z/OS
2.5

XL C/C++
Messages

IBM

> **Note**
>
> Before using this information and the product it supports, read the information in "Notices" on page 347.

Contents

About this document

This edition of *z/OS XL C/C++ Messages* is intended for users of the IBM® z/OS XL C/C++ compiler with the IBM Language Environment® element provided with z/OS. It provides you with information on the compiler return codes, compiler messages, utility messages, and C/C++ legacy class libraries messages.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line (|) to the left of the change.

You may notice changes in the style and structure of some of the contents in this document; for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our documents.

z/OS XL C/C++ and related documents

This topic summarizes the content of the z/OS XL C/C++ documents and shows where to find related information in other documents.

Table 1. z/OS XL C/C++ and related documents

Document Title and Number	Key Sections/Chapters in the Document
z/OS XL C/C++ Programming Guide	Guidance information for: • XL C/C++ input and output • Debugging z/OS XL C programs that use input/output • Using linkage specifications in C++ • Combining C and assembler • Creating and using DLLs • Using threads in z/OS UNIX System Services applications • Reentrancy • Handling exceptions, error conditions, and signals • Performance optimization • Network communications under z/OS UNIX • Interprocess communications using z/OS UNIX • Structuring a program that uses C++ templates • Using environment variables • Using System Programming C facilities • Library functions for the System Programming C facilities • Using runtime user exits • Using the z/OS XL C multitasking facility • Using other IBM products with z/OS XL C/C++ (IBM CICS® Transaction Server for z/OS, CSP, DWS, IBM DB2®, IBM GDDM, IBM IMS, ISPF, IBM QMF) • Globalization: locales and character sets, code set conversion utilities, mapping variant characters • POSIX character set • Code point mappings • Locales supplied with z/OS XL C/C++ • Charmap files supplied with z/OS XL C/C++ • Examples of charmap and locale definition source files • Converting code from coded character set IBM-1047 • Using built-in functions • Using vector programming support • Using runtime check library • Using high performance libraries • Programming considerations for z/OS UNIX C/C++

Table 1. z/OS XL C/C++ and related documents (continued)

Document Title and Number	Key Sections/Chapters in the Document
z/OS XL C/C++ User's Guide	Guidance information for: • z/OS XL C/C++ examples • Compiler options • Binder options and control statements • Specifying Language Environment runtime options • Compiling, IPA Linking, binding, and running z/OS XL C/C++ programs • Utilities (Object Library, CXXFILT, DSECT Conversion, Code Set and Locale, ar and make, BPXBATCH, c89, xlc) • Diagnosing problems • Cataloged procedures and IBM REXX EXECs • Customizing default options for the z/OS XL C/C++ compiler
z/OS XL C/C++ Language Reference	Reference information for: • The C and C++ languages • Lexical elements of z/OS XL C and C++ • Declarations, expressions, and operators • Implicit type conversions • Functions and statements • Preprocessor directives • C++ classes, class members, and friends • C++ overloading, special member functions, and inheritance • C++ templates and exception handling • z/OS XL C and C++ compatibility
z/OS XL C/C++ Messages	Provides error messages and return codes for the compiler, and its related application interface libraries and utilities. For the XL C/C++ runtime library messages, refer to *z/OS Language Environment Runtime Messages*. For the c89 and xlc utility messages, refer to *z/OS UNIX System Services Messages and Codes*.
z/OS XL C/C++ Runtime Library Reference	Reference information for: • header files • library functions

Table 1. z/OS XL C/C++ and related documents (continued)

Document Title and Number	Key Sections/Chapters in the Document
z/OS C Curses	Reference information for: • Curses concepts • Key data types • General rules for characters, renditions, and window properties • General rules of operations and operating modes • Use of macros • Restrictions on block-mode terminals • Curses functional interface • Contents of headers • The terminfo database
z/OS XL C/C++ Compiler and Runtime Migration Guide for the Application Programmer	Guidance and reference information for: • Common migration questions • Application executable program compatibility • Source program compatibility • Input and output operations compatibility • Class library migration considerations • Changes between releases of z/OS • Pre-z/OS C and C++ compilers to current compiler migration • Other migration considerations
z/OS Metal C Programming Guide and Reference	Guidance and reference information for: • Metal C run time • Metal C programming • AR mode
Standard C++ Library Reference	The documentation describes how to use the following three main components of the Standard C++ Library to write portable C/C++ code that complies with the ISO standards: • ISO Standard C Library • ISO Standard C++ Library • Standard Template Library (C++) The ISO Standard C++ library consists of 51 required headers. These 51 C++ library headers (along with the additional 18 Standard C headers) constitute a hosted implementation of the C++ library. Of these 51 headers, 13 constitute the Standard Template Library, or STL.

Table 1. z/OS XL C/C++ and related documents (continued)

Document Title and Number	Key Sections/Chapters in the Document
z/OS Common Debug Architecture User's Guide	This documentation is the user's guide for IBM's `libddpi` library. It includes: • Overview of the architecture • Information on the order and purpose of API calls for model user applications and for accessing DWARF information • Information on using the Common Debug Architecture with C/C++ source This user's guide is part of the Runtime Library Extensions documentation.
z/OS Common Debug Architecture Library Reference	This documentation is the reference for IBM's `libddpi` library. It includes: • General discussion of Common Debug Architecture • Description of APIs and data types related to stacks, processes, operating systems, machine state, storage, and formatting This reference is part of the Runtime Library Extensions documentation.
DWARF/ELF Extensions Library Reference	This documentation is the reference for IBM's extensions to the `libdwarf` and `libelf` libraries. It includes information on: • Consumer APIs • Producer APIs This reference is part of the Runtime Library Extensions documentation.
IBM Developer for z Systems®	The documentation for IBM Developer for z Systems (www.ibm.com/docs/en/adfz/developer-for-zos) provides guidance and reference information for debugging programs, using IBM Developer for z Systems in different environments, and language-specific information.

Note: For complete and detailed information on linking and running with Language Environment services and using the Language Environment runtime options, refer to *z/OS Language Environment Programming Guide*. For complete and detailed information on using interlanguage calls, refer to *z/OS Language Environment Writing Interlanguage Communication Applications*.

The following table lists the z/OS XL C/C++ and related documents. The table groups the documents according to the tasks they describe.

Table 2. Documents by task

Tasks	Documents
Planning, preparing, and migrating to z/OS XL C/C++	• *z/OS XL C/C++ Compiler and Runtime Migration Guide for the Application Programmer* • *z/OS Language Environment Customization* • *z/OS Language Environment Runtime Application Migration Guide* • *z/OS UNIX System Services Planning* • *z/OS Planning for Installation*

Table 2. Documents by task (continued)

Tasks	Documents
Installing	• *z/OS Program Directory* • *z/OS Planning for Installation* • *z/OS Language Environment Customization*
Option customization	• *z/OS XL C/C++ User's Guide*
Coding programs	• *z/OS XL C/C++ Runtime Library Reference* • *z/OS XL C/C++ Language Reference* • *z/OS XL C/C++ Programming Guide* • *z/OS Metal C Programming Guide and Reference* • *z/OS Language Environment Concepts Guide* • *z/OS Language Environment Programming Guide* • *z/OS Language Environment Programming Reference*
Coding and binding programs with interlanguage calls	• *z/OS XL C/C++ Programming Guide* • *z/OS XL C/C++ Language Reference* • *z/OS Language Environment Programming Guide* • *z/OS Language Environment Writing Interlanguage Communication Applications* • *z/OS MVS Program Management: User's Guide and Reference* • *z/OS MVS Program Management: Advanced Facilities*
Compiling, binding, and running programs	• *z/OS XL C/C++ User's Guide* • *z/OS Language Environment Programming Guide* • *z/OS Language Environment Debugging Guide* • *z/OS MVS Program Management: User's Guide and Reference* • *z/OS MVS Program Management: Advanced Facilities*
Compiling and binding applications in the z/OS UNIX (z/OS UNIX) environment	• *z/OS XL C/C++ User's Guide* • *z/OS UNIX System Services User's Guide* • *z/OS UNIX System Services Command Reference* • *z/OS MVS Program Management: User's Guide and Reference* • *z/OS MVS Program Management: Advanced Facilities*

Table 2. Documents by task (continued)

Tasks	Documents
Debugging programs	• README file • *z/OS XL C/C++ User's Guide* • *z/OS XL C/C++ Messages* • *z/OS XL C/C++ Programming Guide* • *z/OS Language Environment Programming Guide* • *z/OS Language Environment Debugging Guide* • *z/OS Language Environment Runtime Messages* • *z/OS UNIX System Services Messages and Codes* • *z/OS UNIX System Services User's Guide* • *z/OS UNIX System Services Command Reference* • *z/OS UNIX System Services Programming Tools* • IBM Developer for z Systems (www.ibm.com/docs/en/adfz/developer-for-zos) documentation
Developing debuggers and profilers	• *z/OS Common Debug Architecture User's Guide* • *z/OS Common Debug Architecture Library Reference* • *DWARF/ELF Extensions Library Reference*
Packaging XL C/C++ applications	• *z/OS XL C/C++ Programming Guide* • *z/OS XL C/C++ User's Guide*
Using shells and utilities in the z/OS UNIX environment	• *z/OS XL C/C++ User's Guide* • *z/OS UNIX System Services Command Reference* • *z/OS UNIX System Services Messages and Codes*
Using sockets library functions in the z/OS UNIX environment	• *z/OS XL C/C++ Runtime Library Reference*
Using the ISO Standard C++ Library to write portable C/C++ code that complies with ISO standards	• *Standard C++ Library Reference*
Performing diagnosis and submitting an Authorized Program Analysis Report (APAR)	• *z/OS XL C/C++ User's Guide*

Note: For information on using the prelinker, see the appendix on prelinking and linking z/OS XL C/C++ programs in *z/OS XL C/C++ User's Guide*.

Softcopy documents

The z/OS XL C/C++ publications are supplied in PDF format and available for download from the z/OS XL C/C++ documentation library (www.ibm.com/software/awdtools/czos/library).

Note: To ensure that you can access cross-reference links to other z/OS XL C/C++ PDF documents, download each document into the same directory on your local machine and do not change the PDF file names.

To read a PDF file, use the Adobe Reader. If you do not have the Adobe Reader, you can download it (subject to Adobe license terms) from the Adobe website (www.adobe.com).

You can also browse the documents on the World Wide Web by visiting the z/OS Internet Library (www.ibm.com/servers/resourcelink/svc00100.nsf/pages/zosInternetLibrary).

