Guide to the TXL *Error Messages*

What they mean, and what to do about them

Version 10.6

July 2012

James R. Cordy

James R. Cordy

TXL Error Messages Version 10.6

July 2012

Software Technology Laboratory
School of Computing
Queen's University at Kingston
Kingston, Ontario K7L 3N6
Canada

http://www.txl.ca

Table of Contents

1. TXL Diagnostic Messages

TXL is a programming language specifically designed to support transformational programming. The basic paradigm of TXL involves transforming input to output using a set of structural transformation rules that describe by example how different parts of the input are to be changed into output. Each TXL program defines its own context free grammar according to which the input is to be parsed, and rules are constrained to preserve grammatical structure in order to guarantee a well-formed result.

TXL provides a wide range of diagnostic messages reporting errors, warnings and information about TXL programs and transformations. This guide provides a comprehensive list of all of the messages generated by the TXL interpreter, the TXL Professional compiler/interpreter, the TXL virtual machine and the TXL Professional runtime system, and detailed explanations of the meanings and appropriate actions to take to address each one.

All TXL messages are reported by a line of the form:

[*inputfile*, *txlfile* **]** **TXL***nnnnX* **:** **(** *context* **)** *message text*

Where *inputfile* is the input file being transformed, *txlfile* is the TXL main program file making the transformation, *context* identifies the specific line or part of the input file or TXL program involved, and *message text* is a short explanation of the problem. TXL messages are identified by an eight character error code of the form **TXL***nnnnX*, where *nnnn* is a unique four digit message code and *X* is one of **E**, **W** or **I**, where **E** denotes a serious or fatal error, **W** a warning, and **I** an information message.

2. How To Use This Guide

Section 3 of this guide provides a complete list of TXL messages ordered by their error code. For each message, a detailed explanation of the situation flagged by the message is given, and a practical set of recommended actions to address the situation is outlined. When a TXL message is flagged, this guide can be used to better understand what the message is intended to convey, and to provide practical help in how to get past the problem. In order to more easily find a particular message, the table of contents provides a quick index ordered by error code.

Once the message is found, the user should use the message explanation to help isolate the particular part of the program and TXL language feature involved, so that the appropriate section of the *TXL Programming Language* manual can be consulted. The recommended actions then provide a more practical view of what exactly the likely root of the problem is and how it can be addressed.

While this manual tries to anticipate most situations, new programs and new uses of TXL inevitably result in new situations that have not been seen before. If you find that the explanation or suggestions for a message are not applicable or helpful in your situation, please report the situation and your suggestions for improving it using the Bug Report page on the TXL web site, *www.txl.ca* .

3. TXL Messages Ordered by Error Code

The following messages can be generated by TXL and standalone applications created using it. The list is numerically sorted by error code. To more easily find a particular code, use the index provided in the Table of Contents.

```
TXL0101E - (TXL implementation limit) Input too large
      (total text of unique tokens > NNN chars)
```

The total length of the input text to the TXL program, counting the text of each different input item only once even if it appears many times, is greater than the TXL implementation can handle at the present size.

Recommended action: Increase TXL implementation limits using the -*size* option.
For technical reasons, if this particular problem is encountered in a compiled standalone TXL application, then the application must be completely recompiled using the larger size.

```
TXL0102E - (TXL implementation limit) Input too large
      (total number of unique tokens > NNN)
```

The total length of the input to the TXL program, in terms of input items and counting each different input item only once even if it appears many times, is greater than the TXL implementation can handle at the present size.

Recommended action: Increase TXL implementation limits using the -*size* option.
For technical reasons, if this particular problem is encountered in a compiled standalone TXL application, then the application must be completely recompiled using the larger size.

```
TXL0111E - (TXL implementation limit) Too many defined nonterminal types
      (> NNN)
```

The total number of defined nonterminals, including both *define* statements and built-in nonterminals, is larger than the TXL implementation can handle.

Recommended action: Increase TXL implementation limits using the -*size* option.

```
TXL0112E - [TYPENAME] has not been defined
```

The indicated nonterminal type is used but not defined in the TXL program.

Recommended action: Check the spelling of the name and correct any uses if necessary.
If the nonterminal name is spelled correctly, add an appropriate *define* statement for it.
If the nonterminal name that appears in this message is a TXL internal name (one that begins with 'TXL_'), file a TXL bug report.

```
TXL0121E - (TXL internal error) Fatal TXL error in backup_tree
TXL0122E - (TXL internal error) Fatal TXL error in is_empty
```

These errors indicate a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0123W define 'TYPENAME' - (Warning) Empty recursion could not be
      resolved with lookahead 'TOKEN' after NNN recursions
      (using pruning heuristic to recover)
```

The nonterminal definition 'TYPENAME' in the TXL program's grammar contains a [repeat X] where [X] derives the empty string, and TXL was unable to resolve an end to the parse while parsing at or near the indicated symbol.

Recommended action: Modify the grammar to avoid repeating nonterminals that derive the empty string, either by changing [repeat X] to [repeat X+] or by changing the definition of [X] such that it does not derive the empty string. Analyze the grammar using the *-analyze* command line option to identify other potential problems.

```
TXL0126E - Maximum parse depth exceeded
TXL0127E - Parse time limit (NNN cycles) exceeded
```

These messages indicate that the TXL parser ran out of time or stack space when trying to parse the indicated context.

Recommended action: Check for the possibility of an infinite parse by re-running using the verbose (-v) option. Analyze the grammar for other potential problems using the *-analyze* option. If necessary, modify the grammar to avoid any potentially infinite parses.
If the grammar is sound, then increase TXL limits using the *-size* option.

```
TXL0128E - (TXL internal error) Fatal TXL error NNN in parse
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0129E - Unable to create TXL profile file 'txl.pprofout'
```

This error indicates that the TXL profiler was unable to create one of its output files, probably because the user does not have permission to create a file in the current working directory.

Recommended action: Change directory to a directory where you have permission to create files, and rerun the TXL profiler.

```
TXL0130E - Fatal TXL error in parse (signal)
```

The TXL parser caught an unexpected interrupt while parsing.

Recommended action: Possibly caused by a full virtual memory swap space. Reboot the computer and try the run again. If the problem persists, save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0141E - (TXL implementation limit) Input too large
      (total length > NNN tokens)
```

The total length of the input to the TXL program, in terms of individual input items, is longer than the TXL implementation can handle at the present size.

Recommended action: Increase TXL implementation limits using the *-size* option.

```
TXL0144E - (TXL implementation limit) Input line too long
      (> 1048575 characters)
```

An line in the input is longer than the maximum that the TXL implementation can handle.

Recommended action: Check that the input file is a valid text file and reduce line size to a maximum of `1048575` (i.e., 1 Mb - 1) characters if so.

```
TXL0145E - (TXL implementation limit) TXL program line too long
      (> 254 characters)
```

A line in the TXL program source is longer than the maximum that the TXL implementation can handle (254 characters).

Recommended action: Reduce the TXL program source to a maximum line length of 254 characters.

```
TXL0149E - (TXL implementation limit) Too many source include files (> NNN)
```

The total number of source files in the input is larger than the TXL implementation can handle.

Recommended action: Increase TXL implementation limits using the *-size* option.

```
TXL0150E - Unable to find include file 'INCLUDENAME'
```

The indicated include file could not be found by TXL in the present working directory, the *Txl* subdirectory of the present working directory, any of the include directories specified using *-I* options, or the system TXL library directory.

Recommended action: Most often this is due to a misspelling of the include file name. Check the spelling of the include file name. If correct, then check to see that the file exists in one of the specified directories. Remember that TXL include file names are case sensitive.

```
TXL0151E - (TXL implementation limit) Include file nesting too deep (> NNN)
```

The depth of include files that themselves contain include statements is more than the TXL implementation can handle. A possible cause of this error is an include file with an include statement that includes the file itself or one of the files that included it, forming an infinite chain of included files.

Recommended action: Fix any infinite include chains. If there are none, then work around the problem by hand-editing some included files directly into their including file. Please report this problem as a TXL bug.

`TXL0152E - Unable to open source file 'FILENAME'`

The TXL program source file could not be found by TXL in the present working directory, the *Txl* subdirectory of the present working directory, any of the include directories specified using *-i* options, or the system TXL library directory.

Recommended action: Most often this is due to a misspelling of the TXL source file name or input file extension. Check the spelling of the source file name and that the file actually exists in one of the specified directories. If the file exists and is spelled correctly, then see the section "The 'txl' Command" of the User's Guide to TXL manual to check that you are using the command correctly. Remember that TXL source file names are case sensitive.

`TXL0153E - Preprocessor syntax error: missing #endif directive`

No matching *#end if* or *#endif* preprocessor directive was found for a *#if* or *#ifdef* directive.

Recommended action: Check that every *#if* and *#ifdef* directive in the TXL source program is balanced with a matching *#end if* or *#endif*.

`TXL0154E - (TXL implementation limit) Too many preprocessor symbols (> NNN)`

The total number of defined preprocessor symbols used in the TXL program is larger than the maximum that the TXL implementation can handle.

Recommended action: This limit is not extensible. Reduce the number of defined preprocessor symbols used in the program. Report this problem as a bug.

```
TXL0155E - (TXL implementation limit) #ifdef nesting too deep
      (> NNN levels deep)
```

The depth of preprocessor *#if* or *#ifdef* nesting used in the TXL program is larger than the maximum that the TXL implementation can handle.

Recommended action: This limit is not extensible. Reduce the number of defined preprocessor symbols used in the program. Report this problem as a bug.

```
TXL0156E - Preprocessor syntax error: too many #endif directives
      (no matching #if)
```

An extra *#end if* or *#endif* preprocessor directive was found with no matching *#if* or *#ifdef*.

Recommended action: Check that every *#if* and *#ifdef* directive in the TXL source program is balanced with a matching *#end if* or *#endif* and vice-versa.

`TXL0157E - Preprocessor syntax error: missing #endif directive`

No matching *#end if* or *#endif* preprocessor directive was found for a *#if* or *#ifdef* directive.

Recommended action: Check that every *#if* and *#ifdef* directive in the TXL source program is balanced with a matching *#end if* or *#endif*.

`TXL0158E - Preprocessor syntax error: missing symbol in #define or #undefine directive`

A *#define, #def, #undefine* or *#undef* preprocessor directive in the TXL program is malformed.

Recommended action: Check each *#define, #def, #undefine* and *#undef* in the program and correct any errors.

`TXL0159E - Preprocessor syntax error: missing symbol in #if or #elsif directive`

An *#if, #elsif* or *#elif* preprocessor directive in the TXL program is malformed.

Recommended action: Check each *#if, #elsif* and *#elif* in the program and correct any errors.

`TXL0161E - Preprocessor syntax error: #else or #elsif not nested inside #if`

An *#elsif, #elif* or *#else* preprocessor directive in the TXL program was found outside the range of a *#if* or *#ifdef*.

Recommended action: Check each *#elsif, #elif* and *#else* in the program and correct any errors.

`TXL0163E - Preprocessor directive syntax error at or near:`

A syntax error was found in the indicated preprocessor directive.

Recommended action: Correct any syntax errors in the preprocessor directive.

