
{
  matd = 2834;
tempd = 300.0;
}
card_6
{
  newfor = 0;
}
card_7 {}
}

Expected NJOY Input Instructions for Test Problem 08
1 moder
2 20 -21/ ### card_1
3 reconr
4 -21 -22/ ### card_1
5 'pendf tape for endf/b-vi.1 28-ni-61a'/ ### card_2
6 2834 1 0/ ### card_3
7 0.01/ ### card_4
8 '28-ni-61a from endf/b-vi.1 t124 (hetrick, fu; ornl)'/ ### card_5
9 0/ ### card_3
10 broadr
11 -21 -22 -23/ ### card_1
12 2834 1/ ### card_2
13 0.01/ ### card_3
14  300/  ### card_4
15  0/  ### card_5
16  heatr
17  -21  -23  -24/  ### card_1
18  2834  6  0  1  0  2/  ### card_2
19  302  303  304  402  443  444/  ### card_3
20  noder
21  -24  28/  ### card_1
22  groupr
23  -21  -24  0  -22/  ### card_1
24  2834  3  3  9  4  1  1/  ### card_2
25  'ni61a endf/b-vi.1 30x12'/  ### card_3
26  300/  ### card_4
27  1e10/  ### card_5
28  3/  ### card_9
29  3  251  'nubar'/  ### card_9
30  3  252  'xi'/  ### card_9
31  3  253  'gamma'/  ### card_9
32  3  259  '1/v'/  ### card_9
33  6/  ### card_9
34  16/  ### card_9
35  0/  ### card_9
36  0/  ### card_10
37  acer
38  -21  -24  0  25  26/  ### card_1
39  1  1  1/  ### card_2
40  '28-ni-61a from endf-vi.1'/  ### card_3
41  2834  300.0/  ### card_5
42  0/  ### card_6
43  /  ### card_7
44  stop
B.9 Test Problem 10 (tp10)

NIF Version of Test Problem 10

```plaintext
1 noder
2 {
3    card_1
4    {
5        nin = 20;
6        nout = -21;
7    }
8 }
9
10 reconr
11 {
12    card_1
13    {
14        nendf = -21;
15        npend = -22;
16    }
17
18    card_2
19    {
20        tlabel = "pendf tape for pu-238 from endf/b-iv tape 404";
21    }
22
23    card_3
24    {
25        mat = 1050;
26        ncards = 3;
27    }
28
29    card_4
30    {
31        /* Note C-style float compared to the original declaration above. */
32        err = 0.005;
33    }
34
35    card_5
36    {
37        cards = "94-pu-238 from endf/b tape t404";
38    }
39
40    card_6
41    {
42        cards = "processed by the njoy nuclear data processing system";
43    }
44
45    card_6
46    {
47        cards = "see original endf/b-iv tape for details of evaluation";
48    }
49
50    /* Terminate execution of reconr with mat = 0 as usual. */
51    card_3
52    {
53        mat = 0;
54    }
55 }
56
57 broadr
58 {
59    card_1
60    {
61        nendf = -21;
62        nin = -22;
63        nout = -23;
64    }
```
```plaintext
65    card_2
66    {
67        mat1 = 1050;
68        ntemp2 = 3;
69        istart = 0;
70        istrap = 1;
71        temp1 = 0;
72    }
73
74    card_3
75    {
76        errthn = 0.005;
77    }
78
79    card_4
80    {
81        temp2[0] = 300.0;
82        temp2[1] = 900.0;
83        temp2[2] = 2100.0;
84    }
85
86    /* Terminate execution of broadr with mat1 = 0 as usual. */
87    card_5
88    {
89        mat1 = 0;
90    }
91
92    unresr
93    {
94        card_1
95        {
96            nendf = -21;
97            nin = -23;
98            nout = -24;
99        }
100    card_2
101    {
102        matd = 1050;
103        ntemp = 3;
104        nsigz = 7;
105        iprint = 1;
106    }
107    card_3
108    {
109        temp[0] = 300;
110        temp[1] = 900;
111        temp[2] = 2100;
112    }
113    card_4
114    {
115        sigz[0] = 1.0e10;
116        sigz[1] = 1.0e5;
117        sigz[2] = 1.0e4;
118        sigz[3] = 1000.0;
119        sigz[4] = 100.0;
120        sigz[5] = 10.0;
121        sigz[6] = 1;
122    }
123    card_2
124    {
125        matd = 0;
126    }
```
purrr