Softcopy examples

Most of the larger examples in the following documents are available in machine-readable form:

- *z/OS XL C/C++ Language Reference*
- *z/OS XL C/C++ User's Guide*
- *z/OS XL C/C++ Programming Guide*

In the following documents, a label on an example indicates that the example is distributed as a softcopy file:

- *z/OS XL C/C++ Language Reference*
- *z/OS XL C/C++ Programming Guide*
- *z/OS XL C/C++ User's Guide*

The label is the name of a member in the CBC.SCCNSAM data set. The labels begin with the form CCN or CLB. Examples labelled as CLB appear only in the *z/OS XL C/C++ User's Guide*, while examples labelled as CCN appear in all three documents, and are further distinguished by *x* following CCN, where *x* represents one of the following:

- R and X refer to *z/OS XL C/C++ Language Reference*
- G refers to *z/OS XL C/C++ Programming Guide*
- U refers to *z/OS XL C/C++ User's Guide*

z/OS XL C/C++ on the World Wide Web

Additional information on z/OS XL C/C++ is available on the product page for z/OS XL C/C++ (www.ibm.com/products/xl-cpp-compiler-zos).

This page contains late-breaking information about the z/OS XL C/C++ product, including the compiler, the C/C++ libraries, and utilities. There are links to other useful information, such as the z/OS XL C/C++ information library and the libraries of other z/OS elements that are available on the web. The z/OS XL C/C++ home page also contains links to other related websites.

Where to find more information

For an overview of the information associated with z/OS, see *z/OS Information Roadmap*.

z/OS Basic Skills in IBM Documentation

z/OS Basic Skills in IBM Documentation is a Web-based information resource intended to help users learn the basic concepts of z/OS, the operating system that runs most of the IBM mainframe computers in use today. IBM Documentation is designed to introduce a new generation of Information Technology professionals to basic concepts and help them prepare for a career as a z/OS professional, such as a z/OS system programmer.

Specifically, z/OS Basic Skills is intended to achieve the following objectives:

- Provide basic education and information about z/OS without charge
- Shorten the time it takes for people to become productive on the mainframe
- Make it easier for new people to learn z/OS.

z/OS Basic Skills in IBM Documentation (www.ibm.com/docs/en/zos-basic-skills?topic=zosbasics/com.ibm.zos.zbasics/homepage.html) is available to all users (no login required).

Technical support

Additional technical support is available from the z/OS XL C/C++ Support page (www.ibm.com/mysupport/s/topic/0TO0z0000006v6TGAQ/xl-cc?language=en_US&productId=01t0z000007g72LAAQ). This page provides a portal with search capabilities to a large selection of technical support FAQs and other support documents.

If you cannot find what you need, you can e-mail:

compinfo@cn.ibm.com

For the latest information about z/OS XL C/C++, visit the product page for z/OS XL C/C++ (www.ibm.com/products/xl-cpp-compiler-zos).

For information about boosting performance, productivity and portability, visit IBM Z and LinuxONE Community (community.ibm.com/community/user/ibmz-and-linuxone/groups/topic-home?CommunityKey=5805da79-8284-4015-97fb-5a19f6480452).

How to send your comments to IBM

We invite you to submit comments about the z/OS product documentation. Your valuable feedback helps to ensure accurate and high-quality information.

Important: If your comment regards a technical question or problem, see instead "If you have a technical problem" on page xiii.

Submit your feedback by using the appropriate method for your type of comment or question:

Feedback on z/OS function
If your comment or question is about z/OS itself, submit a request through the IBM RFE Community (www.ibm.com/developerworks/rfe/).

Feedback on IBM Documentation function
If your comment or question is about the IBM Documentation functionality, for example search capabilities or how to arrange the browser view, send a detailed email to IBM Documentation Support at ibmdocs@us.ibm.com.

Feedback on the z/OS product documentation and content
If your comment is about the information that is provided in the z/OS product documentation library, send a detailed email to mhvrcfs@us.ibm.com. We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information.

To help us better process your submission, include the following information:

- Your name, company/university/institution name, and email address
- The following deliverable title and order number: z/OS XL C/C++ Messages, GC14-7305-50
- The section title of the specific information to which your comment relates
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive authority to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

If you have a technical problem or question, do not use the feedback methods that are provided for sending documentation comments. Instead, take one or more of the following actions:

- Go to the IBM Support Portal (support.ibm.com).

- Contact your IBM service representative.
- Call IBM technical support.

Summary of changes

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line to the left of the change.

Note: IBM z/OS policy for the integration of service information into the z/OS product documentation library is documented on the z/OS Internet Library under IBM z/OS Product Documentation Update Policy (www-01.ibm.com/servers/resourcelink/svc00100.nsf/pages/ibm-zos-doc-update-policy?OpenDocument).

Summary of changes for z/OS XL C/C++ Messages for Version 2 Release 5 (V2R5)

This information contains no technical changes for this release.

Summary of changes for z/OS XL C/C++ Messages for Version 2 Release 4 (V2R4)

The following lists indicate the messages that are new, changed, or no longer issued in z/OS V2R4 and its updates. Messages that have been added, updated, or that are no longer issued in an updated edition of V2R4 are identified by the quarter and year that the message was updated, in parentheses. For example, (4Q2019) indicates that a message was updated in the fourth quarter of 2019.

New

The following message is new.

　CCN3404

Changed

The following message is changed.

　CCN8160

Deleted

The following messages were deleted.

　None

Summary of changes for z/OS XL C/C++ Messages for Version 2 Release 3 (V2R3)

The following lists indicate the messages that are new, changed, or no longer issued in z/OS V2R3 and its updates. Messages that have been added, updated, or that are no longer issued in an updated edition of V2R3 are identified by the quarter and year that the message was updated, in parentheses. For example, (4Q2017) indicates that a message was updated in the fourth quarter of 2017.

New

The following message is new.

　CCN1153

CDA1022
CDA3506

Changed

The following message is changed.

None

Deleted

The following messages were deleted.

None

Chapter 1. About IBM z/OS XL C/C++

For introductory information concerning z/OS XL C/C++ compiler, including information on the changes that have been made for the current release, see About IBM z/OS XL C/C++ in *z/OS XL C/C++ User's Guide*.

Chapter 2. z/OS XL C/C++ compiler return codes and messages

This topic contains information about the compiler messages and should not be used as programming interface information.

Return codes

For every compilation job or job step, the compiler generates a return code that indicates to the operating system the degree of success or failure it achieved:

Table 3. Return codes from compilation of a z/OS XL C/C++ program

Return Code	Type of Error Detected	Compilation Result
0	No error detected; informational messages may have been issued.	Compilation completed. Successful execution anticipated.
4	Warning error detected.	Compilation completed. Execution may not be successful.
8	Error detected.	Compilation may have been completed. Successful execution not possible.
12	Severe error detected.	Compilation may have been completed. Successful execution not possible.
16	Terminating error detected.	Compilation terminated abnormally. Successful execution not possible.
33	A library level prior to the z/OS V2R5 Language Environment library level was used.	Compilation terminated abnormally. Successful execution not possible.

The return code indicates the highest possible error severity that the compiler detected. Therefore, a particular entry in the "Type of Error Detected" column, includes *all* error types above it. For example, return code 12 indicates that the compiler has issued a severe error and may have also issued any combination of error, warning, and informational messages. But it does not necessarily mean that all these error types are present in that particular compile.

Compiler messages

Message format: CCNnnnn text <&*n*> or **CCNnnnn text <&*n$s*>** where:

nnnn
　　error message number

text
　　message which appears on the screen

&*n* or &*n$s*
　　compiler substitution variable

3

CCN0000 **&1(&2) Phase(&3) Level(&4)**

Explanation

This message indicates the version of the compiler component being invoked, or the version of the library that the component is using. The information can be used to verify the service level of the compiler and help your service representative determine if you have the latest maintenance.

In the message text:

&1 is the string "Product" or "Library", &2 is the product number or a library name, &3 is the compiler phase name and &4 is the version of the compiler phase or the library.

User response

In case there is a compiler related problem you should provide this information to your service representative.

CCN0001 **Incompatible compiler components detected: &1 is the version from &2. &3 is the &4 with which &5 was compiled.**

Explanation

One or more compiler components within the compiler data set are out of date.

In the message text:

&1 is a hexadecimal value representing the version of '&2'. &2 is the main compiler component name. &3 is a hexadecimal value representing the version that '&5' expects. &4 is the version of the internal control block. &5 is the name of a compiler component that contains an incompatible internal control block definition.

User response

Ensure that the compiler PTFs have been applied without errors. If the problem persists, contact the IBM service representative responsible for your installation.

CCN0002 **Incompatible options class detected: &1 is the offset of &2 from the &3 component. &4 is the offset of &5 from the &6 component.**

Explanation

One or more compiler components within the compiler data set are out of date.

In the message text:

&1 is a decimal value representing the offset of '&2'. &2 is the class member whose offset is shown. &3 is the main compiler component name. &4 is a decimal value representing the offset of '&5'. &5 is the class member whose offset is shown. &6 is the name of a compiler component that contains an incompatible internal control block definition.

User response

Ensure that the compiler PTFs have been applied without errors. If the problem persists, contact the IBM service representative responsible for your installation.

CCN0007 **&1 Elapsed Time: &2s, CPU: &3s**

Explanation

This message reports the approximate amount of time in seconds the indicated compiler phase took to complete.

In the message text:

&1 is the compiler phase name, &2 the elapsed time of the phase in seconds, and &3 the CPU time consumed by the phase in seconds.

User response

In case there is a compilation time related problem you should provide this information to your service representative.

CCN0008 **Source file &1 cannot be opened.**

Explanation

The compiler could not open the specified source file.

In the message text:

&1 is a file name, enclosed in quotes or angle brackets as specified in the corresponding "include" directive.

User response

Ensure the source file name is correct. Ensure that the correct file is being read and has not been corrupted. If the file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN0015   The compiler could not open the output file "&1".

Explanation

The compiler could not open the specified output file.

In the message text:

&1 is a file name.

User response

Ensure the output file name is correct. Also, ensure that the location of the output file has sufficient storage available. If using a LAN drive, ensure that the LAN is working properly and you have permission to write to the disk.

CCN0049   The option "&1" is not supported.

Explanation

The command line contained an option that is not supported. Note that some option parameters must not have spaces between the option and the parameter.

In the message text:

&1 is an option.

User response

Remove the option. Check the syntax of the options.

CCN0358   The "&1" option is not allowed with the "&2" option.

Explanation

The specified options cannot be used together. The first option specified in the message is ignored.

In the message text:

&1 and &2 are both option names.

User response

Remove one of the options.