`TXL0164E - Syntax error - comment ends at end of file`

The input file contains a bracketed comment with no ending bracket.

Recommended action: Check that all bracketed comments are properly ended.

`TXL0165W - (Warning) Token pattern for 'TYPENAME' accepts the null string`

The pattern specified for TYPENAME in a *tokens* statement accepts empty as a token. While the program will probably run correctly, this can severely slow down parsing and should be corrected.

Recommended action: Change the token pattern to require at least one character in the token.

`TXL0166E - (TXL implementation limit) Too many token patterns (links) (> NNN)`

The total number of token patterns defined in *tokens* statements is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Work around the problem by merging multiple patterns for one token kind into a single token pattern. Please report this problem as a TXL bug.

```
TXL0167E - Syntax error in token pattern for 'TYPENAME'
      (\ or # at end of pattern)
```

In a *tokens* statement, the pattern string for a token definition is ended with one of the metacharacter escape characters '\' or '#'.

Recommended action: If the metacharacter was intended to escape the closing string quote of the pattern, add another quote to end the pattern string. If a literal '\' or '#' was intended, escape the metacharacter itself by preceding it with another '\'.

```
TXL0168W - (TXL implementation limit) Unmatched opening quote accepted as
      literal token
```

An unmatched opening quote (' or ") with no closing quote was found in the input and accepted as a literal ' or " token.

Recommended action: Make sure that the input is correct and is being correctly interpreted. If the input language is not intended to have [charlit] or [stringlit] tokens in it, disable them by overriding them in a *tokens* statement with the token pattern "".

```
TXL0169W - (Warning) Escaped character \C in token pattern for 'TYPENAME'
      is not a valid token pattern meta-character
```

The indicated escaped (i.e., proceeded by backslash) character appears in a token pattern but is not a token pattern metacharacter. The backslash is ignored and \C is taken to mean C in the token pattern.

Recommended action: If a literal character C was intended, then remove the backslash. Otherwise replace C with the intended metacharacter.

```
TXL0170E - Syntax error in token pattern for 'TYPENAME' (unbalanced () or [])
```

In a *tokens* statement, the pattern string for a token definition contains one or more unbalanced ( ) or [ ] metacharacters. Possibly a literal parenthesis or bracket was intended, in which case the metacharacters should be escaped using '\'.

Recommended action: Check that all ( ) and [ ] metacharacters are properly nested and balanced in the pattern string. If a literal was intended, escape it by preceding it with '\' in the pattern.

```
TXL0172E - Syntax error - expected 'end WORD1', got 'end WORD2'
```

In a *comments, compounds, keys* or *tokens* statement, the *end* statement does not match the statement kind. Usually this indicates a missing *end* in one or more statements.

Recommended action: Check that each of the statements has a matching *end*.

```
TXL0173E - (TXL implementation limit) Too many compound literals (> NNN)
```

The total number of compound tokens specified in the *compounds* statements of the TXL program is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Reduce the number of compound tokens specified if possible. Work around the problem by removing some of the least used compounds and surrounding them by [SPOFF] and [SPON] in the grammar instead. Please report this problem as a TXL bug.

```
TXL0174E - (TXL implementation limit) Too many keywords (> NNN)
```

The total number of keywords specified in the *keys* statements of the TXL program is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Work around the problem by removing "lesser" keywords from the list. Under normal circumstances this will not affect parsing and transformation except in cases of erroneous input. Please report this problem as a TXL bug.

```
TXL0175E - (TXL implementation limit) Too many comment conventions (> NNN)
```

The total number of comment brackets specified in the *comments* statements of the TXL program is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Reduce the number of comment brackets specified if possible. If your grammar truly needs all the comment brackets you have specified, please report this problem as a TXL bug.

```
TXL0176E - Syntax error in token pattern definition
      (expected token name, got 'TEXT')
```

In a *tokens* statement, unrecognized TEXT was found in place of a token name. The *tokens* statement must consist of a sequence of token name, token pattern pairs where each token name is a TXL identifier and each token pattern is a string literal.

Recommended action: Check the format of the *tokens* statement.

```
TXL0177E - Syntax error in token pattern definition
      (expected pattern string, got 'TEXT')
```

In a *tokens* statement, unrecognized TEXT was found in place of a pattern string. Each defined token name must be followed by a string literal giving the pattern for the token.

Recommended action: Check that each defined token name has a string literal pattern.

```
TXL0178E - (TXL implementation limit) Too many user-defined token patterns
      (> NNN)
```

The total number of token patterns defined in *tokens* statements is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Work around the problem by merging multiple patterns for one token kind into a single token pattern. Please report this problem as a TXL bug.

```
TXL0179E - (TXL implementation limit) Too many user-defined token kinds
      (> NNN)
```

The total number of different user-defined token names defined in *tokens* statements is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Work around the problem by merging one or more different token classes into one. Please report this problem as a TXL bug.

```
TXL0180E - Token pattern for [id] does not allow TXL keywords
```

In a *tokens* statement, the token pattern string given for [id] does not allow recognition of standard TXL identifiers and keywords. In order to allow processing of the TXL program itself, token definitions are constrained to allow recognition of all TXL keywords and special symbols as themselves.

Recommended action: Extend the token pattern for [id] to allow TXL keywords.

```
TXL0181E - Syntax error in token pattern for 'TYPENAME'
      (lookahead test \: must be a trailing pattern)
```

In a *tokens* statement, the pattern string for a token definition contains a lookahead constraint that is not at the end of the pattern. Lookaheads are supported only as trailing context.

Recommended action: Correct the pattern to use only a trailing lookahead constraint.
If literal ':' was intended, remove the \ escape character from the pattern.

```
TXL0191E at end of FILENAME - Syntax error at end of: TEXT
```

TXL was unable to complete a successful parse of the indicated file according to the grammar in the TXL program. The parse succeeded up until the end of the file and could not be resolved at that point.

Recommended action: Check that the indicated file is legal input to the TXL program according to the program's grammar. Most likely any errors of syntax are near the indicated position.

```
TXL0192E line NNN of FILENAME - Syntax error at or near: TEXT
```

TXL was unable to complete a successful parse of the input according to the grammar of the TXL program. The parse succeeded up until the point indicated by >>> <<< in TEXT and could not be resolved beyond that point.

Recommended action: Check that the indicated file is legal input to the TXL program according to the program's grammar. Most likely any errors of syntax are near the indicated position.

```
TXL0193E predefined function [FUNCTIONNAME], called from [RULENAME] -
      Syntax error parsing FILENAME as a [TYPENAME], at or near: TEXT
```

The indicated predefined function was unable to complete a successful parse of FILENAME according to the nonterminal definition for [TYPENAME] in the grammar of the TXL program. The parse succeeded up until the point indicated by >>> <<< in TEXT and could not be resolved beyond that point.

Recommended action: Check that the indicated file is legal input to the TXL program according to the program's grammar for [TYPENAME]. Most likely any errors of syntax are near the indicated position.

```
TXL0194E - (TXL internal error) Fatal TXL error in printPatternToken
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0195E PARTNAME of RULENAME with goal production [TYPENAME] -
      Syntax error at or near: TEXT
```

TXL was unable to complete a successful parse of the indicated pattern or replacement according to the nonterminal definition for [TYPENAME] in the grammar of the TXL program. The parse succeeded up until the point indicated by >>> <<< in TEXT and could not be resolved beyond that point.

Recommended action: Check that the pattern or replacement legally matches the nonterminal definition in the TXL program's grammar. Most likely any errors of syntax are near the indicated position.

```
TXL0196E PARTNAME of RULENAME - Parse interrupted, at or near: TEXT
```

Compilation of the TXL program was halted by a termination signal (normally ^C typed on the terminal) while parsing the indicated context. If the compile was taking much longer than expected, it is possible that the grammar is circular or infinitely recursive. Use the *-v* option to get more grammar analysis messages, and the *-Dstack* option to get more information about the parse stack.

Recommended action: Check for infinitely recursive nonterminal definitions using the *-v* option. Check the grammar for other potential problems using the *-analyze* option.

```
TXL0197E line NNN of FILENAME - Parse interrupted, at or near: TEXT
```

Parsing of the input to a TXL program was halted by a termination signal (normally ^C typed on the terminal) while parsing the indicated context. If the parse was taking much longer than expected, it is possible that the grammar is circular or infinitely recursive. Use the *-v* option to get more grammar analysis messages, and the *-Dstack* option to get more information about the parse stack.

Recommended action: Check for infinitely recursive nonterminal definitions using the *-v* option. Check the grammar for other potential problems using the *-analyze* option.

```
TXL0198E PARTNAME of RULENAME - Stack use limit (NNNk) reached,
      at or near: TEXT
```

The TXL parser ran out of stack space while trying to parse the indicated pattern or replacement in the TXL program.

Recommended action: Check for the possibility of an infinite parse by re-running using the verbose (-v) option. Check the grammar for other potential problems using the *-analyze* option. If necessary, modify the grammar to avoid any potentially infinite parses.
If the grammar is sound, then increase the TXL stack limit using the *-size* option.

```
TXL0199E line N of FILENAME - Stack use limit (NNNk) reached,
      at or near: TEXT
```

The TXL parser ran out of stack space while trying to parse the indicated context in the input.

Recommended action: Check for the possibility of an infinite parse by re-running using the verbose (-v) option. Check the grammar for other potential problems using the *-analyze* option. If necessary, modify the grammar to avoid any potentially infinite parses.
If the grammar is sound, then increase TXL stack limit using the *-size* option.

```
TXL0201W - (Warning) Type name 'TYPENAME' used as a literal identifier
      (use [TYPENAME] or 'TYPENAME instead)
```

A nonterminal type name was used as an unquoted literal identifier in a pattern or replacement in a rule or function.

Recommended action: If the identifier was really intended to be a literal identifier, quote it using a single leading quote ‘.

```
TXL0202E - (TXL internal error) Fatal TXL error in processOneKid
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0203E define 'TYPENAME' - (TXL implementation limit) Too many elements
      or alternatives in one define (maximum NNN)
```

The indicated nonterminal definition has more alternatives or items in one alternative than the TXL implementation can handle in one piece.

Recommended action: Rewrite the indicated nonterminal definition. In the case where there are too many alternatives, break the nonterminal into two alternatives each of which has half of the original alternatives. In the case of too many items in one alternative, factor the alternative using sub-defines to reduce the number of items. For technical reasons critical to TXL efficiency, this limit is not generally extensible in TXL implementations.

```
TXL0204E define 'TYPENAME' - 'list_', 'repeat_', 'opt_', 'attr_', 'lit_',
      'push_', 'pop_' and 'TXL_' name prefixes are reserved for TXL
      internal use
```

Nonterminal definition TYPENAME is named using one of the indicated prefixes. Names beginning with these prefixes are reserved by TXL for its own internal use.

Recommended action: Change the name of the nonterminal to something that does not begin with any of the reserved prefixes.

```
TXL0205E define 'TYPENAME' - Define overrides token definition for [TYPENAME]
```

Nonterminal definition TYPENAME has the same name as a token definition.