{  
  card_1  
  {  
    nendf = -21;
    nin = -24;
    nout = -25;
  }  
  card_2  
  {  
    matd = 1050;
    ntemp = 3;
    nsigz = 7;
    nbin = 20;
    nladr = 4;
  }  
  card_3  
  {  
    temp[0] = 300;
    temp[1] = 900;
    temp[2] = 2100;
  }  
  card_4  
  {  
    sigz[0] = 1.0e10;
    sigz[1] = 1.0e6;
    sigz[2] = 1.0e4;
    sigz[3] = 1000.0;
    sigz[4] = 100.0;
    sigz[5] = 10.0;
    sigz[6] = 1;
  }  
  card_2  
  {  
    matd = 0;
  }  
  acer  
  {  
    card_1  
    {  
      nendf = -21;
      npend = -25;
      ngend = 0;
      nace = 26;
      ndir = 27;
    }  
    card_2  
    {  
      iopt = 1;
    }  
    card_3  
    {  
      hk = "njoy test problem 10";
    }  
    card_5  
    {  
      matd = 1050;
    }
tempd = 300.0;
}

moder
{
  card_1
  {
    nin = -25;
    nout = 28;
  }
}

moder
{
  card_6 {}
  card_7 {}
}

Expected NJOY Input Instructions for Test Problem 10

moder
20 -21/ ### card_1
reconr
21 -22/ ### card_1
'pendf tape for pu-238 from endf/b-iv tape 404'/ ### card_2
22 1050 3/ ### card_3
0.005/ ### card_4
94-pu-238 from endf/b tape t404'/ ### card_5
'processed by the njoy nuclear data processing system'/ ### card_5
'see original endf/b-iv tape for details of evaluation'/ ### card_5
0/ ### card_3
broadr
21 -22 -23/ ### card_1
1050 3 0 1 0/ ### card_2
0.005/ ### card_3
300.0 900.0 2100.0/ ### card_4
0/ ### card_5
unresr
21 -23 -24/ ### card_1
1050 3 7 1/ ### card_2
300 900 2100/ ### card_3
1.0e10 1.0e5 1.0e4 1000.0 100.0 10.0 1/ ### card_4
0/ ### card_2
purr
21 -24 -25/ ### card_1
1050 3 7 20 4/ ### card_2
300 900 2100/ ### card_3
1.0e10 1.0e5 1.0e4 1000.0 100.0 10.0 1/ ### card_4
0/ ### card_2
acer
21 -25 0 26 27/ ### card_1
1/ ### card_2
'njoy test problem 10'/ ### card_3
1050 300.0/ ### card_5
/ ### card_6
/ ### card_7
noder
25 28/ ### card_1
stop
B.10 Test Problem 11 (tp11)

NIF Version of Test Problem 11

moder

{ card_1
  nin = 20;
  nout = -21;
}

reconr

{ card_1
  nendf = -21;
  npend = -22;
}

card_2

  tlabel = "pendf tape for pu-238 from endf/b-iv tape 404";
}

card_3

  mat = 1050;
  ncards = 3;
}

card_4

  err = 0.005; // Use C-style floats.
}

card_5

  cards = "94-pu-238 from endf/b tape t404";
}

card_5

  cards = "processed by the njoy nuclear data processing system";
}

card_6

  cards = "see original endf/b-iv tape for details of evaluation";
}

/* Card 6 skipped since ngrid defaults to 0 in first card 3 */

/* Terminate reconr. */

{ card_3
  mat = 0;
}

broadr

{ card_1
  nendf = -21;
  nin = -22;
  nout = -23;
{ 
    nat1 = 1050; 
    ntemp2 = 3; 
    istart = 0; 
    istrap = 1; 
    temp1 = 0; 
} 

/* Terminate broadr. */
{ 
    errthn = 0.005; // Use C-style floats. 
}

{ 
    temp2[0] = 300.0; // Use C-style floats. 
    temp2[1] = 900.0; 
    temp2[2] = 2100.0; 
} 

/* Terminate unresr. */
{ 
    nendf = -21; 
    nin = -23; 
    nout = -24; 
} 

{ 
    natd = 1050; 
    ntemp = 3; 
    nsigz = 7; 
    iprint = 1; 
} 

{ 
    temp[0] = 300; 
    temp[1] = 900; 
    temp[2] = 2100; 
} 

{ 
    sigz[0] = 1.0e10; 
    sigz[1] = 1.0e5; 
    sigz[2] = 1.0e4; 
    sigz[3] = 1000.0; 
    sigz[4] = 100.0; 
    sigz[5] = 10.0; 
    sigz[6] = 1; 
} 

/* Terminate unresr. */
{ 

matd = 0;
}
}
thermr
{
    card_1
    {
        nendf = 0;
        nin = -24;
        nout = -25;
    }
    card_2
    {
        matde = 0;
        matdp = 1050;
        nbix = 8;
        ntemp = 3;
        iinc = 1;
        icoh = 0;
        natom = 1;
        ntref = 221;
        iprint = 0;
    }
    card_3
    {
        tempr[0] = 300.0; // Use C-style floats.
        tempr[1] = 900.0;
        tempr[2] = 2100.0;
    }
    card_4
    {
        tol = 0.05; // Use C-style floats.
        emax = 4.2;
    }
}
groupr
{
    card_1
    {
        nendf = -21;
        npsed = -26;
        ngout1 = 0;
        ngout2 = -26;
    }
    card_2
    {
        matb = 1050;
        ign = 9;
        igg = 0;
        iwt = 5;
        lord = 3;
        ntemp = 3;
        nsigz = 7;
        iprint = 1;
    }
    card_3
    {
        title = "94-pu-238";
    }
    card_4
/* ntemp in card_2 denotes the number of expected temperatures. */
temp[0] = 300.0;
temp[1] = 900.0;
temp[2] = 2100.0;
}