CCN0458   Option &1 is invalid because option &2 is not specified.

Explanation

The option &1 is only valid when used in conjunction with &2.

In the message text:

&1 and &2 are both option names.

User response

Compile with &2 or remove &1.

CCN0459   An incomplete compile option for "&1" has been specified. "&2" was expected.

Explanation

The command line contained an incomplete option. The message identifies what the compiler expected and what it actually found.

In the message text:

&1 is the option name. &2 is the token that was missing.

User response

Complete the compile option.

CCN0460   Negative form of option "&1" is not allowed.

Explanation

Specified option is not allowed in negative form.

In the message text:

&1 is the option name.

User response

Remove the option or change it to the positive form.

CCN0461   "&1" is not a valid suboption for "&2". Option is ignored.

Explanation

The command line contained an option with an invalid suboption.

In the message text:

&1 is the option name.

User response

Remove the suboption.

CCN0462   "&1" must have a suboption specified.

Explanation

The command line contained an option that was missing a suboption.

In the message text:

&1 is the option name.

User response

Specify a suboption.

CCN0463 **Suboption is not allowed in "&1" option.**

Explanation

Suboption is not allowed in the specified option.

In the message text:

&1 is the option name.

User response

Remove the suboption.

CCN0464 **"&1" requires exactly "&2" sub-option(s) to be specified. "&3" were given.**

Explanation

The command line contained an option that had an incorrect number of sub-options specified. The sub-option(s) are ignored.

In the message text:

&1 is the option name. &2 is the number of sub-options expected. &3 is the number of sub-options given.

User response

Ensure that the correct number of sub-option(s) is given.

CCN0465 **"&1" requires at most "&2" sub-option(s) to be specified. "&3" were given.**

Explanation

The command line contained an option that had more sub-option(s) specified than are allowed. The option is ignored.

In the message text:

&1 is the option name. &2 is the maximum number of suboption(s) allowed. &3 is the number of sub-options given.

User response

Ensure that the maximum number of sub-options is not exceeded.

CCN0466 **"&1" requires at least "&2" sub-option(s) to be specified. "&3" were given.**

Explanation

The command line contained an option that had fewer sub-option(s) specified than are required. The option is ignored.

In the message text:

&1 is the option name. &2 is the minimum number of sub-option(s) required. &3 is the number of sub-options given.

User response

Ensure that the minimum number of sub-options is specified.

CCN0569 **Option "&1" is not supported for &2.**

Explanation

The option is not supported by this compiler.

In the message text:

&1 is the option name. &2 is either C or C++.

User response

Remove the option.

CCN0611 **Unable to access options file &1.**

Explanation

The compiler could not access the specified options file. It was either unable to open it or unable to read it.

In the message text:

&1 is the options file name specified on OPTFILE option.

User response

Ensure the options file name and other specifications are correct. Ensure that the access authority is sufficient. Ensure that the file being accessed has not been corrupted.

CCN0612 **Option &1 specified in an options file is ignored.**

Explanation

Option &1 is not allowed in an options file.

In the message text:

&1 is an option name specified in the options file.

User response

Remove the &1 option from the options file. Option OPTFILE can not be nested.

CCN0613	The continuation character on the last line of the options file &1 is ignored.

Explanation

The continuation character on the last line of a file is unnecessary.

In the message text:

&1 is the options file name.

User response

Remove the continuation character on the last line of the options file. Make sure that it is not a typo for something else.

CCN0614	Macro name "&1" contains characters not valid on the "&2" option.

Explanation

Macro names can contain only alphanumeric characters and the underscore character and must not begin with a numeric character.

In the message text:

&1 is the invalid macro name and &2 is the option name.

User response

Change the macro name.

CCN0615	Semantic function for processing "&1" option is missing.

Explanation

Option &1 cannot be processed because its semantic function is missing.

In the message text:

&1 is the option name.

User response

Provide the option semantic function.

CCN0616	Options file "&1" is already specified. All subsequent occurrences are ignored.

Explanation

An options file can only be specified once.

In the message text:

&1 is the options file name.

User response

Ensure that the options file is specified only once.

CCN0623	Option "&1" ignored because option "&2" specified.

Explanation

Specifying the second option indicated means the first has no effect.

In the message text:

&1 and &2 are both option names.

User response

Remove one of the options.

CCN0624	&1 is not a valid data set name.

Explanation

The data set name is not valid because it is too long.

In the message text:

&1 is a data set name.

User response

Use a shorter data set name.

CCN0625	&1 does not exist.

Explanation

The data set does not exist.

In the message text:

&1 is a data set name.

User response

Supply an existing data set.

CCN0626	There are no members in &1 to compile.

Explanation

There are no members in the partitioned data set to compile.

In the message text:

&1 is a data set name.

User response

Supply a partitioned data set that contains members.

CCN0627	&1 should be a partitioned data set.

Explanation

A partitioned data set is expected.

In the message text:

&1 is a data set name.

User response

Supply a partitioned data set.

CCN0628	&1 should not be a partitioned data set.

Explanation

A non-partitioned data set is expected.

In the message text:

&1 is a data set name.

User response

Supply a non-partitioned data set.

CCN0629	&1 has invalid attributes.

Explanation

The attributes of the data set do not match the attributes expected by the compiler.

In the message text:

&1 is a data set name.

User response

Check the informational messages issued with this message and change the data set attributes accordingly.

CCN0630	&1 has attributes &2.

Explanation

The data set has the attributes indicated.

In the message text:

&1 is a data set name, &2 is a set of data set attributes.

User response

No user response is required.

CCN0631	The attributes should be &1.

Explanation

The data set should have the attributes indicated.

In the message text:

&1 is a set of data set attributes.

User response

No user response is required.

CCN0632	The attributes should be one of the following:

Explanation

The data set should have one of the sets of attributes indicated.

User response

No user response is required.

CCN0633	Unable to allocate &1.

Explanation

Unable to allocate the data set.

In the message text:

&1 is a data set name.

User response

Check that the data set has a valid name and can be accessed.

CCN0634	Unable to load &1. Compilation terminated.

Explanation

Unable to fetch one of the compiler phases.

In the message text:

&1 is the name of a program module.

User response

Check that the compiler is installed correctly. Make sure there is enough memory in the region to fetch the module. You may need to specify the runtime option HEAP(,,,FREE,,) to prevent the compiler from running out of memory.

CCN0635   **Timestamp error on &1.**

Explanation

Timestamp error while compiling a partitioned data set.

In the message text:

&1 is a data set name.

User response

Check to see if the data set is corrupted.

CCN0636   **The file allocated to &1 cannot be opened, because it is already opened by another process.**

Explanation

The file allocated to the DD name was opened for output by another process.

In the message text:

&1 is a DD name.

User response

Ensure that the file is not shared for output.

CCN0637   **Unable to locate CICS translator.**

Explanation

The CICS option was enabled but the compiler could not load the CICS translator.

User response

Ensure that CICS is correctly installed on the system and is included during the compile.

CCN0702   **An error was encountered in accessing the alternate ddname table. The default ddnames will be used.**

Explanation

The compiler could not access the alternate ddname table. Compilation will continue, using the default ddname table.

User response

Check that the alternate ddname table was coded correctly.

CCN0703   **An error was encountered in a call to &1 while processing &2.**

Explanation

A library function called by the compiler encountered an error. The compiler will issue a perror() message with more specific information on the failure.

In the message text:

&1 is the name of the library function. &2 is the name of the file or path.

User response

If the file was created by the user, verify that it was created correctly; See the programmer response for the accompanying perror() message for additional information.

CCN0704   **There are no files with the default extension in &1.**

Explanation

There are no files in the given directory which match the default extension. The compiler returned without compiling any files.

In the message text:

&1 is a directory name.

User response

Supply a directory which contains files with the appropriate extension. The default extension for C is ".c" and the default extension for C++ is ".C".

CCN0705   **The output file &1 is not supported in combination with source file &2.**

Explanation

The output file specified in a compiler option is of a type which is not supported in combination with the type of the source file. An informational message describing supported output file types for the given source file type follows.

In the message text:

&1 is an output file specified in a compiler option, and &2 is the source file to be compiled.

User response

Supply an output file of one of the supported types in the compiler sub-option, or let the compiler generate a default output file name.

CCN0706	**The source file is a CMS file. The suboption should specify a CMS file or a BFS file in an existing directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0707	**The source file is a BFS file. The suboption should specify a CMS file, a BFS file in an existing directory, or an existing BFS directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0708	**The source file is a BFS directory. The suboption should specify an existing BFS directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0709	**The source file is a Sequential data set. The suboption should specify a sequential data set, a PDS member, or a UNIX file in an existing directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0710	**The source file is a PDS member. The suboption should specify a sequential data set, a PDS member, a PDS, a UNIX file in an existing directory, or an existing UNIX directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0711	**The source file is a PDS. The suboption should specify a PDS or an existing UNIX directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0712	**The source file is a UNIX file. The suboption should specify a sequential data set, a PDS member, a UNIX file in an existing directory, or an existing UNIX directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0713   **The source file is a UNIX directory. The suboption should specify an existing UNIX directory.**

Explanation

The output file specified in the suboption has a type that is incompatible with the source file type.

User response

Specify an output file with a compatible type as indicated in the message.

CCN0721   **Option "&1" cannot be specified with option "&2". Option "&3" is ignored.**

Explanation

A SEARCH or LSEARCH option cannot be specified on the same compiler invocation with a SYSPATH or USERPATH option. All previous specifications of the conflicting options are ignored.

In the message text:

&1 option name, &2 option name, &3 option name.

User response

Use the correct syntax for specifying the option

CCN0745   **&1 should be a partitioned data set or UNIX directory.**

Explanation

A partitioned data set or UNIX directory is expected.

In the message text:

&1 is a data set name.

User response

Supply a partitioned data set or UNIX directory.

CCN0750   **Suboptions "&1" and "&2" of option "&3" conflict.**

Explanation

Sub-options of the specified option are in conflict with each other.

In the message text:

&3 is the option name. &1 and &2 are the sub-option names.

User response

Change the sub-option.

CCN0756   **The data definition name "&1" cannot be resolved. File "&2" could not be opened:**

Explanation

The compiler tried to open the indicated file and associate it with the indicated data definition name. However, the file could not be opened. This is usually because the file does not exist, or you do not have permission to use the file.

In the message text:

&1 is the data definition name. &2 is the file name.

User response

Specify an existing file name to which you have permission.

CCN0757   **The data definition name "&1" cannot be resolved. Information for character special file "&2", needed to allocate file "&3", cannot be obtained:**

Explanation

fdN character special files are used for all path name allocations. The compiler tried to validate the indicated character special file by using the `stat` function, but that function failed. The compiler cannot use the indicated file without the indicated character special file.