Recommended action: Change the name of the nonterminal to something that does not clash with the token name.

```
TXL0206W define 'TYPENAME' - (Warning) Define overrides previous declaration
      ('redefine' should be used if override intended)
```

A second *define* statement for the nonterminal TYPENAME was encountered in the TXL program. TXL has assumed that the new definition is intended to replace the old in the grammar.

Recommended action: If the new definition for TYPENAME was intended to override the old, use *redefine* instead. Otherwise, choose a different name for the new nonterminal.

```
TXL0207E define/redefine 'TYPENAME' - Extended define 'TYPENAME' has not
      been previously defined
```

A *redefine* or *define* statement for the nonterminal TYPENAME uses the '...' extension notation, but no previous definition for the nonterminal exists.

Recommended action: If the define/redefine was intended to extend a later definition, reorder the grammar such that the extended definition appears first. If a literal '...' was intended in the definition, quote it using a leading single quote.

```
TXL0208E define 'TYPENAME' - Empty defines not allowed - use [empty] instead
```

The nonterminal definition for TYPENAME is implicitly empty (i.e, has no items in it). Nonterminals intended to derive nothing must be explicitly specified using [empty].

Recommended action: Use [empty].

```
TXL0209E define 'TYPENAME' - Empty alternatives not allowed -
      use [empty] instead
```

The nonterminal definition for TYPENAME has an alternative which is implicitly empty (i.e, has no items in it). Empty alternatives must be explicitly specified using [empty]. This error can be due to an attempt to specify an alternative beginning with '|' as a literal terminal symbol. If '|' is intended to be a terminal symbol in the grammar, then it must be quoted using a leading single quote to distinguish it from the TXL alternative symbol.

Recommended action: If an empty alternative was intended, use [empty] to specify it explicitly. If a literal '|' was intended, then quote it using a leading single quote. Otherwise remove the extra '|' from the definition.

```
TXL0210E define 'TYPENAME' - (TXL implementation limit) Too many elements
      or alternatives in one define (maximum NNN)
```

The indicated nonterminal definition has more alternatives or items in one alternative than the TXL implementation can handle in one piece.

Recommended action: Rewrite the indicated nonterminal definition. In the case where there are too many alternatives, break the nonterminal into two alternatives each of which has half of the original alternatives. In the case of too many items in one alternative, factor the alternative using sub-defines to reduce the number of items. For technical reasons critical to TXL efficiency, this limit is not generally extensible in TXL implementations.

```
TXL0212E - Defines cannot be both pre- and post-extended in the
      same definition (split into two redefines)
```

A *redefine* or *define* statement for the nonterminal TYPENAME uses the '...' extension notation as both the first and the last alternative. Nonterminal extensions can be made either to the beginning or the end of the alternatives in the original definition, but not to both in the same extension.

Recommended action: Split the extended definition into two cascaded definitions.

```
TXL0213I define 'TYPENAME' - (Information) Optimized left recursive define
```

The TXL compiler has found an opportunity to use left-factoring to increase parse efficiency.

Recommended action: None.

```
TXL0214E define 'TYPENAME' - Definition is circular
```

The definition of TYPENAME is directly circular (i.e., consists entirely of or has an alternative consisting entirely of exactly itself).

Recommended action: Remove the direct circular reference.

```
TXL0215W define 'TYPENAME1' - (Warning) [TYPENAME1] is combinatorially
      ambiguous when parsing sequences of [TYPENAME2]
TXL0216W define 'TYPENAME1' - (Warning) [TYPENAME1] is combinatorially
      ambiguous when parsing sequences of [TYPENAME2]
```

Nonterminal TYPENAME1 derives sequences of sequences of [TYPENAME2]. This allows for a combinatorial number of different parses, and can lead to slow parsing and syntax error detection.

Recommended action: Modify the grammar to make it clear at which level such sequences are to be handled.

```
TXL0217W define 'TYPENAME1' - (Warning) [TYPENAME1] is locally ambiguous
      when parsing sequences of [TYPENAME2]
```

Nonterminal TYPENAME1 derives leading sequences of leading sequences of [TYPENAME2]. This allows for a combinatorial number of different parses, and can lead to slow parsing and syntax error detection.

Recommended action: Modify the grammar to make it clear at which level such sequences are to be handled.

```
TXL0218W - [TYPENAME] has not been defined
```

The indicated nonterminal is used but not defined in the TXL program.

Recommended action: Check the spelling of the name and correct any uses if necessary.
If the nonterminal name is spelled correctly, add an appropriate *define* statement for it.
If the nonterminal name that appears in this message is a TXL internal name (one that begins with 'TXL_'), file a TXL bug report.

```
TXL0219E - [program] has not been defined
```

The required goal nonterminal [program] has not been defined in the TXL program. The goal symbol is the nonterminal as which the entire input to the TXL program is to be parsed.

Recommended action: Add a *define* statement for the nonterminal [program].

```
TXL0222I - (Information) Analyzing the object language grammar
      (this may take a while)
```

The TXL compiler has begun analyzing the TXL program's grammar definition in detail as requested by the *-analyze* option. This can take a significant amount of time, up to several minutes depending on the grammar.

Recommended action: None.

```
TXL0223E - [push] / [pop] target must be token type
```

A context-sensitive matching nonterminal ([*push* T] or [*pop* T]) is used with a target type [T] that is not a token type. Only predefined or user token types may be used in context-sensitive matching.

Recommended action: If the target type of the context matching can be a token type, then define it in a *tokens* statement rather than a *define*. If it can not be a token then use a context-free approximation grammar (without [*push*] / [*pop*]) and use pattern matching rules to restructure to the desired parse.

```
TXL0301W PARTNAME of 'RULENAME' - (Warning) Scope of function 'FUNCTIONNAME'
      is not of target type
```

In the indicated context, the non-searching function FUNCTIONNAME is applied to a TXL variable of a type different from the pattern type of the function. Because they match their pattern only to the entire scope, it does not make sense for non-searching TXL functions to be applied to scopes of types other than the function's pattern type.

<u>Recommended action</u>: If the function was intended to search, use 'replace *' to make it a searching function, or make it a rule. If the function was not intended to search, check that the type of the scope of the function application is exactly the same as the function's pattern type.

```
TXL0302E PARTNAME of 'RULENAME1' - 'replace' rule 'RULENAME2' used as
      'where' condition
```

The rule/function RULENAME2 is declared as a transformation rule (one that uses the keyword *replace*) but is used as a condition rule in the indicated context. Condition rules (those used to test conditions in a *where* clause, and declared using the *match* keyword) are constrained to do pattern matching only - they must not construct a replacement.

<u>Recommended action</u>: If the rule is intended to be a condition rule only, change the *replace* keyword to *match* in the rule declaration and remove any *by* clause. If it is intended to be used as both a transformation rule and a condition, then annotate uses of it as a condition in *where* clauses using '?'. If it is intended to be a transformation rule only, then replace any uses of it in *where* clauses with an appropriate condition rule.

```
TXL0303E PARTNAME of 'RULENAME1' - Number of parameters to rule/function
      'RULENAME2' differs from definition or previous call
```

Rule/function RULENAME2 is declared or previously used with a different number of parameters than it is used with in the indicated context.

<u>Recommended action</u>: Check that the number of formal parameters in the rule declaration agrees with the number of actual parameters specified in each call to the rule.

```
TXL0304E PARTNAME of 'RULENAME1' - Rule/function 'RULENAME2' used with empty
      or double 'each'
```

The indicated rule/function application uses the keyword *each* incorrectly in its parameter list. *Each* can only be used when followed by at least one actual parameter that is of type [list X] or [repeat X] for some [X].

<u>Recommended action</u>: Check that at least one parameter of the appropriate kind follows *each*.

```
TXL0305E PARTNAME of 'RULENAME1' - 'each' argument of rule/function
      'RULENAME2' is not a list or repeat
```

A parameter following *each* in a call to the rule/function RULENAME2 is not of type [list X] or [repeat X} for some [X]. *Each* is defined only for actual parameters that are sequences or lists, and applies to all parameters following *each* in the parameter list.
<u>Recommended action</u>: Check that the type of all parameters following *each* are sequences or lists.

```
TXL0306E PARTNAME of 'RULENAME1' - Type of actual parameter 'VARNAME' of
      rule/function 'RULENAME2' does not agree with definition or previous
      call
```

The type of the indicated actual parameter of rule/function RULENAME2 is not the same as the type of the corresponding parameter in the rule declaration or a previous call.

Recommended action: Check that the type of the formal parameter in the rule declaration agrees with the type of the corresponding actual parameter specified in each call to the rule.

```
TXL0307W PARTNAME of 'RULENAME1' - (Warning) Literal actual parameter
      'VARNAME' of rule/function 'RULENAME2' is not quoted
```

The function or rule RULENAME2 is called in the indicated context with an identifier actual parameter that is not a TXL variable, and is therefore assumed to be intended to be a literal identifier. Often this indicates that a TXL variable name has been misspelled.

Recommended action: If the actual parameter was intended to be a literal identifier then quote it using a single leading quote '. Otherwise correct the spelling of the intended TXL variable name.

```
TXL0309E PARTNAME of 'RULENAME1' - Number of parameters passed to
      rule/function 'RULENAME2' differs from definition or previous call
```

Rule/function RULENAME2 is declared or previously used with a different number of parameters than it is used with in the indicated context.

Recommended action: Check that the number of formal parameters in the rule declaration agrees with the number of actual parameters specified in each call to the rule.

```
TXL0311E PARTNAME of 'RULENAME1'- (TXL implementation limit) Too many
      parameters passed to rule/function 'RULENAME2' (> NNN)
```

The indicated rule call implies that RULENAME2 has more parameters than the TXL implementation can handle.

Recommended action: Reprogram RULENAME2 to reduce the number of parameters by passing higher-level parameters and using deconstructs to access the original lower-level parts.
This limit is not generally extensible in TXL implementations.

```
TXL0312E PARTNAME of 'RULENAME1' - Rule/function 'RULENAME2' used with empty
      or double 'each'
```

The indicated rule/function application uses the keyword *each* incorrectly in its parameter list. *Each* can only be used when followed by at least one actual parameter that is of type [list X] or [repeat X] for some [X].

Recommended action: Check that at least one parameter of the appropriate kind follows *each*.

```
TXL0313E PARTNAME of 'RULENAME1' - 'each' argument of rule/function
      'RULENAME2' is not a list or repeat
```

A parameter following *each* in a call to the rule/function RULENAME2 is not of type [list X] or [repeat X} for some [X]. *Each* is defined only for actual parameters that are sequences or lists, and applies to all parameters following *each* in the parameter list.

Recommended action: Check that the type of all parameters following *each* are sequences or lists.

```
TXL0315W - (Warning) Type name 'TYPENAME' used as a literal identifier
      (use [TYPENAME] or 'TYPENAME instead)
```

A nonterminal type name was used as an unquoted literal identifier in a pattern or replacement in a rule or function.

Recommended action: If the identifier was really intended to be a literal identifier, quote it using a single leading quote ‘.

```
TXL0316W - (Warning) Variable name 'VARNAME' is quoted as a literal
      identifier (possibly by mistake?)
```

The TXL variable VARNAME was used as a quoted literal identifier in a pattern or replacement in a rule or function.