/* nsigz in card_2 denotes the number of expected sigma zeroes. */
sigz[0] = 1.0e10;
sigz[1] = 1.0e6;
sigz[2] = 1.0e4;
sigz[3] = 1000.0;
sigz[4] = 100.0;
sigz[5] = 10.0;
sigz[6] = 1;

/* Reactions for temperature 300.0. */

mfd = 3;
mtd = 1;
mtname = "total";
}

card_9

mfd = 3;
mtd = 2;
mtname = "elastic";
}

card_9

mfd = 3;
mtd = 16;
mtname = "n2n";
}

card_9

mfd = 3;
mtd = 17;
mtname = "n3n";
}

card_9

mfd = 3;
mtd = 18;
mtname = "fission";
}

card_9

mfd = 3;
mtd = 102;
mtname = "capture";
}

card_9

mfd = 3;
mtd = 221;
mtname = "free gas thermal";
{ mfd = 6; mtd = 2; mtname = "elastic"; }

card_9
{ mfd = 6; mtd = 16; mtname = "n2n"; }

card_9
{ mfd = 6; mtd = 17; mtname = "n,3n"; }

card_9
{ mfd = 6; mtd = 18; mtname = "fission"; }

card_9
{ mfd = 6; mtd = 51; mtname = "discrete inelastic"; }

card_9
{ mfd = 6; mtd = -59; mtname = "continued"; }

card_9
{ mfd = 6; mtd = 91; mtname = "continuum inelastic"; }

card_9
{ mfd = 6; mtd = 221; mtname = "free gas thermal"; }

/* Terminate temperature 300.0. */
card_9
{ mfd = 0; }

/* Reactions for temperature 900.0. */
card_9
{ mfd = 3; mtd = 1; }
mtname = "total";
}

card_9
{
    nfd = 3;
    mtd = 2;
    mtname = "elastic";
}

card_9
{
    nfd = 3;
    mtd = 18;
    mtname = "fission";
}

card_9
{
    nfd = 3;
    mtd = 102;
    mtname = "capture";
}

card_9
{
    nfd = 6;
    mtd = 221;
    mtname = "free gas thermal";
}

card_9
{
    nfd = 6;
    mtd = 2;
    mtname = "elastic";
}

card_9
{
    nfd = 6;
    mtd = 221;
    mtname = "free gas thermal";
}

/* Terminate temperature 900.0. */
card_9
{
    nfd = 0;
}

/* Reactions for temperature 2100.0. */
card_9
{
    nfd = 3;
    mtd = 1;
    mtname = "total";
}

card_9
{
    nfd = 3;
    mtd = 2;
    mtname = "elastic";
}

card_9
{
nfd = 3;
mtd = 18;
mtname = "fission";
}
card_9
{
  nfd = 3;
  mtname = "capture";
}
card_9
{
  nfd = 6;
  mtd = 2;
  mtname = "elastic";
}
card_9
{
  nfd = 6;
  mtd = 221;
  mtname = "free gas thermal";
}
card_9
{
  nfd = 3;
  mtd = 102;
  mtname = "free gas thermal";
}
card_9
{
  nfd = 6;
  mtd = 221;
  mtname = "free gas thermal";
}
/* Terminate temperature 2100.0. */
card_9
{
  nfd = 0;
}
/* Terminate group. */
card_10
{
  matd = 0;
}

winsr
{
card_1
{
  ngendf = -26;
  nout = 27;
}
card_2
{
  iprint = 1;
}
card_3
{
  mat = 1050;
  nfid = 1;
  rdfid = 1050.0;
}
card_4
nntemp = 3;
nnsigz = 7;
sgref = 1e10;
ires = 3;
sigp = 10.890;
mti = 221;
mtc = 0;
}
card_7
{
lambd[0] = 1.0;
lambd[1] = 1.0;
lambd[2] = 1.0;
lambd[3] = 1.0;
lambd[4] = 1.0;
lambd[5] = 1.0;
lambd[6] = 1.0;
lambd[7] = 1.0;
lambd[8] = 1.0;
lambd[9] = 1.0;
lambd[10] = 1.0;
lambd[11] = 1.0;
lambd[12] = 1.0;
}

Expected NJOY Input Instructions for Test Problem 11

1 moder
2 20

3 reconr
4 -21

5 'pendf tape for pu-238 from endf/b-iv tape 404'/
6 1050 3

7 0.005

8 '94-pu-238 from endf/b tape t404'/
9 'processed by the njoy nuclear data processing system'/
10 'see original endf/b-iv tape for details of evaluation'/
11 0