In the message text:

&1 is the data definition name. &2 is the character special file name. &3 is the real path name.

System programmer response

Ensure that the **fdN** character special files were correctly created with the `mknod` command, and that there are enough of them.

User response

Retry the compile. If the problem persists, contact the IBM service representative responsible for your installation.

CCN0758   **The data definition name "&1" cannot be resolved. File "&2", needed to allocate file "&3", is not character special.**

Explanation

fdN character special files are used for all path name allocations. The compiler validated the indicated character special file, using the stat function, and determined that the indicated file is not a character special file. The compiler cannot use the indicated file without the indicated character special file.

In the message text:

&1 is the data definition name. &2 is the character special file name. &3 is the real path name.

System programmer response

Ensure that the **fdN** character special files were correctly created with the mknod command, and that there are enough of them.

User response

Retry the compile. If the problem persists, contact the IBM service representative responsible for your installation.

CCN0759	The data definition name "&1" cannot be resolved. Character special file "&2", needed to allocate file "&3", is not major 5.

Explanation

fdN character special files are used for all path name allocations. The compiler validated the indicated character special file, using the stat function, and determined that the indicated character special file does not have the correct major number. The compiler cannot use the indicated file without the indicated character special file.

In the message text:

&1 is the data definition name. &2 is the character special file name. &3 is the real path name.

System programmer response

Ensure that the **fdN** character special files were correctly created with the mknod command, and that there are enough of them.

User response

Retry the compile. If the problem persists, contact the IBM service representative responsible for your installation.

CCN0760	The data definition name "&1" cannot be resolved. Character special file "&2", needed to

allocate file "&3", is not minor "&4".

Explanation

fdN character special files are used for all path name allocations. The compiler validated the indicated character special file, using the stat function, and determined that the indicated character special file does not have the correct minor number. The compiler cannot use the indicated file without the indicated character special file.

In the message text:

&1 is the data definition name. &2 is the character special file name. &3 is the real path name. &4 is the minor number.

System programmer response

Ensure that the **fdN** character special files were correctly created with the mknod command, and that there are enough of them.

User response

Retry the compile. If the problem persists, contact the IBM service representative responsible for your installation.

CCN0764	Compiler cannot create temporary files.

Explanation

The intermediate code files could not be created. Please verify that the target file system exists, is writable and is not full.

User response

Ensure that the designated location for temporary objects exists, is writable and is not full.

CCN0767	The "&1" feature of z/OS is not enabled. Contact your system programmer.

Explanation

This feature of z/OS is not enabled at your installation.

In the message text:

&1 is the feature name.

User response

Your system programmer can contact IBM z/OS service to have this element enabled.

CCN0768   **Compiling "&1".**

Explanation

Informational message issued during PDS or UNIX directory compiles to indicate when the compiler has started compiling the next member.

In the message text:

&1 is the member name.

User response

No user action is required.

CCN0770   **The name &1 is invalid. Please correct and recompile.**

Explanation

The name shown is invalid.

In the message text:

&1 is the file name.

User response

Please correct the name and retry.

CCN0790   **Options "&1" and "&2" are in conflict.**

Explanation

The specified options cannot be used together.

In the message text:

&1 and &2 are both option names.

User response

Remove one of the options.

CCN0791   **Options "&1" and "&2" are not compatible.**

Explanation

The specified options cannot be used together.

In the message text:

&1 and &2 are both option names.

User response

Change option values.

CCN0793   **Compilation failed for file &1. Object file not created.**

Explanation

The compiler detected an error and terminated the compilation. Object file was not created.

In the message text:

&1 is a file name

User response

Correct the reported errors and recompile.

CCN0795   **Unable to open existing data set &1.**

Explanation

Although the data set exists, the compiler was unable to open and/or obtain file information about it.

In the message text:

&1 is a data set name.

User response

Check the informational messages issued with this message and correct the corresponding problems associated with the data set.

CCN0796   **This compiler requires a runtime environment __librel() value of &1.**

Explanation

The compiler cannot run with the current runtime environment because it needs the runtime release indicated.

In the message text:

&1 is the required runtime level in the __librel() format.

User response

Check the informational message issued with this message to determine your current runtime release. Make sure you are running with the runtime environment required.

CCN0797   **You are currently running with the runtime environment &1.**

Explanation

The message displays the current runtime level installed on your system.

In the message text:

&1 is the current runtime level in the __librel() format.

User response

No user response is required.

CCN0822	Option &1 is locked and cannot be changed.

Explanation

The option has been locked during system installation. The option settings cannot be changed.

In the message text:

&1 is an option name.

User response

Remove the option from the command line, or ask the system programmer to unlock the option.

CCN0823	Lock suboption &1 is not supported.

Explanation

The lock suboption specified is not supported and is ignored.

In the message text:

&1 is an option name.

User response

The suboption to the lock option must itself be a valid option. The lock option is set during compiler installation. Check with the system programmer.

CCN0832	Option "&1" was set because option "&2" required it.

Explanation

The compiler requires both options to be specified.

In the message text:

&1 and &2 are both option names.

User response

Ensure that both options are set.

CCN0833	"&1" is not compatible with "&2". "&3" is being set.

Explanation

The option is not compatible with another option so it is ignored.

In the message text:

&1, &2 and &3 are all option names.

User response

Remove one of the options.

CCN0834	The blocksize "&1" specified in "&2" exceeds maximum valid size of "&3".

Explanation

The blocksize specified is too large and will be ignored.

In the message text:

&1 and &3 are integers, &2 the name of the parameter controlling the SPACE allocation.

User response

Set the blocksize to a smaller, valid value.

CCN0835	TARGET suboption "&1" is discontinued. Unexpected behavior might occur if an out-of-support TARGET level is specified. Use a TARGET level of "&2", or later instead.

Explanation

The specified target suboption is not supported.

In the message text:

&1 is the identifier of the unsupported target level. &2 is the identifier of the earliest supported target level.

User response

Specify a target level of "&2", or later.

CCN1001	INTERNAL COMPILER ERROR: &1.

Explanation

An internal compiler error occurred during compilation.

User response

Contact your Service Representative.
For more information, visit Internal
Compiler Error (www.ibm.com/support/docview.wss?uid=swg21110810).

CCN1002	Virtual storage exceeded.

Explanation

The compiler ran out of memory trying to compile the file. This sometimes happens with large files or programs with large functions. Note that very large programs limit the amount of optimization that can be done.

User response

Shut down any large processes that are running or increase your TSO region size. You can also divide the file into several small sections or shorten the function.

CCN1003   **&1.**

Explanation

General error message.

In the message text:

&1 is the detailed message text.

User response

There is no user response for this message.

CCN1031   **Unable to open file "&1".**

Explanation

The compiler could not open the specified file.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Ensure that the correct file is specified. If the file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN1032   **An error occurred while reading file "&1".**

Explanation

The compiler detected an error while reading from the specified file.

In the message text:

&1 is a file name.

User response

Ensure that the correct file is being read and has not been damaged. If the file is located on a LAN drive, ensure the LAN is working properly.

CCN1033   **An error occurred while writing to file "&1".**

Explanation

The compiler detected an error while writing to the specified file.

In the message text:

&1 is a file name.

User response

Ensure that the correct file is specified. If the file is located on a LAN drive, ensure the LAN is working properly.

CCN1034   **Read-only pointer initialization of dynamically allocated object &1 is not valid.**

Explanation

The value of a read-only pointer must be known at compile time; a pointer cannot be read-only and point to a dynamically allocated object at the same time because the address of the pointee is known at run time only.

User response

Modify the code so that the pointer is initialized with a read-only value or make the pointer read-write.

CCN1051   **Function &1 exceeds size limit.**

Explanation

The ACU for the function exceeds the LIMIT specified in the INLINE suboption.

User response

Increase LIMIT if feasible to do so.

CCN1052   **Function &1 is (or grows) too large to be inlined.**

Explanation

A function is too large to be inlined into another function.

User response

Use #pragma inline if feasible to do so.

CCN1053	Some calls to function &1 cannot be inlined.

Explanation

At least one call is either directly recursive, or the wrong number of parameters were specified.

User response

Check all calls to the function specified and make that number of parameters match the function definition.

CCN1054	Automatic storage for function &1 increased to over &2.

Explanation

The size of automatic storage for function increased by at least 4 KB due to inlining.

User response

Avoid inlining of functions which have large automatic storage.

CCN1055	Parameter area overflow while compiling &1. Parameter area size exceeds the allowable limit of &2.

Explanation

The parameter area for a function resides in the first 4K of automatic storage for that function. This message indicates that the parameter area cannot fit into 4K.

User response

Reduce the size of the parameter area by passing fewer parameters or by passing the address of a large structure rather than the structure itself.

CCN1057	&1 section size cannot exceed 16777215 bytes. Total section size is &2 bytes.

Explanation

A Data or Code section cannot exceed 16M in size.

User response

Partition input source files into multiple source files which can be compiled separately.

CCN1101	Maximum spill size of &2 is exceeded in function &1.

Explanation

Spill size is the size of the spill area. Spill area is the storage allocated if the number of machine registers is not sufficient for program translation.

User response

Reduce the complexity of the program and recompile.

CCN1102	Spill size for function &1 is not sufficient. Recompile specifying option SPILL(n) where &2 < n <= &3.

Explanation

Spill size is the size of the spill area. Spill area is the storage allocated if the number of machine registers is not sufficient for program translation.

User response

Recompile using the SPILL(n) option &2 < n <= &3 or with a different OPT level.

CCN1103	Internal error while compiling function &1. &2.

Explanation

An internal compiler error occurred during compilation.

User response

Contact your Service Representative or compile with a different OPT level. For more information, visit Internal Compiler Error (www.ibm.com/support/docview.wss?uid=swg21110810).

CCN1104	Internal error while compiling function &1. &2. Compilation terminated.

Explanation

An internal compiler error of high severity has occurred.

User response

Contact your Service Representative. Be prepared to quote the text of this message. For more information, visit Internal Compiler Error (www.ibm.com/support/docview.wss?uid=swg21110810).

CCN1105   **Constant table overflow compiling function &1. Compilation terminated.**

Explanation

The constant table is the table that stores all the integer and floating point constants.

User response

Reduce the number of constants in the program and recompile.

CCN1106   **Instruction in function &1 on line &2 is too complex. Compilation terminated.**

Explanation

The specified instruction is too complex to be optimized.

User response

Reduce the complexity of the instruction and recompile, or recompile with a different OPT level.