Recommended action: If the identifier was intended to refer to the variable, remove the leading single quote. If the identifier was really intended to be a literal identifier, then it is recommended that the name of the variable be changed to avoid any possible confusion on the part of the maintainer.

```
TXL0318E - (TXL internal error) Fatal TXL error in checkVarOrExp
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0319E PARTNAME of 'RULENAME' - (TXL implementation limit) Pattern or
      replacement too large (> NNN tokens)
```

The indicated pattern or replacement contains more items than the TXL implementation can handle in one piece.

Recommended action: This limit is not extensible. Factor the pattern or replacement into parts using TXL variables, constructs and deconstructs. If this is not practical or the problem persists, report the problem as a TXL bug.

```
TXL0320E PARTNAME of 'RULENAME' - Constructed variable has already been
      defined
```

The indicated *construct* statement binds a variable that has already been bound in the rule. TXL does not permit reassignment of local variables.

Recommended action: Change the name of the constructed variable to a new name.

```
TXL0321E PARTNAME of 'RULENAME' - Type required for exported variable
```

The indicated *export* statement exports a variable that has not yet been bound in the rule. A nonterminal type must be given for the variable.

Recommended action: Add a nonterminal type for the variable to the *export* statement.

```
TXL0322E PARTNAME of 'RULENAME' - Anonymous construct/replacement cannot be
      of type [TYPENAME]
```

The indicated anonymous *construct* statement or replacement (i.e., using '_' as the replacement) is not of an appropriate type. Anonymous constructs are defined for the types [id], [stringlit], [charlit], [number], [comment] or other tokien type, for types of the form [opt X], [repeat X] or [list X] for some type [X], and for any other type that can derive [empty].

Recommended action: Check that the type of the anonymous construct is one of the allowed set. If not, insert a new *construct* statement for a new variable with an explicit placeholder instance of the type preceding the anonymous construct, and replace the '_' in the anonymous construct with the new variable name.

```
TXL0323W - PARTNAME of 'RULENAME1' - (Warning) Typed deconstruct can never
      match
```

The target type of the indicated *deconstruct* statement is not the same as the type of the deconstructed TXL variable and thus can never match.

Recommended action: If the deconstructor was intended to search, use 'deconstruct *' to make it a searching deconstruct. If it was not intended to search, check that the type of the deconstructed variable is exactly the same as the target type of the *deconstruct*.

```
TXL0324E PARTNAME of 'RULENAME' - 'where' condition requires a rule call
```

The indicated *where* statement does not invoke a condition rule/function to implement the test.

Recommended action: Add a condition rule/function call after the variable name in the *where*.

```
TXL0325E PARTNAME of 'RULENAME' - Imported variable has already been defined
```

The indicated *import* statement imports a variable that has already been bound in the rule. TXL does not permit reassignment of local variables.

Recommended action: Change the name of the previously bound variable to a new name.

```
TXL0326E PARTNAME of 'RULENAME' - Type required for imported variable
```

The indicated *import* statement imports a variable that has not been previously imported in the rule. A nonterminal type must be given for the variable.

Recommended action: Add a nonterminal type for the variable to the *import* statement.

```
TXL0327W PARTNAME of 'RULENAME' - (Warning) Imported variable already has a
      type (import type ignored)
```

The indicated *import* statement imports a variable that has already been previously imported in the rule, and hence already has a nonterminal type.

Recommended action: Remove the nonterminal type from the *import* statement.

```
TXL0328E PARTNAME of 'RULENAME' - Type required for exported variable
```

The indicated *export* statement exports a variable that has not yet been bound in the rule. A nonterminal type must be given for the variable.

Recommended action: Add a nonterminal type for the variable to the *export* statement.

```
TXL0329W PARTNAME of 'RULENAME' - (Warning) Exported variable already has a
      type (export type ignored)
```

The indicated *export* statement exports a variable that has already been bound in the rule, and hence already has a nonterminal type.

Recommended action: Remove the nonterminal type from the *export* statement.

```
TXL0330E PARTNAME of 'RULENAME' - Exported variable has not been bound
      (export value required)
```

The indicated *export* statement binds a new variable but does not give a replacement for the variable's value.

Recommended action: Add a replacement for the variable to the *export* statement. Empty replacements are not permitted for *export* statements. If the intended value is empty, use a *construct* statement to bind the variable to an empty value, then export the constructed variable.

```
TXL0331E rule/function 'RULENAME' - Number of formal parameters does not
      agree with previous call
```

The indicated rule/function is declared with a different number of parameters than it has been used with.

Recommended action: Check that the number of formal parameters in the rule declaration agrees with the number of actual parameters specified in each call to the rule.

```
TXL0332E rule/function 'RULENAME' - Type of formal parameter 'VARNAME' does
      not agree with previous call
```

The indicated formal parameter is of a type different from the actual parameters passed to it in rule calls.

Recommended action: Check that the formal parameter and the actual parameters passed to it are all of the same nonterminal type.

```
TXL0334E rule/function 'RULENAME' - (TXL implementation limit) Too many total
      local variables in rules of TXL program (> NNN)
```

The total number of local variables in all rules is greater than the TXL implementation can handle at this size. A larger size is required for this TXL program.

Recommended action: Increase TXL limits using the -*size* option.

```
TXL0335E rule/function 'RULENAME' - (TXL implementation limit) Rule/function
      is too complex - simplify using subrules
```

The indicated rule has more total *construct*, *deconstruct*, *import*, *export* and *where* statements than the TXL implementation can handle in one rule.

Recommended action: Reprogram the indicated rule to reduce the number of constructs, deconstructs and conditions either by amalgamating or by using additional subrules.

```
TXL0336E rule/function 'RULENAME' - (TXL implementation limit) Too many
      total parameters of rules in TXL program (> NNN)
```

The total number of parameters of all rules and functions is greater than the TXL implementation can handle at this size. A larger size is required for this TXL program.

Recommended action: Increase TXL limits using the -*size* option.

```
TXL0337E rule/function 'RULENAME' - (TXL implementation limit) Too many total
      constructs, deconstructs and conditions in TXL program (> NNN)
```

The total number of *construct*, *deconstruct*, *import*, *export* and *where* statements of all rules is greater than the TXL implementation can handle at this size. A larger size is required for this TXL program.

Recommended action: Increase TXL limits using the -*size* option.

```
TXL0338E rule/function 'RULENAME' - 'replace' rule/function must have a
      replacement
```

The indicated rule/function is declared as a transformation rule (one that uses the keyword *replace*) but has no *by* clause. Transformation rules must construct a replacement.

Recommended action: If the rule is intended to be a transformation rule, add a *by* clause for the result. If the rule is intended to be a condition rule, change the *replace* keyword to *match* in the rule declaration.

```
TXL0339E rule/function 'RULENAME' - 'match' rule/function cannot have a
      replacement
```

The indicated condition rule/function contains a *by* clause. Condition rules, indicated by the *match* keyword, are constrained to do pattern matching only - they must not construct a replacement.

Recommended action: If the rule is intended to be a transformation rule, change the *match* keyword to *replace* in the rule declaration. If it is also intended to be used as a condition, annotate uses of it as a condition in *where* clauses using '?'. If it is intended to be a condition rule only, then remove the *by* clause.

```
TXL0340E rule/function 'RULENAME' - 'list_', 'repeat_', 'opt_', 'attr_',
      'lit_', 'push_', 'pop_' and 'TXL_' name prefixes are reserved for
      TXL internal use
```

Rule/function RULENAME is named using one of the indicated prefixes. Names beginning with these prefixes are reserved by TXL for its internal use.

Recommended action: Change the name of the rule/function to something that does not begin with any of the reserved prefixes.

```
TXL0341W rule/function 'RULENAME' - (Warning) Rule/function declaration
      overrides predefined function
```

Rule/function RULENAME is named using one of the TXL predefined function names, so the predefined function will not be available in this program.

Recommended action: Change the name of the rule/function to another name, unless the intention is to replace the predefined function for this program.

```
TXL0342E rule/function 'RULENAME' - Rule/function has been previously defined
```

A rule/function with the indicated name has been previously declared.

Recommended action: Change the name of one or the other of the rules involved and check all calls to see that the intended one of the two is called in each case.

```
TXL0343E rule/function 'RULENAME' - 'replace' rule/function has been
      previously used as a 'where' condition
```

The indicated rule/function is declared as a transformation rule (one that uses the keyword *replace*) but is used as a condition rule. Condition rules (those used to test conditions in a *where* clause, and declared using the *match* keyword) are constrained to do pattern matching only - they must not construct a replacement.

Recommended action: If the rule is intended to be a condition rule only, change the *replace* keyword to *match* in the rule declaration and remove any *by* clause. If it is intended to be used as both a transformation rule and a condition, then annotate uses of it as a condition in *where* clauses using '?'. If it is intended to be a transformation rule only, then replace any uses of it in *where* clauses with an appropriate condition rule.

```
TXL0344W rule/function 'RULENAME' - (Warning) 'replace' rule/function has
      target type [any] (results may not be well-formed)
```

The indicated rule/function is a *replace* rule with target type [any]. Such rules are potentially dangerous since they may invalidate the grammatical structure of the scope.

Recommended action: None. This warning is a reminder of the inherently dangerous nature of this feature.

```
TXL0348W rule/function 'RULENAME' - (Warning) Target type of function does
      not match scope of previous calls
```

The non-searching function RULENAME is declared with a pattern type different from one or more variables it is applied to in the program. Because they match their pattern only to the entire scope, it does not make sense for non-searching TXL functions to be applied to scopes of types other than the function's pattern type.

Recommended action: If the function was intended to search, use 'replace *' to make it a searching function, or make it a rule. If the function was not intended to search, check each application of the function in the program to see that the variable it is applied to is of the same type as the function's pattern type.

```
TXL0352E - [?] is not defined on predefined function [RULENAME]
```

The *replace* predefined function RULENAME is used as a condition in a *where* clause using [?RULENAME]. *Replace* predefined functions can not be used as condition functions.

Recommended action: Write an appropriate condition function to test the desired condition.

```
TXL0353E - Rule/function 'RULENAME' has not been defined
```

The indicated rule/function is used in the TXL program but never declared.

Recommended action: Check calls to the rule for correct spelling of the rule name. If the rule name is spelled correctly, add a declaration for the missing rule.

```
TXL0354E PARTNAME of 'RULENAME' - Type of imported/exported variable
      'VARNAME' does not match global variable
```

The nonterminal type given for a global variable in the indicated *import* or *export* statement is not the same as the type it is given in other rules/functions. Global variables must have the same nonterminal type throughout the program.

Recommended action: Check that all *import*s and *export*s of the global variable in the program have the same nonterminal type, and correct any that differ.

```
TXL0356W PARTNAME of 'RULENAME1' - (Warning) Scope 'VARNAME [TYPENAME]' of
      call to rule/function 'RULENAME2' can never contain a match
```

The indicated variable has a nonterminal type that can never contain the target type of the main pattern of the rule/function RULENAME2. This may not be an error if the scope has been constructed using [any] rules/functions, or if the RULENAME2 is being called only for global side effects of its pre-pattern statements.