12 broadr
13 -21

14 1050 3 0 1 0

15 0.005

16 300.0 900.0 2100.0

17 0

18 unresr
19 -21

20 1050 3 7 1

21 300 900 2100

22 1.0e10 1.0e5 1.0e4 1000.0 100.0 10.0 1

23 0

24 thermr
25 0

26 1050 8 3 1 0 1 221 0

27 300.0 900.0 2100.0

28 0.05 4.2

29 groupr
30 -21

31 1050 9 0 5 3 3 7

32 '94-pu-238'/

33 300.0 900.0 2100.0

34 1.0e10 1.0e5 1.0e4 1000.0 100.0 10.0 1

35 3 1

36 3 2

37 3 16

38 3 17

86
39 3 18 'fission'/ ### card_9
40 3 102 'capture'/ ### card_9
41 3 221 'free gas thermal'/ ### card_9
42 6 2 'elastic'/ ### card_9
43 6 16 'n2n'/ ### card_9
44 6 17 'n,3n'/ ### card_9
45 6 18 'fission'/ ### card_9
46 6 51 'discrete inelastic'/ ### card_9
47 6 59 'continued'/ ### card_9
48 6 91 'continuum inelastic'/ ### card_9
49 6 221 'free gas thermal'/ ### card_9
50 0/ ### card_9
51 3 1 'total'/ ### card_9
52 3 2 'elastic'/ ### card_9
53 3 18 'fission'/ ### card_9
54 3 102 'capture'/ ### card_9
55 3 221 'free gas thermal'/ ### card_9
56 6 2 'elastic'/ ### card_9
57 6 221 'free gas thermal'/ ### card_9
58 0/ ### card_9
59 3 1 'total'/ ### card_9
60 3 2 'elastic'/ ### card_9
61 3 18 'fission'/ ### card_9
62 3 102 'capture'/ ### card_9
63 3 221 'free gas thermal'/ ### card_9
64 6 2 'elastic'/ ### card_9
65 6 221 'free gas thermal'/ ### card_9
66 0/ ### card_9
67 0/ ### card_10
68 wimr
69 -26 27/ ### card_1
70 1/ ### card_2
71 1050 1 1050.0/ ### card_3
72 3 7 e10 3 10.890 221 0/ ### card_4
73 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0/ ### card_7
74 stop
B.11 Test Problem 12 (tp12)

NIF Version of Test Problem 12

1 reconr
2 {
3   card_1
4   {
5     nendf = 20;
6     npend = 21;
7   }
8   card_2
9   {
10    tlabel = "pendf tape for endf/b-vi.1 28-ni-61a";
11   }
12   card_3
13   {
14     mat = 2834;
15     ncards = 1;
16     ngrid = 0;
17   }
18   card_4
19   {
20    /* Note C-style float compared to the original declaration above. */
21    err = 0.01;
22   }
23   card_5
24   {
25    cards = "28-ni-61a from endf/b-vi.1 t124 (hetrick,fu;ornl)";
26   }
27   /* Terminate execution of reconr with mat = 0 as usual. */
28   card_3
29   {
30    mat = 0;
31   }
32 }
33 gaspr
34 {
35   card_1
36   {
37     nendf = 20;
38     nin = 21;
39     nout = 22;
40   }
41 }
42 plotr
43 {
44   card_0
45   {
46     nplt = 23;
47   }
48   card_1
49   {
50     lori = 1;
51     istyle = 1;
52     size = 0.3;
53     ipcol = 2;
54   }
55   /* New axes, new page. */
65     card_2
66     {
67         iplot = 1;
68             iwcol = 3;
69     }
70     card_3
71     {
72         t1 = "<endf/b-vi n>1-61"
73     }
74     card_3a
75     {
76         t2 = "<r>esonance <c>ross <s>ections"
77     }
78     card_4
79     {
80         itype = 2;
81             jtype = 0;
82             igrid = 3;
83             ileg = 1;
84             xtag = 23e3;
85             ytag = 5e2;
86     }
87     card_5
88     {
89         e1 = 0.5e4;
90             eh = 3e4;
91             xstep = 0.5e4;
92     }
93     card_5a {}
94     card_6
95     {
96         y1 = 1e-3;
97             yh = 1e3;
98     }
99     card_6a {}
100     /* card 7 and card 7a skipped since jtype = 0. */
101     card_8
102     {
103         iverf = 6;
104             nin = 22;
105             matd = 2834;
106             mfd = 3;
107             mtd = 2;
108     }
109     /* itype is positive, resulting in 2d plot. */
110     card_9
111     {
112         icon = 0;
113             isym = 0;
114             idash = 0;
115             iccol = 3;
116             ithick = 2;
117     }
118     /* ileg = 1, resulting in card 10 but no card 10a. */
119     card_10
120     {
121         aleg = "elastic"
122     }
123     /* card 11-13 skipped since it's a 2d plot and iverf != 0. */
/* New curve; 2nd additional plot on existing axes. */
card_2
{
iplot = 2;
}
/* card 2-7 skipped since iplot = 2. */
card_8
{
    iverf = 6;
nin = 22;
    natd = 2834;
    mfd = 3;
    mtd = 102;
}
/* itype is positive on the current axes, resulting in 2d plot. */
card_9
{
    icon = 0;
    isym = 0;
    idash = 0;
    iccol = 1;
    ithick = 2;
}
/* ileg = 1 on current axes, resulting in card 10 but no card 10a. */
card_10
{
    aleg = "capture";
}
/* New axes, new page. */
card_2
{
iplot = 1;
iwcol = 7;
}
card_3
{
t1 = "<endf/b-vi n>i-61";
}
card_3a
{
t2 = "<g>as <p>rodution";
}
card_4
{
    itype = 1;
    jtype = 0;
    igrid = 3;
    ileg = 1;
}
card_5
{
el = 0;
    eh = 2e7;
    xstep = 5e6;
}
card_5a {}
card_6 {}
card_6a {}
/* card 7 and card 7a skipped since jtype = 0. */
card_8
{
  iverf = 6;
  nin = 22;
  matd = 2834;
  mfd = 3;
  mtd = 207;
  temper = 0.0;
}