CCN1107   **Program too complex in function &1.**

Explanation

The specified function is too complex to be optimized.

User response

Reduce the complexity of the program and recompile, or recompile with a different OPT level.

CCN1108   **Expression too complex in function &1. Some optimizations not performed.**

Explanation

The specified expression is too complex to be optimized.

User response

Reduce the complexity of the expression or compile with a different OPT level.

CCN1109   **Infinite loop detected in function &1. Program may not stop.**

Explanation

An infinite loop has been detected in the given function.

User response

Recode the loop so that it will end.

CCN1110   **Loop too complex in function &1. Some optimizations not performed.**

Explanation

The specified loop is too complex to be optimized.

User response

No action is required.

CCN1111   **Division by zero detected in function &1. Runtime exception may occur.**

Explanation

A division by zero has been detected in the given function.

User response

Recode the expression to eliminate the divide by zero.

CCN1112   **Exponent is non-positive with zero as base in function &1. Runtime exception may occur.**

Explanation

This is a possible floating-point divide by zero.

User response

Recode the expression to eliminate the divide by zero.

CCN1113   **Unsigned division by zero detected in function &1. Runtime exception may occur.**

Explanation

A division by zero has been detected in the given function.

User response

Recode the expression to eliminate the divide by zero.

CCN1114   **Internal error while compiling function &1. &2.**

Explanation

An internal compiler error of low severity has occurred.

User response

Contact your Service Representative or compile with a different OPT level. For more information, visit Internal Compiler Error (www.ibm.com/support/docview.wss?uid=swg21110810).

CCN1115   **Control flow too complex in function &1; number of basic blocks or edges exceeds &2.**

Explanation

Basic blocks are segments of executable code without control flow. Edges are the possible paths of control flow between basic blocks.

User response

Reduce the complexity of the program and recompile.

CCN1116   **Too many expressions in function &1; number of symbolic registers exceeds &2.**

Explanation

Symbolic registers are the internal representation of the results of computations.

User response

Reduce the complexity of the program and recompile.

CCN1117   **Too many expressions in function &1; number of computation table entries exceeds &2.**

Explanation

The computation table contains all instructions generated in the translation of a program.

User response

Reduce the complexity of the program and recompile.

CCN1118   **Too many instructions in function &1; number of procedure list entries exceeds &2.**

Explanation

The procedure list is the list of all instructions generated by the translation of each subprogram.

User response

Reduce the complexity of the program and recompile.

CCN1119   **Number of labels in function &1 exceeds &2.**

Explanation

Labels are used whenever the execution path of the program could change; for example: if statements, switch statements, loops or conditional expressions.

User response

Reduce the complexity of the program and recompile.

CCN1120   **Too many symbols in function &1; number of dictionary entries exceeds &2.**

Explanation

Dictionary entries are used for variables, aggregate members, string literals, pointer dereferences, function names and internal compiler symbols.

User response

Compile the program at a lower level of optimization or simplify the program by reducing the number of variables or expressions.

CCN1121   **Program is too complex in function &1. Specify MAXMEM option value greater than &2.**

Explanation

Some optimizations not performed.

User response

Recompile specifying option MAXMEM with the suggested value for additional optimization.

CCN1122   **Parameter area overflow while compiling &1. Parameter area size exceeds &2.**

Explanation

The parameter area is used to pass parameters when calling functions. Its size depends on the number of

reference parameters, the number and size of value parameters, and on the linkage used.

User response

Reduce the size of the parameter area by passing fewer parameters or by passing the address of a large structure rather than the structure itself.

CCN1123	Spill size for function &1 is exceeded. Recompile specifying option SPILL(n) where &2 < n <= &3 for faster spill code.

Explanation

Spill size is the reserved size of the primary spill area. Spill area is the storage allocated if the number of machine registers is not sufficient for program translation.

User response

Recompile using the SPILL(n) option &2 < n <= &3 for improved spill code generation.

CCN1130	An error occurred while opening file "&1".

Explanation

The compiler could not open the specified file.

In the message text:

&1 is a file name

User response

Ensure the file name is correct. Ensure that the correct file is being opened and has not been damaged. If the file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN1131	An error occurred while writing file "&1".

Explanation

The compiler could not write to the specified file.

In the message text:

&1 is a file name

User response

Ensure the file name is correct. Ensure that the correct file is being written to and has not been damaged.

If the file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN1132	An error occurred while closing file "&1".

Explanation

The compiler could not write to the specified file.

In the message text:

&1 is a file name

User response

Ensure the file name is correct. Ensure that the correct file is being closed and has not been damaged. If the file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN1141	Automatic area for &1 is too large.

Explanation

Automatic data resides in the stack; the stack size is limited by the target machine addressabilty.

User response

Avoid large structures and / or arrays as local variables; try using dynamically allocated data. Alternatively, try to break down the procedure into several smaller procedures.

CCN1142	NOSTRICT may alter the semantics of a program.

Explanation

The NOSTRICT option has the potential to alter the semantics of a program. NOSTRICT is the default for high levels of optimization, such as OPT(3). Please refer to documentation on the STRICT/NOSTRICT option for more information.

User response

Please refer to the documentation of the STRICT/NOSTRICT option to ensure that this option will not alter the semantics of your program.

CCN1143	The register name "&2" is unknown.

Explanation

The register name in the clobber set of the asm statement is not recognized. The clobbered register information is ignored.

In the message text:

&2 is a register name.

User response

Correct the name of the clobbered register.

CCN1144	Too many registers are required in the asm statement.

Explanation

The asm statement cannot be compiled because it requires too many registers.

User response

Reduce the complexity of the asm statement.

CCN1145	The function "&1" could not be inlined into "&2".

Explanation

Function inlining is the process of replacing a call to a routine with the actual code for that routine.

In the message text:

&1 is the name of the function to be inlined. &2 is the name of the function to be inlined into.

User response

Remove the function attribute "always_inline" or refer to the inlining report for the reason why the function could not be inlined.

CCN1146	"&2", line &3: "&4" is used before it is set.

Explanation

Variable &4 is used before it is initialized at line &3.

Response

Check and make sure variable &4 is initialized before use.

CCN1147	"&2", line &3: "&4" might be used before it is set.

Explanation

Variable &4 might be used before it is initialized at line &3.

Response

Check and make sure variable &4 is initialized before use.

CCN1148	The HLASM interface module ASMA96 could not be loaded.

Explanation

The module ASMA96 was not found or virtual storage was unavailable.

Response

Make sure the ASM.SASMMOD1 dataset is in your search path.

CCN1149	Inline assembly statements caused HLASM RC=&1 with the following messages:

Explanation

Check for the cause of the assembler messages.

Response

With the LIST option, the assembler messages are placed after the assembly statements in question in the pseudo-assembly listing.

CCN1150	Inline assembly statements caused HLASM RC=&1.

Explanation

Check for the cause of the assembler messages.

Response

With the LIST option, the assembler messages are placed after the assembly statements in question in the pseudo-assembly listing.

CCN1151	Inline assembly statements caused HLASM RC=&1. Compilation terminated.

Explanation

Fix the inline assembly statements that caused the error.

Response

With the LIST option, the assembler error messages are placed after the assembly statements in error in the pseudo-assembly listing.

CCN1152   **GPR &2 in __asm on line &1 should not be clobbered.**

Explanation

The content in GPR &2 is defined by the compiler generated code. Altering this register can lead to execution error.

Response

Choose another register. See the C/C++ Language Reference for information on inline assembly statements.

CCN1153   **The function &1 is not protected against stack smashing.**

Explanation

The function &1 does not contain a valid data structure that is equal to or exceeds STACKPROTECT(SIZE(n)). Or, the compiler has determined that the function &1 does not call any other functions and thus does not require stack protection.

Response

Change the parameter SIZE(n).

CCN1501   **INTERNAL COMPILER ERROR: Procedure %1$s.**

Explanation

An internal compiler error occurred during compilation.

In the message text:

%1$s is the procedure where the error has occurred.

User response

This is an internal error. Provide the indicated error text to the IBM service representative responsible for your installation.

CCN1502   **Unable to open file %1$s for processing.**

Explanation

The system cannot open the file for processing.

In the message text:

%1$s is the processing file name.

User response

Make sure the file is available and not in use.

CCN1503   **Unable to allocate memory for processing.**

Explanation

The compiler ran out of memory generating debug information for this file. This sometimes happens with large files. Note that a very large program may produce a very large amount of debugging information.

User response

Shut down any large processes that are running or increase your TSO region size. You can also divide the file into several small sections or shorten the function.

CCN1504   **Unable to find any debug information.**

Explanation

No debug information is generated for this compilation unit.

User response

Make sure the source file contains code or data.

CCN1505   **Debug information may be incomplete.**

Explanation

The debug information generated may be corrupted or incomplete.

User response

This is an internal error. Provide the indicated error text to the IBM service representative responsible for your installation.

CCN1506   **Unable to resolve the absolute pathname for the generated debug side file.**

Explanation

The compiler cannot record the absolute pathname of the generated debug side file into an object file. A relative pathname is used instead.

User response

Make sure all the components for the generated debug side file have the proper read and execute permission set.

CCN1508   **The DLL %1$s is not found.**

Explanation

The indicated Common Debug Architecture run-time library cannot be found.

In the message text:

%1$s is the name of the Common Debug Architecture run-time library.

User response

The indicated Common Debug Architecture run-time library should be installed in the SCEERUN2 data set. Verify that the run-time library is installed properly.

CCN1509   **An incompatible DLL has been detected.**
The compiler requires LIBDDPI_DLL_VERSION to be at least 0x%1$.8x.
The version found in the system is 0x%2$.8x.

Explanation

The Common Debug Architecture run-time library version is outdated.

In the message text:

%1$x is the Common Debug Architecture run-time library that CCNEDWRT is compiled with. %2$x is the Common Debug Architecture run-time library that is currently being used.

User response

Contact the IBM service representative responsible for your installation and verify that you have the latest Common Debug Architecture run-time library installed.

CCN1510   **An incompatible DLL has been detected.**
The compiler requires LIBELF_DLL_VERSION to be at least
0x%1$.8x.
The version found in the system is 0x%2$.8x.

Explanation

The Common Debug Architecture run-time library version is outdated.

In the message text:

%1$x is the Common Debug Architecture run-time library that CCNEDWRT is compiled with. %2$x is the Common Debug Architecture run-time library that is currently being used.

User response

Contact the IBM service representative responsible for your installation and verify that you have the latest Common Debug Architecture run-time library installed.

CCN1511   **Cannot find the function %1$s in DLL %2$s.**

Explanation

The indicated function cannot be found in the Common Debug Architecture run-time library.