Recommended action: Check that the type of the variable and the rule/function RULENAME2 are as intended, and correct any errors.

```
TXL0357W PARTNAME of 'RULENAME1' - (Warning) Call to rule/function
      'RULENAME2' with scope 'VARNAME [repeat TYPENAME1+]' may yield an empty
      result for embedded [repeat TYPENAME2]
```

Although the variable VARNAME is of nonterminal type [repeat TYPENAME1+], indicating that it should not be empty, it can still yield an empty result in the given context when acted on by rule/function RULENAME2, which can replace it with an empty [repeat TYPENAME2].

Recommended action: Check whether this behaviour was intended, and reprogram the grammar to make the empty choice explicit if so.

```
TXL0359E - (TXL internal error) Fatal TXL error in checkRuleCallScopes
TXL0360E - (TXL internal error) Fatal TXL error in callsRule
```

These errors indicate a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug.

```
TXL0361E - Rule/function 'main' has not been defined
```

The TXL program does not declare a rule/function named *main*. Execution of a TXL program consists of applying the main rule to the entire input parse tree. Thus a main rule is required to execute the program.

Recommended action: Write a main rule or rename an existing rule to be the main rule.

```
TXL0362E - Function 'main' can never match input type [program]
      (use rule, searching function, or target type [program] instead)
```

The *main* function of the TXL program is a non-searching function with a target type other than [program]. The input to a TXL program is always parsed as nonterminal type [program], which forms the scope of the main rule/function. TXL functions (as opposed to rules) do not search inside their scope, so the function can never match the input.

Recommended action: Change the main function to a rule or searching function, or change the target type of the function to [program].

```
TXL0363I - (Information) Analyzing the transformation rule set
```

The TXL processor has begun analyzing the ruleset as requested by the *-analyze* command line option.

Recommended action: None.

```
TXL0401E - Not enough memory to compile TXL program
      (a larger size is required for this program)
```

When compiling using *txlc* or *txl -compile*, a garbage collection was required to compile the TXL program due to a shortage of tree memory. For technical reasons, compiled TXL programs cannot be stored if a garbage collection has occurred in the TXL compiler.

Recommended action: Recompile with a larger size using the *-size* command line option.

```
TXL0402E - Load file is not a compiled TXL object file
```

This error indicates that, when using the *-load* option, the compiled*.CTxl* object file being loaded was either not in *.CTxl* format, or was in a format used by a different version of TXL.

Recommended action: Recompile the *.CTxl* file using the *-compile* option.
If the problem persists, report the problem as a TXL bug.

```
TXL0403E - (TXL internal error) TXL size does not match compiled size
```

This error indicates that, when using the *-load* option, the compiled*.CTxl* object file being loaded was found to be corrupted, or to be in a format used by another version of TXL.

Recommended action: Recompile the *.CTxl* file using the *-compile* option.
If the problem persists, report the problem as a TXL bug.

```
TXL0404E - TXL object file was compiled by a different version of TXL
```

This error indicates that, when using the *-load* option, the compiled *.CTxl* object file being loaded was compiled by a different version of TXL.

Recommended action: Recompile the *.CTxl* file using the *-compile* option.
If the problem persists, report the problem as a TXL bug.

```
TXL0405E - (TXL internal error) Synchronization error NNN on compiled file
```

This error indicates that, when using the *-load* option, either the compiled *.CTxl* object file being loaded was found to be corrupted, or the TXL loader has encountered a catastrophic failure.

Recommended action: Recompile the *.CTxl* file using the *-compile* option.
If the problem persists, report the problem as a TXL bug.

```
TXL0409E - Unable to create TXL bytecode file 'FILENAME.CTxl'
```

This error indicates that, when compiling using *txlc* or *txl -compile*, the compiled *.CTxl* object file could not be written to disk. Most likely this indicates that the directory containing the TXL source file is not writable.

Recommended action: Move the TXL source to a writeable directory.

```
TXL0501E rule/function 'RULENAME' - Maximum search depth (NNN) exceeded when
      searching for pattern match (a larger size is required for this
      transform)
```

The depth of the tree being searched for the indicated pattern is more than the TXL implementation can handle. A larger transform size is required to complete this transformation.

Recommended action: Increase TXL limits using the *-size* option.

```
TXL0502E rule/function 'RULENAME1' - (TXL implementation limit) Maximum call
      depth (NNN) exceeded when calling rule/function 'RULENAME2'
      (Probable cause: infinite recursion)
```

The depth of rule calls is more than the TXL implementation can handle. Either the transform size is too small for the transformation, or the indicated rules are involved in a nonterminating transform.

Recommended action: Increase TXL limits using the *-size* option. If a nonterminating transform is suspected, check the indicated rules and the rules they call for termination. If necessary, add appropriate termination conditions.

```
TXL0503E rule/function 'RULENAME' - Imported global variable 'VARNAME' has
      not been bound
```

The indicated *import* statement imports a global variable that has not yet be exported from a rule/function.

Recommended action: Check that the order of rule/function invocation is such that the global variable is exported before it is imported. If the import really was intended to be first, add an initial *export* for the variable to the main rule.

```
TXL0504E rule/function 'RULENAME' - Assertion on 'VARNAME' failed
```

The indicated *assert* condition failed. Assertions are required to succeed in all circumstances.

Recommended action: If the *assert* was intended to be a guard rather than an assertion, then change it to be a *where*. Otherwise, use the TXL debugger or add debugging output to investigate why the assertion has failed.

```
TXL0505E - (TXL implementation limit) 'each' limited to at most two
      parameters
```

A rule call using *each* has more than two actual parameters following the *each*. In this implementation of TXL, the number of parallel each'ed parameters is limited to two.

Recommended action: Use a recursive function to parallel deconstruct each of the three parameters into first element, rest of elements on each recursion instead.

```
TXL0506E - (TXL internal error) Fatal TXL error in
      resolveReplacementExpressions
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0507E - Repeated space failure in same replacement
      (a larger size is required for this transform)
```

The TXL garbage collector was unable to recover enough tree space to successfully continue the transformation.

Recommended action: Increase TXL limits using the *-size* option.

```
TXL0508E - (TXL internal error) Fatal TXL error in processParts
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0509E rule/function 'RULENAME' - One pass rule failed to terminate
      (probable cause: part of replacement matches pattern)
```

The available stack space was exhausted while running the transformation. In a one-pass rule, this can only mean that the rule will never terminate since the replacement is directly matching the rule's pattern.

Recommended action: Modify the pattern or add appropriate *where* conditions to the rule to avoid directly matching previous replacements.

```
TXL0510E rule/function 'RULENAME' - (TXL implementation limit) Transform
      recursion limit exceeded (Probable cause: infinite recursion, small
      size or stack limit)
```

The available stack space was exhausted while running the transformation. Normally this means that the allocated stack space was too small to run the transform, or possibly the indicated rule or the rules it calls are involved in a nonterminating transform.

Recommended action: Increase TXL limits using the *-size* option. If a nonterminating transform is suspected, check the indicated rule and the rules it calls for termination. If necessary, add appropriate termination conditions.

```
TXL0511E rule/function 'RULENAME' - Transform interrupted by user
```

Execution of a TXL program was halted by a termination signal (normally ^C typed on the terminal) while executing the indicated rule. If the run was taking much longer than expected, it is possible that the indicated rule or rules it calls are involved in a nonterminating transform.

Recommended action: Make sure that the rules involved will terminate. If necessary, add a termination condition.

```
TXL0512E rule/function 'RULENAME' - Fatal TXL error (signal)
```

The TXL transformer caught an unexpected interrupt while applying the indicated rule/function.

Recommended action: Possibly caused by a full virtual memory swap space. Reboot the computer and try the run again. If the problem persists, save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0513W - (Warning) NNN garbage recoveries were required
      (larger size recommended for improved performance)
```

A larger than usual number of garbage recoveries was required to complete the transformation. Garbage recoveries significantly slow down TXL execution.

Recommended action: Increase TXL limits using the *-size* option. If the size required seems inordinately large, use the TXL profiler *txlp* to analyze rule space usage and tune rules appropriately.

```
TXL0514E - Unable to create TXL profile file 'txl.rprofout'
```

This error indicates that the TXL profiler was unable to create one of its output files, probably because the user does not have permission to create a file in the current working directory.

Recommended action: Change directory to a directory where you have permission to create files, and rerun the TXL profiler.

```
TXL0521E - Out of tree space - NNN trees have been allocated
```

The total number of trees needed to implement the transformation is larger than the TXL implementation can handle at the present size. Normally this indicates that the TXL size is too small for the transformation, but it is also a possible indication of an infinite transform (i.e., a program that never terminates).

Recommended action: Increase TXL implementation limits using the *-size* option. If an infinite transform is suspected, double check that the rules of the program will terminate.

```
TXL0522E - Out of kid space - NNN kids have been allocated
```

The total number of kids (subtree cells) needed to implement the transformation is larger than the TXL implementation can handle at the present size. Normally this indicates that the TXL size is too small for the transformation, but it is also a possible indication of an infinite transform (i.e., a program that never terminates).

Recommended action: Increase TXL implementation limits using the *-size* option. If an infinite transform is suspected, double check that the rules of the program will terminate.

```
TXL0530E - (TXL implementation limit) Rule/function is too complex -
      simplify using subrules
```

The indicated rule has more local variables or subrule calls than the TXL implementation can handle in one rule.

Recommended action: Reprogram the indicated rule to reduce the number and size of patterns, constructs, deconstructs and conditions by using additional subrules.

```
TXL0531E - (TXL implementation limit) Too many rule/function definitions
      (> NNN)
```

The total number of rules and functions, including both built-in and programmer-defined rules and functions, is larger than the TXL implementation can handle at this size.

Recommended action: Increase TXL implementation limits using the *-size* option.

```
TXL0532E PARTNAME of 'RULENAME' - Variable 'VARNAME' has not been defined
```

In the indicated rule, TXL variable 'VARNAME' is used to but never bound, or used before it is bound. In TXL all variables must be bound before they are used.

Recommended action: Check that the indicated variable is bound before it is used.

```
TXL0533E PARTNAME of 'RULENAME' - Variable 'VARNAME' has already been defined
```

The indicated pattern binds a variable that has already been bound in the rule. TXL does not permit reassignment of local variables.

Recommended action: Change the name of the bound variable to a new name, or if it is intended to be a reference to the previously bound variable, then remove the '[TYPE]' from the reference.

```
TXL0534E - (TXL implementation limit) Too many total local variables in rules
      of TXL program (> NNN)
```

The total number of local variables in all rules is greater than the TXL implementation can handle at this size. A larger size is required for this TXL program.

Recommended action: Increase TXL limits using the *-size* option.

```
TXL0535E - (TXL implementation limit) Too many total rule calls in rules of
      TXL program (> NNN)
```

The total number of subrule calls in all rules is greater than the TXL implementation can handle at this size. A larger size is required for this TXL program.

Recommended action: Increase TXL limits using the *-size* option.

```
TXL0536E PARTNAME of 'RULENAME' - Scope of [.] predefined function is not a
      repeat
```

In the indicated context, the [.] built-in function is applied to a TXL variable that is not a [repeat]. [.] is defined only when the scope is of type [repeat X] or [repeat X+] for some [X], and the parameter is either of type [repeat X], [repeat X+] or [X].

Recommended action: Check that the scope of the function application is of type [repeat X] or [repeat X+] for some type [X].