/* itype is positive, resulting in 2d plot. */
card_9
{
  icon = 0;
  isym = 0;
  idash = 0;
  iccol = 1;
  ithick = 2;
}

/* ileg = 1, resulting in card 10 but no card 10a. */
card_10
{
  aleg = "hydrogen";
}

/* card 11-13 skipped since it's a 2d plot and iverf != 0. */
/* New curve; 2nd additional plot on existing axes. */
card_2
{
  iplot = 2;
}

/* card 2-7 skipped since iplot = 2. */
card_8
{
  iverf = 6;
  nin = 22;
  matd = 2834;
  mfd = 3;
  mtd = 207;
  temper = 0.0;
}

/* itype is positive on the current axes, resulting in 2d plot. */
card_9
{
  icon = 0;
  isym = 0;
  idash = 0;
  iccol = 2;
  ithick = 2;
}

/* ileg = 1 on current axes, resulting in card 10 but no card 10a. */
card_10
{
  aleg = "helium-4";
}

/* Terminate plotting job. */
card_2
{
/* Documentation names the first two cards as card 1. Use card 0 to denote the first card, just like in plotr.
*/
card_0 {
  infile = 23;
nps = 24;
}

Expected NJOY Input Instructions for Test Problem 12

1 reconr
2 20 21/ ### card_1
3 'pendf tape for endf/b-vi.1 28-ni-61a'/ ### card_2
4 2834 1/ ### card_3
5 0.01/ ### card_4
6 '28-ni-61a from endf/b-vi.1 t124 (hetrick, fu; ornl)'/ ### card_5
7 0/ ### card_3
8 gaspr
9 20 21 22/ ### card_1
10 plotr
11 23/ ### card_0
12 1 1 0.3 2/ ### card_1
13 1 3/ ### card_2
14 '<endf/b-vi n>1-61'/ ### card_3
15 '<r>esonance <c>ross <s>ections'/ ### card_3a
16 2 0 3 1 23e3 5e2/ ### card_4
17 0.5e4 3e4 0.5e4/ ### card_5
18 / ### card_5a
19 1e-3 1e3/ ### card_6
20 / ### card_6a
21 6 22 2834 3 2/ ### card_8
22 0 0 0 3 2/ ### card_9
23 'elastic'/ ### card_10
24 2/ ### card_2
25 6 22 2834 3 102/ ### card_8
26 0 0 0 1 2/ ### card_9
27 'capture'/ ### card_10
28 1 7/ ### card_2
29 '<endf/b-vi n>1-61'/ ### card_3
30 '<g>as <p>roduction'/ ### card_3a
31 1 0 3 1/ ### card_4
32 0 2e7 5e6/ ### card_5
33 / ### card_5a
34 / ### card_6
35 / ### card_6a
36 6 22 2834 3 203 0.0/ ### card_8
37 0 0 0 1 2/ ### card_9
38 'hydrogen'/ ### card_10
39 2/ ### card_2
40 6 22 2834 3 207 0.0/ ### card_8
41 0 0 0 2 2/ ### card_9
42 'helium-4'/ ### card_10
43 99/ ### card_2
44 viewr
45 23 24/ ### card_0
46 stop
B.12 Test Problem 13 (tp13)