In the message text:

%1$s is the name of the function in the Common Debug Architecture run-time library. %2$s is the name of the Common Debug Architecture run-time library.

User response

Contact the IBM service representative responsible for your installation and verify that you have the latest Common Debug Architecture run-time library installed.

CCN2000   **Option "&1" is not recognized.**

Explanation

An invalid option was specified.

In the message text:

&1 is the option name

User response

Correct the spelling of the option.

CCN2001   **Suboption "&1" of option "&2" is not supported.**

Explanation

The invocation option contained an unsupported suboption.

In the message text:

&2 is the option name. &1 is the suboption name.

User response

Change the suboption. Check the syntax of the suboption.

CCN2002	Required parameters for option "&1" are not specified.

Explanation

This option requires that one or more parameters be specified.

In the message text:

&1 is the option name

User response

Specify appropriate parameters for the option. Check the option syntax for details.

CCN2003	Parameter "&1" of option "&2" is not supported.

Explanation

The parameter for the specified option has invalid syntax.

In the message text:

&2 is the option name. &1 is the option parameter.

User response

Change the option parameter. Check the syntax of the option parameter.

CCN2004	Option "&1" parameter error; "&2" is not a digit.

Explanation

A non-numeric character was found in the option parameter.

In the message text:

&1 is the option name. &2 is invalid character.

User response

Change the option parameter. Check the syntax of the option.

CCN2005	"&1" is not a decimal number.

Explanation

A non-numeric character was found in the option parameter.

In the message text:

&1 is the invalid character.

User response

Change the option parameter. Check the syntax of the option.

CCN2006	The name in option LOCALE (&1) is not valid. The option is reset to NOLOCALE.

Explanation

The specified locale is not installed on the host system.

User response

Change the value of the LOCALE option to the name of a locale which has been installed on the host system.

CCN2007	Suboption "&1" of option "&2" is not supported.

Explanation

The invocation option contained an unsupported suboption.

In the message text:

&2 is the option name. &1 is the suboption name.

User response

Change the suboption. Check the syntax of the suboption.

CCN2010	"&1" requires "&2" suboptions to be specified. "&3" are specified.

Explanation

An incorrect number of suboptions was specified for this option. The message identifies the number of suboptions the compiler expected and the number it actually found.

In the message text:

&1 is the option name. &2 is the number of options expected. &3 is the number of options specified.

User response

Ensure the correct number of suboptions are specified.

CCN2011   **At most "&2" suboptions must be specified for &1. "&3" are specified.**

Explanation

Too many suboptions were specified for this option.

In the message text:

&1 is the option name. &2 is the number of options expected. &3 is the number of options specified.

User response

Ensure that the maximum number of suboptions is not exceeded.

CCN2012   **"&1" requires at least "&2" suboptions to be specified. "&3" are specified.**

Explanation

Not enough suboptions were specified for this option.

In the message text:

&1 is the option name. &2 is the number of options expected. &3 is the number of options specified.

User response

Ensure that the minimum number of suboptions are specified.

CCN2013   **Suboptions "&1" and "&2" of option "&3" conflict.**

Explanation

The specified suboptions of the specified option are in conflict.

In the message text:

&3 is the option name. &1 and &2 are the suboption names.

User response

Determine which suboption is required. Remove the other suboption to eliminate the conflict.

CCN2015   **Incompatible specifications for options ARCH and TUNE.**

Explanation

As documented in the User Guide, only certain ARCH/TUNE combinations are compatible.

User response

Determine what target machine/architecture family is desired and select a compatible target machine for tuning.

CCN2020   **Option "&1" is turned on because option "&2" is specified.**

Explanation

If you specify option &2, the compiler turns on option &1 to achieve a better options combination.

In the message text:

&1 and &2 are both option names.

User response

Specify option &1 to eliminate this message.

CCN2021   **Option "&1" is ignored because option "&2" was specified.**

Explanation

Specifying the second option indicated means the first has no effect.

In the message text:

&1 and &2 are both option names.

User response

Remove one of the options.

CCN2022   **Option "&1" is not supported for IPA processing.**

Explanation

The specified option (or corresponding #pragma) is not supported for an IPA compilation. Processing is terminated.

In the message text:

&1 is an option name.

User response

Correct the option or #pragma specification, as appropriate.

CCN2023   **Option "&1" has been promoted to "&2" because option "&3" was specified.**

Explanation

Specifying the &3 option caused sufficient information to be available to support the &2 option instead of the &1 option.

In the message text:

&1, &2 and &3 are all option names.

User response

No user response is required.

CCN2024   **Option "&1" is no longer supported as a valid compiler option. Suggested alternative option(s) are: &2**

Explanation

The specified option (or corresponding #pragma) is no longer supported for this version of the compiler. Option will be ignored.

In the message text:

&1 and &2 are both option names.

User response

Remove the option or #pragma specification, as appropriate.

CCN2030   **&1**

Explanation

General informational message.

In the message text:

&1 is the detailed message text.

User response

The user response is based on the text of the message. For further information contact your Service Representative.

CCN2031   **&1**

Explanation

General warning message.

In the message text:

&1 is the detailed message text.

User response

The user response is based on the text of the message. For further information contact your Service Representative.

CCN2032   **&1**

Explanation

General error message.

In the message text:

&1 is the detailed message text.

User response

The user response is based on the text of the message. For further information contact your Service Representative.

CCN2033   **&1**

Explanation

General severe error message.

In the message text:

&1 is the detailed message text.

User response

The user response is based on the text of the message. For further information contact your Service Representative.

CCN2034   **The STRICT option is unsupported by IPA Link. The strict settings defined in the compilation unit will be used and processing continues without this option.**

Explanation

The STRICT option cannot be overridden at IPA Link.

Response

Remove the STRICT option from IPA Link.

CCN2050   **IPA Link control file: Syntax error.**

Explanation

A syntax error was detected in the IPA Link control file. Processing is terminated.

User response

Correct the IPA Link control file syntax.

CCN2051　　　　**IPA Link control file: Unmatched quote.**

Explanation

A quoted string representing a directive operand was detected in the IPA Link control file, but this string was not terminated by a matching quote before the end of file. Processing is terminated.

User response

Correct the IPA Link control file operand syntax.

CCN2052　　　　**IPA Link control file: Directive "&1" is incorrect.**

Explanation

An incorrectly specified directive was detected in the IPA Link control file. The directive is ignored, and processing continues.

In the message text:

&1 is the directive in error.

User response

Correct the specified directive in the IPA Link control file.

CCN2053　　　　**IPA Link control file: &1.**

Explanation

An error was detected in the IPA Link control file. Processing is terminated.

In the message text:

&1 is the detailed message text.

User response

Correct the specified IPA Link control file error.

CCN2059　　　　**IPA Link control file: INTERNAL COMPILER ERROR - &1.**

Explanation

An internal compiler error occurred during processing of the IPA Link control file.

In the message text:

&1 is the detailed message text.

User response

Contact your Service Representative and provide the detailed message text. For more information, visit Internal Compiler Error (www.ibm.com/support/docview.wss?uid=swg21110810).

CCN2060　　　　**CSECT name entry &1 ("&2") is not unique. It conflicts with entry &3.**

Explanation

The specified CSECT name prefix entry in the IPA Link control file duplicates an previous CSECT name prefix entry.

In the message text:

&1 and &3 are CSECT name entry numbers, &2 is the CSECT name entry.

User response

Provide a unique value for the CSECT name prefix that caused the conflict.

CCN2061　　　　**A CSECT name prefix is not specified for partition &1. The CSECT option is active.**

Explanation

The CSECT option is active, which requires that a CSECT name prefix entry be specified in the IPA Link control file for each partition in the generated object module. A system-generated name prefix has been provided for the current partition.

In the message text:

&1 is the number of the current partition.

User response

Provide one or more additional CSECT name prefixes so that each partition will have a unique name.

CCN2062　　　　**A CSECT name prefix is not specified for partition &1.**

Explanation

One or more CSECT name prefixes were specified in the IPA Link control file, but there were insufficient entries for all partitions in the generated object module. The CSECT option is not active, so these missing names are not considered an error. A system-generated name prefix has been provided for the current partition.

In the message text:

&1 is the number of the current partition.

User response

Provide one or more additional CSECT name prefixes so that each partition will have a unique name.

CCN2063   IPA Link control file: Directive "&1" is ignored because option "&2" was specified.

Explanation

The directive in the IPA Link control file is not supported with the option. The directive is ignored, and processing continues.

In the message text:

&1 is the directive. &2 is the option name

User response

Remove the directive.

CCN2100   No object files were specified as input to the IPA Link step.

Explanation

No object files were specified for IPA Link step processing.

User response

Specify at least one object file.

CCN2101   No IPA object was found.

Explanation

IPA object information was not found during IPA Link step processing.

User response

Ensure that the appropriate object files include IPA object information.

CCN2102   IPA object information is missing "&1" records.

Explanation

A damaged IPA object file was encountered during IPA Link step processing.

In the message text:

&1 is an object record type.

User response

Recompile the source file and try IPA Link step processing again. If the problem persists, call your Service Representative.

CCN2103   IPA object information has invalid "&1" record.

Explanation

A damaged IPA object file was encountered during IPA Link step processing.

In the message text:

&1 is an object record type.

User response

Recompile the source file and try IPA Link step processing again. If the problem persists, call your Service Representative.

CCN2104   Object information is missing "&1" records.

Explanation

A damaged non-IPA object file was encountered during IPA Link step processing.

In the message text:

&1 is an object record type.

User response

Recompile the source file and try IPA Link step processing again. If the problem persists, call your Service Representative.

CCN2105   Object information has an invalid "&1" record.

Explanation

A damaged non-IPA object file was encountered during IPA Link step processing.

In the message text:

&1 is an object record type.

User response

Recompile the source file and try IPA Link step processing again. If the problem persists, call your Service Representative.

CCN2106   An error was encountered during object information processing.

Explanation

A damaged or incompatible object file was encountered during IPA Link step processing.

In the message text:

&1 is an object record type.

User response

Recompile the source file and try IPA Link step processing again. If the problem persists, call your Service Representative.

CCN2107   **"&1" is not the first symbol on the object record.**

Explanation

A damaged IPA object file was encountered during IPA Link step processing.

In the message text:

&1 is an object record type.

User response

Recompile the source file and try IPA Link step processing again. If the problem persists, call your Service Representative.

CCN2108   **Object information has incorrect format.**

Explanation

An object file with an incorrect format was encountered during IPA Link step processing.