```
TXL0537E PARTNAME of 'RULENAME' - Parameter of [.] predefined function does
      not match scope type
```

The [.] built-in function is used in the indicated context with an actual parameter that does not match the type of the TXL variable the function is applied to. [.] is defined only when the scope is of type [repeat X] or [repeat X+] for some [X], and the parameter is either of type [repeat X], [repeat X+] or [X].

Recommended action: Check that the scope and parameter of the function application are of appropriate matching types.

```
TXL0538E PARTNAME of 'RULENAME' - Scope of [,] predefined function is not a
      list
```

In the indicated context, the [,] built-in function is applied to a TXL variable that is not a list. [,] is defined only when the scope is of type [list X] or [list X+] for some [X], and the parameter is either of type [list X], [list X+] or [X].

Recommended action: Check that the scope of the function application is of type [list X] or [list X+] for some type [X].

```
TXL0539E PARTNAME of 'RULENAME' - Parameter of [,] predefined function does
      not match scope type
```

The [,] built-in function is used in the indicated context with an actual parameter that does not match the type of the TXL variable the function is applied to. [,] is defined only when the scope is of type [list X] or [list X+] for some [X], and the parameter is either of type [list X], [list X+] or [X].

Recommended action: Check that the scope and parameter of the function application are of appropriate matching types.

```
TXL0540E PARTNAME of 'RULENAME' - Scope of [+], [-], [*], [/], [div] or
      [rem] is not a number, string or token
```

In the indicated context, one of the [+], [-], [*], [/], [div] or [rem] built-in functions is applied to a TXL variable that is not of an appropriate type. [+] is defined only when both the scope and the parameter are of type [number] or other numeric type (for addition) or [id], [stringlit], [charlit], [comment] or other token type (for text concatenation). [-], [*], [/], [div] and [rem] are defined only when both the scope and the parameter are of type [number] or other nueric type.

Recommended action: Check that the scope and parameter of the function application are of appropriate matching types.

```
TXL0541E PARTNAME of 'RULENAME' - Parameter of [+], [-], [*], [/], [div] or
      [rem] predefined function does not match scope type
```

One of the [+], [-], [*], [/], [div] or [rem] built-in functions is used the indicated context with an actual parameter of a type different from the type of the TXL variable the function is applied to. [+] is defined only when both the scope and the parameter are of type [number] or other numeric type (for addition) or [id], [stringlit], [charlit], [comment] or other token type (for text concatenation). [-], [*], [/], [div] and [rem] are defined only when both the scope and the parameter are of type [number] or other numeric type.

Recommended action: Check that the parameter is of a type compatible with the TXL variable the function is applied to.

```
TXL0542E PARTNAME of 'RULENAME' - Scope of [:] predefined function is not a
      string, identifier or token
```

In the indicated context, the [:] (textual substring) built-in function is applied to a TXL variable of a type other than [id], [stringlit], [charlit], [comment] or other token type. [:] is defined only when the scope is an [id], [stringlit], [charlit], [comment] or other token type and both parameters are of type [number] or other numeric type.

<u>Recommended action</u>: Check that the scope of the function application is of type [id], [stringlit], [charlit], [comment] or other token type.

```
TXL0543E PARTNAME of 'RULENAME' - Parameters of [:] predefined function are
      not numbers
```

The [:] (textual substring) built-in function is used the indicated context with one or both parameters of a type other than [number] or other numeric type. [:] is defined only when the scope is an [id], [stringlit], [charlit], [comment] or other token type and both parameters are of type [number] or other numeric type.

<u>Recommended action</u>: Check that both parameters are of type [number] or other numeric type.

```
TXL0544E PARTNAME of 'RULENAME' - Scope of [#] predefined function is not a
      number
```

In the indicated context, the [#] (textual length) built-in function is applied to a TXL variable that is not of type [number] or other numeric type. [#] is defined only when the scope is a [number] or other numeric type and the parameter is an [id], [stringlit], [charlit], [comment] or other token type.

<u>Recommended action</u>: Check that the scope of the function application is of type [number] or other numeric type.

```
TXL0545E PARTNAME of 'RULENAME' - Parameter of [#] predefined function is
      not a string, identifier or token
```

The [#] (textual length) built-in function is used the indicated context with an actual parameter that is not of type [id], [stringlit], [charlit], [comment] or other token type. [#] is defined only for parameters of those types.

<u>Recommended action</u>: Check that the parameter is of one of the appropriate types.
If the length of a [repeat] or [list] was intended, use the [length] predefined function instead.

```
TXL0546E PARTNAME of 'RULENAME' - Parameter of [=] or [~=] predefined
      function does not match scope type
```

The [=] or [~=] built-in function is used the indicated context with an actual parameter of a type different from the type of the TXL variable the function is applied to. [=] and [~=] are defined only when both the scope and the parameter are of the same nonterminal type.

<u>Recommended action</u>: Check that the parameter is of the same type as the TXL variable the function is applied to.

```
TXL0547E PARTNAME of 'RULENAME' - Scope of [>], [<], [>=] or [<=] predefined
      function is not a number, string or identifier
```

In the indicated context, one of the [>], [<], [>=] or [<=] built-in functions is applied to a TXL variable that is not of an appropriate type. [>], [<], [>=] or [<=] are defined only when both the scope and the parameter are of type [id], [stringlit], [charlit] or [number] or other numeric type.

Recommended action: Check that the scope of the function application is of type [id], [stringlit], [charlit] or [number] or other numeric type.

```
TXL0548E PARTNAME of 'RULENAME' - Parameter of [>], [<], [>=] or [<=]
      predefined function does not match scope type
```

One of the [>], [<], [>=] or [<=] built-in functions is used the indicated context with an actual parameter of a type different from the type of the TXL variable the function is applied to. [>], [<], [>=] or [<=] are defined only when both the scope and the parameter are of type [number] or other numeric type (for numerical ordering), or [id], [stringlit] or [charlit] (for textual ordering).

Recommended action: Check that the parameter is of a type compatible with the TXL variable the function is applied to.

```
TXL0549E PARTNAME of 'RULENAME' - Scope of [^] predefined function is not a
      repeat
```

In the indicated context, the [^] built-in function is applied to a TXL variable that is not a [repeat]. [^] is defined only when the scope is of type [repeat X] for some type [X].

Recommended action: Check that the scope of the function application is of type [repeat X] for some type [X].

```
TXL0550E PARTNAME of 'RULENAME' - Parameters of [$] predefined function are
      of different types
```

The [$] built-in function is used the indicated context with a second actual parameter of a type different from the type of the first parameter. [$] is defined only when both parameters are of the exact same nonterminal type.

Recommended action: Check that the parameters are of the same type, or write an appropriate one-pass rule to do the replacement.

```
TXL0551E PARTNAME of 'RULENAME' - Scope of [!] predefined function is not an
      identifier
```

In the indicated context, the [!] (uniqueify identifier) built-in function is applied to a TXL variable that is not of type [id] or other identifier type. [!] is defined only when the scope is an [id] or other identifier type.

Recommended action: Check that the scope of [!] is of type [id] or other identifier type.

```
TXL0552E PARTNAME of 'RULENAME' - Scope of [_] predefined function is not an
      identifier
```

In the indicated context, the [_] (concatenate using underscore) built-in function is applied to a TXL variable that is not of type [id] or other identifier type. [_] is defined only when both the scope and the parameter are of type [id] or other identifier type.

Recommended action: Check that the scope of [_] is of type [id] or other identifier type.

```
TXL0553E predefined function [_], called from 'RULENAME' - Parameter of [_]
      predefined function is not an identifier
```

The [_] (concatenate using underscore) built-in function is used the indicated context with an actual parameter that is not of type [id] or other identifier type. [_] is defined only when both the scope and the parameter are of type [id] or other identifier type.

Recommended action: Check that the parameter of [_] is of type [id] or other identifier type.

```
TXL0554E predefined function [quote/unparse], called from 'RULENAME' -
      Scope of [quote] or [unparse] predefined function is not a string,
      identifier or token
```

In the indicated context, the [quote] or [unparse] built-in function is applied to a TXL variable that is not of type [id], [stringlit], [charlit], [comment] or other token type. [quote] and [unparse] are defined only for scopes of type [id], [stringlit], [charlit], [comment] or other token type.

Recommended action: Check that the scope of [quote] or [unparse] is of type [id], [stringlit], [charlit], [comment] or other token type.

```
TXL0555E PARTNAME of 'RULENAME' - Scope of [unquote] predefined function is
      not an identifier or token
```

The [unquote] built-in function is applied to a TXL variable that is not of type [id] or other unquoted token type. [unquote] is defined only when the scope is an identifier or other token type other than [charlit] and [stringlit].

Recommended action: Check that the scope of the function application is of type [id] or other unquoted token type. If conversion to another type is required, use [parse] instead.

```
TXL0556E PARTNAME of 'RULENAME' - Parameter of [unquote] predefined function
      is not a string
```

The [unquote] built-in function is used with a parameter that is not of type [stringlit] or [charlit]. [unquote] is defined only when the parameter is of type [stringlit] or [charlit].

Recommended action: Check that the parameter of the function application is of type [stringlit] or [charlit]. If conversion from another type is required, use [quote] to convert it to a [stringlit] or [charlit] first.

```
TXL0557E PARTNAME of 'RULENAME' - Parameter of [parse] predefined function
      is not a string, identifier or token
```

The [parse] (scan and parse text) built-in function is used the indicated context with an actual parameter that is not of type [id], [stringlit], [charlit], [comment] or other token type. [parse] is defined only for parameters of those types.

Recommended action: Check that the parameter is of one of the appropriate types. If not, and the text to be reparsed is limited to at most `1048575` characters, use [quote] to create a parseable string. If a more general large-scale reparse is required, use [reparse] or [write] followed by [read].

```
TXL0558E PARTNAME of 'RULENAME' - Parameter of [read] or [write] predefined
      function is not a string or identifier
```

The [read] (scan and parse file) or [write] (unparse and write text to file) built-in function is used in the indicated context with an actual parameter that is not of type [stringlit], [charlit], [id] or other identifier type. The file name must be given as a string or identifier.

Recommended action: Check that the file name parameter is given as a string or identifier.

```
TXL0559E PARTNAME of 'RULENAME' - Parameter of [getp] predefined function is
      not a string
```

The [getp] (interactive input with prompt) built-in function is used in the indicated context with an actual parameter that is not of type [stringlit] or [charlit]. The input prompt must be given as a string.

Recommended action: Check that the input prompt parameter is given as a [stringlit] or [charlit].

```
TXL0560E PARTNAME of 'RULENAME' - Parameter of [putp] predefined function is
      not a string
```

The [putp] (interactive output with format) built-in function is used in the indicated context with an actual parameter that is not of type [stringlit] or [charlit]. The output format must be given as a string.

Recommended action: Check that the output format parameter is given as a [stringlit] or [charlit].