NIF Version of Test Problem 13

```plaintext
1 noder
2 {
3     card_1
4     {
5         nin = 20;
6         nout = -21;
7     }
8 }
9
10 reconr
11 {
12     card_1
13     {
14         nendf = -21;
15         npend = -22;
16     }
17     card_2
18     {
19         tlabel = "pendf tape for endf/b-vi.1 28-ni-61a";
20     }
21     card_3
22     {
23         mat = 2834;
24         ncards = 1;
25         ngrid = 0;
26     }
27     card_4
28     {
29         err = 0.01;
30     }
31     card_5
32     {
33         cards = "28-ni-61a from endf/b-vi.1 t124 (hetrick, fu;ornl)";
34     }
35     card_3
36     {
37         nat = 0;
38     }
39 }
40
41 broadr
42 {
43     card_1
44     {
45         nendf = -21;
46         nin = -22;
47         nout = -23;
48     }
49     card_2
50     {
51         mat1 = 2834;
52         ntemp2 = 1;
53     }
54     card_3
55     {
56         errthn = 0.01;
57     }
58 }
59
60 93
```
card_4
{
  temp2[0] = 300;
}
card_5
{
  mat1 = 0;
}
heatr
{
  card_1
  {
    nendf = -21;
    nin = -23;
    nout = -24;
    /* nplot is not required */
  }
  card_2
  {
    matd = 2834;
    npk = 6;
    nqa = 0;
    ntemp = 1;
    local = 0;
    iprint = 2;
  }
card_3
  {
    /* npk = 6 -> 6 values for mtk */
    /* Note that mtk has been defined as an array */
    mtk[0] = 302;
    mtk[1] = 303;
    mtk[2] = 304;
    mtk[3] = 402;
    mtk[4] = 443;
    mtk[5] = 444;
  }
gaspr
{
  card_1
  {
    nendf = -21;
    nin = -24;
    nout = -25;
  }
}
moder
{
  card_1
  {
    nin = -25;
    nout = 28;
  }
}
acer
{
  card_1
  {
nendf = -21;
npend = -25;
ngend = 0;
face = 26;
ndir = 27;
}
card_2
{
iopt = 1;
iprint = 0;
ntype = 1;
}
card_3
{
hk = "28-ni-61a endf-vi.1 njoy99";
}
card_5
{
matd = 2834;
tempd = 300;
}
card_6 {}
card_7 {}

acer
{
card_1
{
nendf = 0;
npend = 26;
ngend = 33;
nace = 34;
ndir = 35;
}
card_2
{
iopt = 7;
iprint = 1;
ntype = 2;
}
card_3
{
hk = "28-ni-61a endf-vi.1 njoy99";
}

card_0
{
infile = 33;
nps = 36;
}

viewr

/* Documentation names the first two cards as card 1. Use card 0 to denote the first card, just like in plotr. */
card_0
{
infile = 33;
nps = 36;
}
Expected NJOY Input Instructions for Test Problem 13

1 noder
2 20 -21/ ### card_1
3 reconr
4 -21 -22/ ### card_1
5 'pendf tape for endf/b-vi.1 28-ni-61a'/ ### card_2
6 2834 1 0/ ### card_3
7 0.01/ ### card_4
8 '28-ni-61a from endf/b-vi.1 t124 (hetrick, fu; ornl)'/ ### card_5
9 0/ ### card_3
10 broadr
11 -21 -22 -23/ ### card_1
12 2834 1/ ### card_2
13 0.01/ ### card_3
14 300/ ### card_4
15 0/ ### card_5
16 heatr
17 -21 -23 -24/ ### card_1
18 2834 6 0 1 0 2/ ### card_2
19 302 303 304 402 443 444/ ### card_3
20 gaspr
21 -21 -24 -25/ ### card_1
22 noder
23 -25 28/ ### card_1
24 acer
25 -21 -25 0 26 27/ ### card_1
26 1 0 1/ ### card_2
27 '28-ni-61a endf-vi.1 njoy99'/ ### card_3
28 2834 300/ ### card_5
29 / ### card_6
30 / ### card_7
31 acer
32 0 26 33 34 35/ ### card_1
33 7 1 2/ ### card_2
34 '28-ni-61a endf-vi.1 njoy99'/ ### card_3
35 viewr
36 33 36/ ### card_0
37 stop
B.13 Test Problem 14 (tp14)

NIF Version of Test Problem 14

```c
acer
{
  card_1
  {
    endf_input = 20;
    pendf_input = 21;
    multigroup_photon_input = 0;
    ace_output = 31;
    mcnp_directory_output = 32;
  }
  card_2
  {
    acer_run_option = 1;
    print_control = 0;
    ace_output_type = 1;
    /* id suffix for zaid (default = 0.00), and
    number of iz,aw pairs to read in (default = 0) are set to their
    default values since they are not provided.
    */
  }
  card_3
  {
    description = "proton + 7-n-14 apt la150 njoy99 mcnpx";
  }
  card_5
  {
    material = 725;
    temperature = 0; // No trailing dots allowed. Use C-style floats.
  }
  /* Card 6 and 7 are empty; the default values will be used. */
  card_6 {}
  card_7 {}
}
acer
{
  card_1
  {
    endf_input = 0;
    pendf_input = 31;
    multigroup_photon_input = 33;
    ace_output = 34;
    mcnp_directory_output = 35;
  }
  card_2
  {
    acer_run_option = 7;
    print_control = 1;
    ace_output_type = 2;
  }
  card_3
  {
    description = "proton + 7-n-14 apt la150 njoy99 mcnpx";
  }
}
viewr
```

97
/* Documentation names the first two cards as card 1. Use card 0 to
denote
the first card, just like in plotr. */
card_0
{
    input = 33;
    output = 36;
}
/* The translator appends the 'stop' instruction, no need to explicitly
declare it. */

Expected NJOY Input Instructions for Test Problem 14
1  acer
2  20 21 0 31 32/ ### card_1
3  1 0 1/ ### card_2
4  'proton + 7-n-14 apt la150 njoy99 mcnpx '/ ### card_3
5  725 0/ ### card_5
6  / ### card_6
7  / ### card_7
8  acer
9  0 31 33 34 35/ ### card_1
10  7 1 2/ ### card_2
11  'proton + 7-n-14 apt la150 njoy99 mcnpx '/ ### card_3
12  viewr
13  33 36/ ### card_0
14  stop
B.14 Test Problem 17 (tp17)