User response

Recompile the source file and try IPA Link step processing again. If the problem persists, call your Service Representative.

CCN2109   **Generated file is too big. Reduce partition size or turn off IPA.**

Explanation

The file generated by IPA exceeds encoding limits.

User response

Relink with a reduced partition size or without IPA.

CCN2110   **"&1" IPA Link control statement has no specifications.**

Explanation

An IPA Link control statement object record without any specifications was encountered during processing. The record is ignored. Processing continues.

In the message text:

&1 is either INCLUDE, LIBRARY, AUTOCALL, IMPORT or ENTRY.

User response

If the IPA Link control statement is required, provide appropriate INCLUDE, LIBRARY, or AUTOCALL, IMPORT or ENTRY specifications and repeat the step. If the record is not required, the warning message can be removed by deleting the invalid record.

CCN2111   **Invalid syntax specified on "&1" IPA Link control statement.**

Explanation

An IPA Link control statement object record with invalid syntax was encountered during processing. The record is processed up to the syntax error and the remainder of the record is ignored. Processing continues. If unmatched quotes were encountered, the IPA LINK control statement type will be listed as "UNKNOWN".

In the message text:

&1 is either INCLUDE, LIBRARY, AUTOCALL, IMPORT, ENTRY, or UNKNOWN.

User response

If the IPA Link control statement is required, correct the syntax errors and repeat the step. If the record is not required, the warning message can be removed by deleting the invalid record.

CCN2112   **Continuation record missing for "&1" IPA Link control statement.**

Explanation

An IPA Link control statement object record of type &1 was encountered with the continuation column set, but there was no subsequent record or the subsequent record was not a valid continuation record. The record is ignored and processing continues.

In the message text:

&1 is the IPA Link control statement type.

User response

Add the appropriate continuation record, or set continuation column 72 to blank if no continuation record is required.

CCN2113   **Continuation records not allowed for "&1" IPA Link control statement. This statement was ignored.**

Explanation

An IPA Link control statement of type &1 had a nonblank character in column 72. Information for a statement of this type must be specified in one record, so continuation of this record is not valid. The statement is ignored and IPA Link step processing continues.

In the message text:

&1 is the IPA Link control statement type.

User response

Correct the record if necessary, set continuation column 72 to blank, and repeat the step.

CCN2114   **More than one "&1" IPA Link control statement found.**

Explanation

More than one IPA Link control statement object record of type &1 was encountered during the processing of &2.

In the message text:

&1 is the IPA Link control statement type.

User response

No recovery is necessary unless the incorrect IPA Link control statement is selected by IPA Link error recovery, or incorrect processing was performed. In this case, remove the offending record and repeat the step.

CCN2115   **"&1" IPA Link control statement is ignored.**

Explanation

An IPA Link control statement of type &1 was found to be invalid. The record is ignored and processing continues.

In the message text:

&1 is the control statement type.

User response

Correct the record if necessary, set continuation column 72 to blank, and repeat the step.

CCN2116   **An error occurred processing the "&1" IPA Link control statement.**

Explanation

An error was encountered during processing of the IPA Link control statement. The record is ignored and processing continues.

In the message text:

&1 is either INCLUDE, LIBRARY, AUTOCALL, IMPORT or ENTRY.

User response

Ensure that the files referenced by this IPA Link control statement object record are available and in the correct format. If the problem persists, call your Service Representative.

CCN2117   **"&1" IPA Link control statement specification not supported.**

Explanation

An IPA Link control statement with a specification syntax that is unsupported by IPA Link was encountered during processing. The record is processed up to this specification, and the remainder of the record is ignored. Processing continues.

In the message text:

&1 is either INCLUDE, LIBRARY, AUTOCALL, IMPORT or ENTRY.

User response

Alter the specification to a format supported by IPA Link, or remove the specification. If the record is not required, the warning message can be removed by deleting the invalid record.

CCN2118   **A 32-bit file is being linked in 64-bit mode, or vice versa.**

Explanation

A 32-bit file is being linked in 64-bit mode (or vice versa).

User response

Change link options or compile file in different mode.

CCN2119   Noobject files used in non-IPA link step.

Explanation

One or more files generated with "NOOBJECT" were being linked directly by the linker.

User response

Recompile and link with "OBJECT" or recompile the file containing the entry point with IPA.

CCN2120   IPA Link control statement has invalid syntax:

Explanation

An IPA Link control statement object record (related to DLL resolution) with invalid syntax was encountered during processing.

User response

Prelink the DLL and generate a valid definition side-deck file.

CCN2121   IPA Link control statement not properly continued:

Explanation

An IPA Link control statement object record (related to DLL resolution) with the continuation column set was encountered, but there was no subsequent record or the subsequent record was not a valid continuation record. The record is ignored and processing continues.

User response

Prelink the DLL and generate a valid definition side-deck file.

CCN2122   Module name "&1" chosen for generated "IMPORT" IPA Link control statements.

Explanation

The default name TEMPNAME was assigned to the module in the DLL definition side-deck file.

In the message text:

&1 is a module name.

User response

Provide a "NAME" IPA Link control statement.

CCN2125   File "&1" uses a sequential format. The member name "&2" can not be specified on the "&3" IPA Link control statement.

Explanation

An IPA Link control statement specification is syntactically correct, but is incorrect for the sequential file which has been allocated. This specification is ignored, and processing continues.

In the message text:

&1 is a file name. &2 is a member name. &3 is INCLUDE.

User response

Ensure the file allocation specification is correct. Correct the file allocation or IPA Link control statement as necessary and repeat the step.

CCN2126   File "&1" uses a partitioned format. A member name must be specified on the "&2" IPA Link control statement.

Explanation

An IPA Link control statement specification is syntactically correct, but is incorrect for the partitioned file which has been allocated. This specification is ignored, and processing continues.

In the message text:

&1 is a file name. &2 is INCLUDE.

User response

Ensure the file allocation specification is correct. Correct the file allocation or IPA Link control statement as necessary and repeat the step.

CCN2127   File "&1" uses a sequential format. A partitioned file or UNIX System Services archive is required for a "&2" IPA Link control statement.

Explanation

An IPA Link control statement specification is syntactically correct, but the corresponding file uses a sequential format. This specification is ignored, and processing continues.

In the message text:

&1 is a file name. &2 is LIBRARY.

User response

Ensure the file allocation specification is correct. Correct the file allocation as necessary and repeat the step.

CCN2128   **File "&1" uses a sequential format. A partitioned file or UNIX System Services archive is required for Autocall processing.**

Explanation

The specified file is allocated to a sequential file, and is unavailable for autocall processing.

In the message text:

&1 is a file name.

User response

Ensure the file allocation specification is correct. Correct the file allocation as necessary and repeat the step.

CCN2130   **A "RENAME" IPA Link control statement can not be used for short name "&1".**

Explanation

A "RENAME" IPA Link control statement object record that attempted to rename a short name &1 to another name was encountered. "RENAME" statements are only valid for long names for which there are no corresponding short names. The "RENAME" statement is ignored and processing continues.

In the message text:

&1 is a short name.

User response

The warning message can be removed by deleting the invalid "RENAME" statement.

CCN2131   **Multiple "RENAME" IPA Link control statements are found for "&1". The first valid one is used.**

Explanation

More than one "RENAME" IPA Link control statement object record was encountered for name &1. The first "RENAME" statement with a valid output name is chosen. The "RENAME" statement is ignored and processing continues.

In the message text:

&1 is a name.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object File Map" section of the listing to determine which output name was chosen. If it was not the intended name, remove the duplicate "RENAME" statements and repeat the step.

CCN2132   **May not "RENAME" long name "&1" to another long name "&2".**

Explanation

A "RENAME" IPA Link control statement object record that attempted to rename a long name &1 to another long name &2 was encountered. The "RENAME" statement is ignored and processing continues.

In the message text:

&1 and &2 are both long names.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object File Map" section of the listing to determine which output name was chosen. If it was not the intended name, replace the invalid "RENAME" statement with a valid output name and repeat the step. The warning message can be removed by deleting the invalid RENAME statement.

CCN2133   **May not "RENAME" defined long name "&1" to defined name "&2".**

Explanation

A "RENAME" IPA Link control statement object record that attempted to rename a defined long name &1 to another defined name &2 was encountered. The "RENAME" statement is ignored and processing continues.

In the message text:

&1 is a long name. &2 is a defined name.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object File Map" section of the listing to determine which output name was chosen. If it was not the intended name, replace the invalid "RENAME" statement with a valid output name and repeat the step. The warning message can be removed by deleting the invalid RENAME statement.

CCN2134   **"RENAME" of "&1" to "&2" is ignored since "&2" is the target of another "RENAME".**

Explanation

Multiple "RENAME" IPA Link control statement object records that attempted to rename two different names to the same name &2 were encountered. The "RENAME" statement is ignored and processing continues.

In the message text:

&1 is a long name. &2 is a defined name.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object File Map" section of the listing to determine which name was renamed to &2. If it was not the intended name, change the name and repeat the step. The warning message can be removed by deleting the extra "RENAME" statements.

CCN2140   **"&1" is mapped to "&2" by the IPA(UPCASE) option. "&3" is an alternative matching definition name.**

Explanation

"&1" is an external symbol reference that maps to multiple definitions due to the IPA(UPCASE) option. Definition "&2" was selected. "&3" is another definition which matches this name, but was not used.

In the message text:

&1, &2 and &3 are names.

User response

If both names (&1 and &2) correspond to the same object the warning can be ignored. If the names do not correspond to the same object or if the warning is to be removed, do one of the following:

- Change one of the names in the source routine.
- Use #pragma map in the source routine for one of the names.

CCN2141   **"&1" is mapped to "&2".**

Explanation

External name "&1" has been replaced by "&2". IPA Link processing required a name that was limited to 8 characters.

In the message text:

&1 and &2 are names.

User response

No user response is required. If you require a specific external name for "&1", use #pragma map in the program source. Any additional names that were mapped to "&1" (and hence "&2") because of IPA(UPCASE) will require equivalent #pragma map statements.

CCN2142   **Unable to map "&1" and "&2" to a common name during IPA(UPCASE) processing.**

Explanation

Due to references by non-IPA objects, a common external name can not be determined during IPA(UPCASE) processing. This will occur if both "&1" and "&2" are referenced by non-IPA objects, or if either is referenced by non-IPA objects and the common name is longer than 8 characters.

In the message text:

&1 and &2 are names.

User response

Modify the program source so that the external names are consistent, and 8 characters or less in length.