```
TXL0561E PARTNAME of 'RULENAME' - Scope of [index] predefined function is
      not a number
```

In the indicated context, the [index] (textual substring search) built-in function is applied to a TXL variable that is not of type [number] or other numeric type. [index] is defined only when the scope is a [number] or other numeric type and the parameters are both of type [id], [stringlit], [charlit], [comment] or other token type.

Recommended action: Check that the scope of the function application is of type [number] or other numeric type.

```
TXL0562E PARTNAME of 'RULENAME' - Parameters of [index] predefined function
      are not strings, identifiers or tokens
```

In the indicated context, the [index] (textual substring search) built-in function is used with parameters that are not of type [id], [stringlit], [charlit], [comment] or other token type. [index] is defined only when the scope is a [number] or other numeric type and the parameters are both of type [id], [stringlit], [charlit], [comment] or other token type.

Recommended action: Check that the parameters of the function are of appropriate types.

```
TXL0563E PARTNAME of 'RULENAME' - Scope of [grep] predefined function is not
      a string, identifier or token
```

In the indicated context, the [grep] (textual substring containment test) built-in function is used with a scope that is not of type [id], [stringlit], [charlit], [comment] or other token type. [grep] is defined only when the scope and parameter are both of type [id], [stringlit], [charlit], [comment] or other token type.

Recommended action: Check that the scope of the function is of an appropriate type.

```
TXL0564E PARTNAME of 'RULENAME' - Parameter of [grep] predefined function is
      not a string, identifier or token
```

In the indicated context, the [grep] (textual substring containment test) built-in function is used with a parameter that is not of type [id], [stringlit], [charlit], [comment] or other token type. [grep] is defined only when the scope and parameter are both of type [id], [stringlit], [charlit], [comment] or other token type.

Recommended action: Check that the parameter of the function is of an appropriate type.

```
TXL0565E PARTNAME of 'RULENAME' - Scope of [length] predefined function is
      not a number
```

The [length] (length of a sequence) built-in function is applied to a TXL variable that is not of type [number] or other numeric type. [length] is defined only when its scope is of type [number] or other numeric type and its parameter is a [repeat] or [list].

Recommended action: Check that the scope of the function application is of type [number] or other numeric type.

```
TXL0566E PARTNAME of 'RULENAME' - Parameter of [length] predefined function
      is not a [repeat] or [list]
```

The [length] (length of a sequence) built-in function is called using an actual parameter that is not a [repeat] or [list]. [length] is defined only when its scope is of type [number] or other numereic type and its parameter is of type [repeat X] or [list X] for some [X].

Recommended action: Check that the actual parameter of the function application is a [repeat] or [list].

```
TXL0567E PARTNAME of 'RULENAME' - Scope of [select] predefined function is
      not a [repeat] or [list]
```

The [select] (subsequence) built-in function is applied to a TXL variable that is not a [repeat] or [list]. [select] is defined only when the scope is of type [repeat X] or [list X] for some [X].

Recommended action: Check that the scope of the function application is a [repeat] or [list].

```
TXL0568E PARTNAME of 'RULENAME' - Parameters of [select] predefined function
      are not numbers
```

The [select] (subsequence) built-in function is used with one or more actual parameters that are not of type [number] or other numeric type. [select] is defined only when the scope is of type [repeat X] or [list X] for some [X] and the parameters are both of type [number] or other numeric type.

Recommended action: Check that the parameters of the function application are both of type [number] or other numeric type.

```
TXL0569E PARTNAME of 'RULENAME' - Scope of [head] or [tail] predefined
      function is not a [repeat] or [list]
```

The [head] (leading subsequence) or [tail] (trailing subsequence) built-in function is applied to a TXL variable that is not a [repeat] or [list]. [head] and [tail] are defined only when the scope is of type [repeat X] or [list X] for some [X].

Recommended action: Check that the scope of the function application is a [repeat] or [list].

```
TXL0570E PARTNAME of 'RULENAME' - Parameter of [head] or [tail] predefined
      function is not a number
```

The [head] (leading subsequence) or [tail] (trailing subsequence) built-in function is used with an actual parameter that is not of type [number] or other numeric type. [head] and [tail] are defined only when the scope is of type [repeat X] or [list X] for some [X] and the parameter is of type [number] or other numeric type.

Recommended action: Check that the parameter of the function application is of type [number] or other numeric type.

```
TXL0571E PARTNAME of 'RULENAME' - Parameter of [quit] predefined function is
      not a number
```

The [quit] (immediate exit) built-in function is used with an actual parameter that is not a number. [quit] is defined only when the exit code parameter is of type [number] or other numeric type.

Recommended action: Check that the exit code parameter is of type [number] or other numeric type.

```
TXL0572E PARTNAME of 'RULENAME' - Parameter of [fget] predefined function is
      not a string filename
```

The [fget] (interactive scan and parse line from file) built-in function is used in the indicated context with a parameter that is not of type [stringlit] or [charlit]. The file name parameter must be given as a string.

Recommended action: Check that the file name parameter is given as a string.

```
TXL0573E PARTNAME of 'RULENAME' - Parameter of [fput] predefined function is
      not a string filename
```

The [fput] (interactive unparse and write line to file) built-in function is used in the indicated context with a parameter that is not of type [stringlit] or [charlit]. The file name parameter must be given as a string.

Recommended action: Check that the file name parameter is given as a string.

```
TXL0574E PARTNAME of 'RULENAME' - First parameter of [fputp] predefined
      function is not a string filename
```

The [fputp] (interactive unparse and format line to file) built-in function is used in the indicated context with a first (file name) parameter that is not of type [stringlit] or [charlit]. The file name parameter must be given as a string.

Recommended action: Check that the file name parameter is given as a string.

```
TXL0575E PARTNAME of 'RULENAME' - Second parameter of [fputp] predefined
      function is not a string
```

The [fputp] (interactive unparse and format line to file) built-in function is used in the indicated context with a second (format) parameter that is not of type [stringlit] or [charlit]. The format parameter must be given as a string.

Recommended action: Check that the format parameter is given as a string.

```
TXL0576E PARTNAME of 'RULENAME' - Parameter of [fclose] predefined function
      is not a string filename
```

The [fclose] (close interactive line file) built-in function is used in the indicated context with a parameter that is not of type [stringlit] or [charlit]. The file name parameter must be a string.

Recommended action: Check that the file name parameter is given as a string.

```
TXL0577E PARTNAME of 'RULENAME' - Parameter of [pragma] predefined function
      is not an options string
```

The [pragma] (interactively change options) built-in function is used in the indicated context with a parameter that is not of type [stringlit] or [charlit]. The options string parameter must be given as a string.

Recommended action: Check that the options string parameter is given as a string.

```
TXL0578E PARTNAME of 'RULENAME' - First parameter of [fopen] predefined
      function is not a string filename
```

The [fopen] (open interactive line file) built-in function is used in the indicated context with a first (file name) parameter that is not of type [stringlit] or [charlit]. The file name parameter must be a string.

Recommended action: Check that the file name parameter is given as a string.

```
TXL0579E PARTNAME of 'RULENAME' - Second parameter of [fopen] predefined
      function is not a string file mode
```

The [fopen] (open interactive line file) built-in function is used in the indicated context with a second (file mode) parameter that is not of type [stringlit] or [charlit]. The file mode parameter must be a string, one of “get”, “put” or “append”.

Recommended action: Check that the file mode parameter is given as a string.

```
TXL0581E PARTNAME of 'RULENAME' - Parameter of [system] predefined function
      is not a command string
```

The [system] (run system shell command) built-in function is used in the indicated context with a parameter that is not of type [stringlit] or [charlit]. The command string parameter must be given as a string.

Recommended action: Check that the command string parameter is given as a string.

```
TXL0582E PARTNAME of 'RULENAME' - Scope of [pipe] predefined function is not
      a string
```

In the indicated context, the [pipe] (run piped system shell command) built-in function is applied to a TXL variable that is not of type [stringlit] or [charlit]. The scope of the [pipe] function must be a string.

Recommended action: Check that the scope of the function application is of type [stringlit] or [charlit].

```
TXL0583E PARTNAME of 'RULENAME' - Parameter of [pipe] predefined function is
      not a command string
```

The [pipe] (run piped system shell command) built-in function is used in the indicated context with a parameter that is not of type [stringlit] or [charlit]. The command string parameter must be given as a string.

Recommended action: Check that the command string parameter is given as a string.

```
TXL0584E PARTNAME of 'RULENAME' - Scope of [typeof] predefined function is
      not of type [id]
```

The [typeof] (get nonterminal type name of the tree bound to a polymorphic variable) built-in function is used in the indicated context with a scope that is not of type [id].

Recommended action: Check that the scope of the function is of type [id].

```
TXL0585E PARTNAME of 'RULENAME' - Parameter of [istype] predefined function
      is not of type [id]
```

The [istype] (check tree bound to a polymorphic variable is of the given nonterminal type) built-in function is used in the indicated context with a parameter that is not of type [id]. The given parameter must be a nonterminal type identifier.

Recommended action: Check that the nonterminal type parameter is given as an [id].

```
TXL0586E PARTNAME of 'RULENAME' - Scope of [tolower] or [toupper] predefined
      function is not an identifier, string or token
```

In the indicated context, the [tolower] or [toupper] built-in function is used with a scope that is not of type [id], [stringlit], [charlit], [comment] or other token type. [tolower] and [toupper] are defined only when the scope is of type [id], [stringlit], [charlit], [comment] or other token type.

Recommended action: Check that the scope of the function is of an appropriate type.

```
TXL0587E PARTNAME of 'RULENAME' - Scope of [round] or [trunc] predefined
      function is not a number
```

In the indicated context, the [round] or [trunc] built-in function is used with a scope that is not of type [number] or other numeric type. [round] and [trunc] are defined only when the scope is of type [number] or other numeric type.

Recommended action: Check that the scope of the function is of a numeric type.

```
TXL0588E PARTNAME of 'RULENAME' - Scope of [gets] predefined function is not
      a string
TXL0589E PARTNAME of 'RULENAME' - Scope of [fgets] predefined function is not
      a string
```

In the indicated context, the [gets] or [fgets] built-in function is used with a scope that is not of type [stringlit or [charlit].

Recommended action: Check that the scope of the function is one of [stringlit] or [charlit].

```
TXL0590E PARTNAME of 'RULENAME' - Parameter of [fgets] predefined function is
      not a string filename
```

The [fgets] built-in function is used in the indicated context with a parameter that is not of type [stringlit] or [charlit]. The file name parameter must be a string.

Recommended action: Check that the file name parameter is given as a string.

```
TXL0591E external function [FUNCTIONNAME] called from [RULENAME] - Function
      is not implemented
```

The indicated external function is not implemented in the *user.c* file of the local version of TXL, or has not been linked into the TXL library.

Recommended action: Check that the function name is spelled correctly. If the function name is spelled correctly and should be implemented, report this problem to your local TXL administrator.

```
TXL0592E predefined function [FUNCTIONNAME] called from [RULENAME] -
      Division by zero
```

In the indicated context, the [/], [div] or [rem] built-in function was called with a parameter that has value zero. Division is undefined for a divisor of zero.

Recommended action: Modify the logic of the program to avoid division by zero.