NIF Version of Test Problem 17

```plaintext
1 reconr
2 {
3    card_1
4    {
5        nendf = 21;
6        npend = 41;
7    }
8
9    card_2
10    {
11        tlabel = "processing jendl-3.3 238u.";
12    }
13
14    card_3
15    {
16        mat = 9237;
17        ncards = 0;
18        ngrid = 0;
19    }
20
21    card_4
22    {
23        err = 0.001;
24    }
25
26    card_3
27    {
28        mat = 0;
29    }
30 }
31
32 broadr
33 {
34    card_1
35    {
36        nendf = 21;
37        nin = 41;
38        nout = 31;
39    }
40
41    card_2
42    {
43        mat1 = 9237;
44        ntemp2 = 1;
45        istart = 0;
46        istrap = 0;
47        temp1 = 0;
48    }
49
50    card_3
51    {
52        errthn = 0.001;
53    }
54
55    card_4
56    {
57        temp2[0] = 300.0;
58    }
59
60    card_5
61    {
62        mat1 = 0;
63    }
64 }
```

reconr
{
    card_1
    {
        nendf = 22;
        npend = 42;
    }
    card_2
    {
        tlabel = "processing jendl-3.3 235u.";
    }
    card_3
    {
        mat = 9228;
        ncards = 0;
        ngrid = 0;
    }
    card_4
    {
        err = 0.001;
    }
    card_3
    {
        mat = 0;
    }
}
broadr
{
    card_1
    {
        nendf = 22;
        nin = 42;
        nout = 32;
    }
    card_2
    {
        mat1 = 9228;
        ntemp2 = 1;
        istart = 0;
        istrap = 0;
        temp1 = 0;
    }
    card_3
    {
        errthn = 0.001;
    }
    card_4
    {
        temp2[0] = 300.0;
    }
    card_5
    {
        mat1 = 0;
    }
}
reconr
{
```c
133  card_1
134  {
135      nendf = 23;
136      npend = 43;
137  }
138  card_2
139  {
140      tlabel = "processing jendl-3.3 239pu."
141  }
142  card_3
143  {
144      mat = 9437;
145      ncards = 0;
146      ngrid = 0;
147  }
148  card_4
149  {
150      err = 0.001;
151  }
152  card_3
153  {
154      mat = 0;
155  }
156  broadr
157  {
158      card_1
159      {
160          nendf = 23;
161          nin = 43;
162          nout = 33;
163      }
```
```c
164  card_2
165  {
166      mat1 = 9437;
167      ntemp2 = 1;
168      istart = 0;
169      istrap = 0;
170      temp1 = 0;
171  }
172  card_3
173  {
174      errthn = 0.001;
175  }
176  card_4
177  {
178      temp2[0] = 300.0;
179  }
180  card_5
181  {
182      mat1 = 0;
183  }
184  }
185  groupr
186  {
187      card_1
188      {
189          nendf = 21;
190      }
191  }
```
npend = 31;
ngout1 = 0;
ngout2 = 91;
}
card_2
{
matb = 9237;
ign = 3;
igg = 0;
iwt = 6;
lord = 1;
ntemp = 1;
nsigz = 1;
iprint = 0;
}
card_3
{
title = "u-238";
}
card_4
{
temp[0] = 300.0;
}
card_5
{
sigz[0] = 1.0e10; // No trailing dots. Use C-style floats.
}
card_6
{
sigz[0] = 1.0e10; // No trailing dots. Use C-style floats.
}
card_9
{
mfd = 3;
/* mtd and mtname does not have to be supplied? */
}
card_9
{
mfd = 3;
mtd = 251;
mtname = "mubar";
}
card_9
{
mfd = 3;
mtd = 252;
mtname = "xi";
}
card_9
{
mfd = 3;
mtd = 452;
mtname = "nu";
}
card_9
{
mfd = 3;
mtd = 455;
mtname = "nu";
}
card_9
{

mfd = 3;
mtd = 456;
mtname = "nu";
}
card_9
{
 mfd = 5;
 mt = 18;
 mtname = "xi";
}
/* Terminate temperature/material with mfd = 0 as usual. */
card_9
{
 mfd = 0;
}
/* Terminate groupr run with matd = 0 as usual. */
card_10
{
 matd = 0;
}
groupr
{
card_1
{
   nendf = 22;
   npend = 32;
   ngout1 = 0;
   ngout2 = 92;
 }
card_2
{
   matb = 9228;
   ign = 3;
   igg = 0;
   iwt = 6;
   lord = 1;
   ntemp = 1;
   nsize = 1;
   iprint = 0;
 }
card_3
{
   title = "u-235";
 }
card_4
{
   temp[0] = 300.0;
 }
card_5
{
   sigz[0] = 1.0e10; // No trailing dots. Use C-style floats.
 }
card_9
{
   mfd = 3;
   /* mtd and mtname does not have to be supplied */
}
/* Terminate temperature/material with mfd = 0 as usual. */
card_9
{
mfd = 0;
}