```
TXL0592E predefined function [parse] called from [RULENAME] -
      Scope of [parse] predefined function is literal (probable cause:
      scope is [token], use [repeat token] instead)
```

In the indicated context, the [parse] built-in function was applied to a TXL variable bound to a literal token. This can happen when the scope of the function is of type [token].

Recommended action: Check that [parse] has been applied to a scope of an appropriate type. If the type of the scope is [token], use [repeat token] instead.

```
TXL0592E predefined function [FUNCTIONNAME] called from [RULENAME] -
      Unable to open input file 'FILENAME' (too many open files)
```

In the indicated context, the indicated built-in function was unable to open its input file because too many files are open in the program.

Recommended action: Use [fclose] to close files as they are no longer needed. This limit is not extensible - if no files can be closed, modify program logic to reduce the need for so many simultaneously open files.

```
TXL0592E predefined function [FUNCTIONNAME] called from [RULENAME] -
      Unable to open input file 'FILENAME'
```

In the indicated context, the indicated built-in function was unable to open its input file.

Recommended action: Check that the file name is spelled correctly. Remember that TXL file names are case sensitive. Check that the file exists and is readable. If the problem persists, report this as a TXL bug.

```
TXL0592E predefined function [FUNCTIONNAME] called from [RULENAME] -
      Unable to open output file 'FILENAME' (too many open files)
```

In the indicated context, the indicated built-in function was unable to open its output file because too many files are open in the program.

Recommended action: Use [fclose] to close files as they are no longer needed. This limit is not extensible - if no files can be closed, modify program logic to reduce the need for so many simultaneously open files.

```
TXL0592E predefined function [FUNCTIONNAME] called from [RULENAME] -
      Unable to open output file 'FILENAME'
```

In the indicated context, the indicated built-in function was unable to create its output file. Most often this is because you do not have write permission in the present working directory or the directory where the file is to be created.

Recommended action: Check that the program is being run in the intended directory and that the permissions on the directory where the file is to be created allow writing.

```
TXL0592E predefined function [fopen] called from [RULENAME] -
      Unable to open file 'FILENAME' (already open)
```

In the indicated context, the [fopen] built-in function was unable to open the indicated file because it was already open.

Recommended action: Check that the program does not try to open the same file twice.

```
TXL0592E predefined function [fopen] called from [RULENAME] -
      Unable to open file 'FILENAME' (too many open files)
```

In the indicated context, the [fopen] built-in function was unable to open the indicated file because too many files are open in the program.

Recommended action: Use [fclose] to close files as they are no longer needed. This limit is not extensible - if no files can be closed, modify program logic to reduce the need for so many simultaneously open files.

```
TXL0592E predefined function [fopen] called from [RULENAME] -
      Unable to open file 'FILENAME'
```

In the indicated context, the [fopen] built-in function was unable to open the indicated file in the mode specified.

Recommended action: If the mode is "get", check that the indicated file exists and is readable. If the mode is "put" or "append", check that the indicated file is writable or createable in the current directory.

```
TXL0592E predefined function [fopen] called from [RULENAME] -
      Unknown open mode 'MODE' (must be one of get, put, append)
```

In the indicated context, the [fopen] built-in function was called with a file mode other than "get", "put" or "append".

Recommended action: Check that the file mode is one of "get", "put" or "append".

```
TXL0592E predefined function [system] called from [RULENAME] -
      [system] is not available on this platform
TXL0592E predefined function [pipe] called from [RULENAME] -
      [pipe] is not available on this platform
```

The environment in which TXL is being run does not support the *system()* library function used by the [system] and [pipe] predefined functions.

Recommended action: Run the program in a Unix, Linux , Windows, MacOSX or other environment that supports the *system()* library function.

```
TXL0911W - (Warning) Stack limit less than recommended for TXL size
      (probable cause: shell stack limit)
```

The available stack segment memory is smaller than the preferred stack size for the current TXL transform size. On UNIX and Linux systems this normally means that the default per-process stack limit on the system is too small to run TXL at the current transform size.

Recommended action: On UNIX and Linux, use the command 'unlimit stacksize' or 'ulimit -s hard' to remove the limit.

```
TXL0912E - Stack limit less than minimum for TXL size
      (probable cause: shell stack limit)
```

The available stack segment memory is smaller than the minimum required to run TXL (approx. 500 kbytes). On UNIX and Linux systems, this normally means that the default per-process stack limit on the system is too small to run TXL.

Recommended action: On UNIX and Linux, use the command 'unlimit stacksize' or 'ulimit -s hard' to remove the limit.

```
TXL0913I - (Information) Recursion stack limit reduced from NNNNNN
      to MMMMMM to fit
```

The available stack segment memory is smaller than the preferred stack size for the current TXL transform size and the TXL recursion stack limit has been reduced to fit in the available space. On UNIX and Linux systems this normally means that the default per-process stack limit on the system is too small to run TXL at the current transform size.

Recommended action: On UNIX and Linux, use the command 'unlimit stacksize' or 'ulimit -s hard' to remove the limit.

```
TXL0921E - (TXL internal error) Garbage collection failure, variable'VARNAME'
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0922E PARTNAME of 'RULENAME' - Garbage recovery unable to recover enough
      space to continue (a larger size is required for this transform)
```

The TXL garbage collector was unable to recover enough tree space to successfully continue the transformation from the indicated point.

Recommended action: Increase TXL limits using the -*size* option. If a larger size does not solve the problem, check to see that the indicated rule is not involved in an unbounded re-application.

```
TXL0931E TXL bootstrap - (TXL internal error) Too many symbols in TXL
      bootstrap
TXL0932E TXL bootstrap - (TXL internal error) Syntax error in TXL bootstrap
      - expected 'TOKEN' + got 'TOKEN'
TXL0933E TXL bootstrap - (TXL internal error) Syntax error in TXL bootstrap
      - expected '|', got 'TOKEN'
TXL0934E TXL bootstrap - (TXL internal error) Syntax error in TXL bootstrap
      - multiple tokens in choice alternative
TXL0935E TXL bootstrap - (TXL internal error) [TYPENAME] has not been defined
```

These errors indicate a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0941E - (TXL implementation limit) Too many -I include directories (> NNN)
```

The total number of include libraries specified using *-I* command line options is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Merge library directories or use symbolic links to reduce the number of include libraries. If the problem persists, report this problem as a bug.

```
TXL0942E - (TXL implementation limit) Too many preprocessor symbols (> NNN)
```

The total number of different preprocessor symbols used in the program is larger than the TXL implementation can handle.

Recommended action: This limit is not extensible. Reduce the dependency on preprocessor symbols by physically splitting source versions. If the problem persists, report this problem as a bug.

```
TXL0943E - Unrecognized debugging option '-DOPTION'
```

The indicated debugging switch is not implemented in this version of TXL.

Recommended action: Check the spelling of the option. TXL debugging options are always lower case identifiers following -D, for example '-Dparse'. If the desired option is not implemented, use other debugging options or *txdb* to investigate the problem.

```
TXL0944E - Can't find TXL program file 'PROGNAME.Txl'
```

The indicated TXL program file could not be found by TXL in the present working directory, the *Txl* subdirectory of the present working directory, any of the directories specified using *-I* command line options, or the system TXL library directory. Most often this is due to a misspelling of the TXL program or input file name.

Recommended action: Check the spelling of the given file names. If correct, then see the section "The 'txl' Command" of the TXL User's Guide to check you are using the command correctly.

`TXL0945E - Can't open TXL load file 'PROGNAME.CTxl'`

When using the *-load* option, the indicated TXL virtual machine byte code file (i.e., *.CTxl* file) could not be found by TXL in the present working directory, the *Txl* subdirectory of the present working directory, any of the directories specified using *-I* command line options, or the system TXL library directory. Most often this is due to a misspelling of the TXL program or input file name, or incorrect use of the *-load* option.

Recommended action: Check the spelling of the given file names. If correct, then see the section "The 'txl' Command" of the TXL User's Guide to check that you are using the command correctly.

`TXL0946E - Unrecognized command line option in #pragma`

An option flag specified using '-OPTION' in a *#pragma* statement in the TXL program is not a valid TXL command line option.

Recommended action: Check that the option flags specified in the *#pragma* statement are valid TXL command line options.

```
TXL0951E - [debug] predefined function not implemented in standalone
      applications
```

A standalone TXL application compiled using *txlc* uses the [debug] built-in function. For efficiency reasons, this function is not implemented in standalone applications.

Recommended action: Use [print], [message], [put] or [putp] to output the required information, or run the program interpretively using the *txl* or *txldb* command.

`TXL0952W - (Warning) Forced to split [SPOFF] output at line boundary`

The range of an [SPOFF] formatting directive yielded an output sequence longer than the specified maximum output line length. The output was split as necessary to stay within the maximum output width.

Recommended action: Check that every [SPOFF] is matched by an [SPON]. If possible, split [SPOFF] ranges into a finer grain. Specify a longer output line length using *-w*.

`TXL0953W - (Warning) Output token too long for output width`

A single output token was longer than the specified maximum output line length. The token was output on a line by itself.

Recommended action: Specify a longer output line length using *-w*.

```
TXL0959E - (TXL implementation limit) Output recursion limit exceeded
      (probable cause: small size or stack limit)
```

The available stack space was exhausted while unparsing the result of the transformation. Normally this means that the allocated stack space was too small.

Recommended action: Increase TXL limits using the *-size* option. If the problem persists, report this problem as a TXL bug.

```
TXL0960E - (TXL implementation limit) Result of [quote] predefined function
      exceeds maximum line length (1048575 characters)
```

The output text of a TXL variable quoted by the [quote] built-in function was longer than the TXL implementation limit of `1048575` characters.

Recommended action: Write the text to a file using [write], and read it back using [read] as required.

```
TXL0971E - (TXL internal error) Fatal TXL error in copyTree
```

This error indicates a catastrophic failure in the TXL compiler/interpreter.

Recommended action: Save a copy of the entire TXL program, its include files, and the input (if any) used for the failing run. Report the problem as a TXL bug, including the copy with the bug report.

```
TXL0981E - Out of tree space - NNN trees have been allocated
```

The total number of trees needed to implement the transformation is larger than the TXL implementation can handle at the present size. Normally this indicates that the TXL size is too small for the transformation, but it is also a possible indication of an infinite transform (i.e., a program that never terminates).

Recommended action: Increase TXL implementation limits using the *-size* option. If an infinite transform is suspected, double check that the rules of the program will terminate.

```
TXL0984E - Out of kid space - NNN kids have been allocated
```

The total number of kids (subtree cells) needed to implement the transformation is larger than the TXL implementation can handle at the present size. Normally this indicates that the TXL size is too small for the transformation, but it is also a possible indication of an infinite transform (i.e., a program that never terminates).

Recommended action: Increase TXL implementation limits using the *-size* option. If an infinite transform is suspected, double check that the rules of the program will terminate.

```
TXL0991E - Unable to open output file 'FILENAME'
```

The output file specified in the *-o* command line option could not be created by TXL. Most often this is because you do not have write permission in the present working directory or the directory where the file is to be created.

Recommended action: Check that the program is being run in the intended directory and that the permissions on the directory where the file is to be created allow writing.