/* Terminate groupr run with matd = 0 as usual. */
card_10
{
matd = 0;
}

groupr
{
card_1
{
nendf = 23;
npend = 33;
ngout1 = 0;
ngout2 = 93;
}
card_2
{
natb = 9437;
ign = 3;
igg = 0;
iwt = 6;
lord = 1;
ntemp = 1;
nsigz = 1;
iprint = 0;
}
card_3
{
title = "pu-239";
}
card_4
{
temp[0] = 300.0;
}
card_5
{
sigz[0] = 1.0e10; // No trailing dots. Use C-style floats.
}
card_9
{
mfd = 3;
/* mtd and mtname does not have to be supplied? */
}

/* Terminate temperature/material with mfd = 0 as usual. */
card_9
{
mfd = 0;
}

/* Terminate groupr run with matd = 0 as usual. */
card_10
{
matd = 0;
}
moder
{
  card_1
  {
    nin = 2;
    nout = 99;
  }
  card_2
  {
    tpid = "merge u235, u-238 and pu-239";
  }
  card_3
  {
    nin = 92;
    matd = 9228;
  }
  card_3
  {
    nin = 91;
    matd = 9237;
  }
  card_3
  {
    nin = 93;
    matd = 9437;
  }
  /* Terminate moder by setting nin = 0. */
  card_3
  {
    nin = 0;
  }
}
errorr
{
  card_1
  {
    nendf = 21;
    npend = 0;
    ngout = 99;
    nout = 26;
    nin = 0;
    nstan = 0;
  }
  card_2
  {
    matd = 9237;
    ign = 3;
    iwt = 6;
    iprint = 1;
  }
  /* Test problem 17 is using a file of the endf-5 format (iverf = 5) */
  card_7
  {
    iread = 2;
    mfcov = 33;
    irespr = 1;
    legord = 1;
    ifissp = -1;
Expected NJOY Input Instructions for Test Problem 17

```
1 reconr
2 21 41/ ### card_1
3 'processing jendl-3.3 238u.:'/ ### card_2
4 9237 0 0/ ### card_3
5 0.001/ ### card_4
6 0/ ### card_3
7 broadr
8 21 41 31/ ### card_1
9 9237 1 0 0 0/ ### card_2
10 0.001/ ### card_3
11 300.0/ ### card_4
12 0/ ### card_5
13 reconr
14 22 42/ ### card_1
15 'processing jendl-3.3 235u..'/ ### card_2
16 9228 0 0/ ### card_3
17 0.001/ ### card_4
18 0/ ### card_3
19 broadr
20 22 42 32/ ### card_1
21 9228 1 0 0 0/ ### card_2
22 0.001/ ### card_3
23 300.0/ ### card_4
24 0/ ### card_5
25 reconr
26 23 43/ ### card_1
27 'processing jendl-3.3 239pu..'/ ### card_2
28 9437 0 0/ ### card_3
29 0.001/ ### card_4
30 0/ ### card_3
31 broadr
32 23 43 33/ ### card_1
33 9437 1 0 0 0/ ### card_2
34 0.001/ ### card_3
35 300.0/ ### card_4
36 0/ ### card_5
37 groupr
38 21 31 0 91/ ### card_1
39 9237 3 0 6 1 1 1 0/ ### card_2
40 'u-238'/ ### card_3
41 300.0/ ### card_4
42 1.0e10/ ### card_5
43 3/ ### card_9
44 3 261 'ubar'/ ### card_9
45 3 262 'xi'/ ### card_9
```
46  3 452 'nu' / ### card_9
47  3 455 'nu' / ### card_9
48  3 466 'nu' / ### card_9
49  5 18 'xi' / ### card_9
50  0/ ### card_9
51  0/ ### card_10
52  groupr
53  22 32 0 92/ ### card_1
54  9228 3 0 6 1 1 1 0/ ### card_2
55  'u-235'/ ### card_3
56  300.0/ ### card_4
57  1.0e10/ ### card_5
58  3/ ### card_9
59  0/ ### card_9
60  0/ ### card_10
61  groupr
62  23 33 0 93/ ### card_1
63  9437 3 0 6 1 1 1 0/ ### card_2
64  'pu-239'/ ### card_3
65  300.0/ ### card_4
66  1.0e10/ ### card_5
67  3/ ### card_9
68  0/ ### card_9
69  0/ ### card_10
70  moder
71  2 99/ ### card_1
72  'merge u235, u-238 and pu-239'/ ### card_2
73  92 9228/ ### card_3
74  91 9237/ ### card_3
75  93 9437/ ### card_3
76  0/ ### card_3
77  errorr
78  21 0 99 26 0 0/ ### card_1
79  9237 3 6 l/ ### card_2
80  2 33 1 1 -1/ ### card_7
81  9228 18/ ### card_10
82  9437 18/ ### card_10
83  0/ ### card_10
84  stop