CCN2143   **Unable to map "&1" to "&2" within same Compilation Unit during IPA(UPCASE) processing.**

Explanation

"&1" is an external symbol that maps to the symbol "&2" within the same Compilation Unit due to the IPA(UPCASE) option. Mapping of symbols in this manner is not supported.

In the message text:

&1 and &2 are names.

User response

Modify the program source so that the external names are consistent. If IPA(UPCASE) resolution is desired, split the program source so that each symbol is defined in a different Compilation Unit.

CCN2150   **Invalid C370LIB-directory encountered.**

Explanation

The specified library file contains an invalid or damaged C370LIB-directory.

User response

Use the C370LIB DIR command to recreate the C370LIB-directory, and repeat the step.

CCN2151	Library does not contain a C370LIB-directory.

Explanation

The specified library file does not contain a C370LIB-directory required to perform the command.

User response

The library was not created with the C370LIB command. Use the C370LIB DIR command to create the C370LIB-directory, and repeat the step.

CCN2152	Member "&1" not found in library.

Explanation

The specified member &1 was not found in the library. Processing continues.

In the message text:

&1 is a library member name.

User response

Use the C370LIB MAP command to display the names of library members.

CCN2153	Unable to access library file.

Explanation

An error was encountered during processing of the specified "LIBRARY" IPA Link control statement. The record is ignored and processing continues.

User response

Ensure that the files referenced by this IPA Link control statement object record are available and in the correct format. If the problem persists, call your Service Representative.

CCN2155	&1 sequential files in library "&2" allocation were ignored.

Explanation

When the list of files allocated to the specified DD was extracted, both sequential and partitioned format files were found. The sequential files were ignored.

In the message text:

&1 is the number of sequential files. &2 is a library DD name.

User response

Correct the library allocation to eliminate the sequential files.

CCN2160	Invalid symbol table encountered in archive library.

Explanation

The specified archive library file contains invalid information in its symbol table. Processing continues.

User response

Rebuild the archive library.

CCN2161	Archive library does not contain a symbol table.

Explanation

The symbol table for the specified archive library file could not be found.

User response

Rebuild the archive library.

CCN2170	Unresolved "IMPORT" references are detected.

Explanation

Unresolved objects were encountered at IPA Link processing termination. Other user objects are required.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the objects in question. To correct unresolved references to user objects, include the user objects during IPA Link processing.

CCN2171	Unresolved "IMPORT" references are detected:

Explanation

The listed unresolved objects were encountered at IPA Link processing termination. Other user objects are required.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the objects in question. To correct unresolved references to user objects, include the user objects during IPA Link processing.

CCN2172	Unresolved references could not be imported.

Explanation

The same symbol was referenced in both DLL and non-DLL code. The DLL reference could have been satisfied by an "IMPORT" IPA Link control statement which was processed, but the non-DLL reference could not.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the symbols in question. You must either supply a definition for the referenced symbol, or use the DLL compiler option to recompile the code containing the non-DLL reference so that it becomes a DLL reference.

CCN2173	Unresolved references could not be imported:

Explanation

The listed symbols were referenced in both DLL and non-DLL code. The DLL reference could have been satisfied by an "IMPORT" IPA Link control statement which was processed, but the non-DLL reference could not.

User response

You must either supply a definition for the referenced symbol, or use the DLL compiler option to recompile the code containing the non-DLL reference so that it becomes a DLL reference.

CCN2174	Duplicate "IMPORT" definitions are detected.

Explanation

A name referenced in DLL code was not defined within the application, but more than one "IMPORT" IPA

Link control statement was detected with that symbol name. The first one encountered was used.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the objects in question, and define these objects once.

CCN2175	Duplicate "IMPORT" definitions are detected:

Explanation

The listed objects were defined multiple times.

User response

Define these objects once.

CCN2177	"ENTRY" symbol "&1" not found.

Explanation

An "ENTRY" IPA Link control statement object record that attempted to specify a program entry point was encountered, but no symbol by this name is present in the application program.

In the message text:

&1 is a symbol name.

User response

If the IPA Link control statement is required, provide an object file which defines the symbol, and repeat the step. If the record is not required, the error message can be removed by deleting the invalid record.

CCN2178	"ENTRY" symbol "&1" not valid.

Explanation

An "ENTRY" IPA Link control statement object record that attempted to specify a program entry point was encountered, but the specified symbol is a reference, or aggregate member.

In the message text:

&1 is a symbol name.

User response

If the IPA Link control statement is required, provide an object file which defines a valid symbol, and repeat the step. If the record is not required, the error message can be removed by deleting the invalid record.

CCN2180   **Load Module information has invalid "&1" record.**

Explanation

A damaged or incompatible Load Module library member was encountered during IPA Link processing.

In the message text:

&1 is an Load Module record type.

User response

Recompile the source file and try IPA Link processing again. If the problem persists, call your Service Representative.

CCN2181   **An error was encountered during Load Module information processing.**

Explanation

A damaged or incompatible Load Module library member was encountered during IPA Link processing.

In the message text:

&1 is an Load Module record type.

User response

Recompile the source file and try IPA Link processing again. If the problem persists, call your Service Representative.

CCN2182   **Load Module information has incorrect format.**

Explanation

A Load Module library member with an incorrect format was encountered during IPA Link processing.

User response

Recompile the source file and try IPA Link processing again. If the problem persists, call your Service Representative.

CCN2183   **Program Object file format is not supported by IPA Link step processing.**

Explanation

During the link portion of IPA Link step processing, an attempt was made to extract object information from a Program Object file. IPA Link step processing supports object information in the form of object modules, and Load Module library members. Program Object files which are generated by the Program Management Binder are not supported.

User response

Repackage the Program Object as either an object module or a Load Module library member, and try IPA Link processing again.

CCN2184   **IPA Object file "&1" has been compiled with an incompatible version of IPA.**

Explanation

The IPA Object format in "&1" is incompatible with the current compiler.

User response

Recompile the file with the current compiler.

CCN2185   **The correct decryption key for object file "&1" was not specified.**

Explanation

The file "&1" was encrypted with different key than the one(s) specified.

User response

Include the correct key or link without IPA.

CCN2187   **"&1" option is incompatible with an IPA object file compiled with "&2" option.**

Explanation

An option used to compile the IPA object file is incompatible with an option used at IPA link.

User response

Recompile the IPA object file with a compatible option, or specify a compatible option at IPA link.

CCN2200   **Unresolved references to writable static objects are detected.**

Explanation

Undefined writable static objects were encountered at IPA Link step processing termination. Other user objects are required.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the objects in question, and include these objects during IPA Link processing.

CCN2201	Undefined writable static objects are detected:

Explanation

The listed writable static objects were undefined at IPA Link processing termination.

User response

Include these objects during IPA Link processing.

CCN2202	Unresolved references to writable static objects are detected:

Explanation

Undefined writable static objects or unresolved objects referring to writable static objects were encountered at IPA Link processing termination. Other user objects are required.

User response

Include these objects during IPA Link processing.

CCN2203	Unresolved references to objects are detected.

Explanation

Unresolved objects were encountered at IPA Link processing termination. Other user objects are required.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the objects in question. To correct unresolved references to user objects, include the required objects during IPA Link processing.

CCN2204	Unresolved references to objects are detected:

Explanation

The listed unresolved objects were encountered at IPA Link processing termination. Other user objects are required.

User response

To correct the unresolved references, include the required objects during IPA Link step processing.

CCN2205	Unresolved reference to symbol "&1".

Explanation

The listed unresolved objects were encountered at IPA Link processing termination. Other user objects are required.

In the message text:

User response

To correct the unresolved references, include the required objects during IPA Link step processing.

CCN2206	Unresolved reference to symbol "&1".

Explanation

The listed unresolved objects were encountered at IPA Link processing termination. Other user objects are required.

In the message text:

User response

To correct the unresolved references, include the required objects during IPA Link step processing.

CCN2210	Duplicate writable static objects are detected.

Explanation

Writable static objects were defined multiple times.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the objects in question, and define the required objects once.

CCN2211	Duplicate writable static objects are detected:

Explanation

The listed writable static objects were defined multiple times.

User response

Define these objects once.

CCN2212   Duplicate objects are detected.

Explanation

Objects were defined multiple times.

User response

Specify the IPA(LINK,MAP) option during processing. Examine the "Object Resolution Warnings" section of the listing to find the objects in question, and define these objects once.

CCN2213   Duplicate objects are detected:

Explanation

The listed objects were defined multiple times.

User response

Define the objects once.

CCN2220   Duplicate writable static object "&1" is detected with different sizes. The largest size is used.

Explanation

The listed writable static object was defined multiple times with different sizes. The larger of the different sizes was used. Incorrect execution could occur unless the object is defined consistently.

In the message text:

&1 is a writable static object name.

User response

Define the objects consistently.

CCN2221   Duplicate object "&1" is detected with different sizes. The largest size is used.

Explanation

The listed object was defined multiple times with different sizes. The larger of the different sizes is used. Incorrect execution could occur unless the object is defined consistently.

In the message text:

&1 is an object name.

User response

Define these objects consistently.

CCN2229   No exported symbols found.

Explanation

After the IPA object files were linked, an unsuccessful attempt was made to locate at least one exported symbols.

User response

Specify at least one exported symbol contained in the IPA object files.

CCN2230   Program entry point not found.

Explanation

After the IPA object files were linked, an unsuccessful attempt was made to identify the program entry point (normally the "main" function).

User response

Provide the IPA object file containing the program entry point.

CCN2231   More than one entry point was found.

Explanation

After the IPA object files were linked, multiple possible program entry points were found.

User response

Eliminate the IPA object files containing the extra program entry points.

CCN2232   Duplicate definition of symbol "&1" ignored.

Explanation

A duplicate definition of the specified symbol has been encountered in the specified file. It is ignored.

In the message text:

&1 is the symbol name.

User response

If possible, eliminate the duplicate symbol definition from the set of input files provided to the IPA Link step.

CCN2233   **Duplicate definition of symbol "&1" in import list is ignored.**

Explanation

A duplicate definition of the specified symbol has been encountered in an import list in the specified file. It is ignored.

In the message text:

&1 is the symbol name.

User response

Eliminate the duplicate import definition for the specified symbol.

CCN2240   **IPA object files "&1" and "&2" have been compiled with differing settings for the "&3" option.**

Explanation

The IPA object files were compiled using conflicting settings for the specified option. A final common option setting will be selected. Alternatively, a common override can be specified during IPA Link invocation.

In the message text:

&1 and &2 are object file names, and &3 is an option name.

User response

Ensure that the final option setting is appropriate. The warning message can be removed by recompiling one or both source files with the same option setting.

CCN2241   **The "&1" option will be used.**

Explanation

This is the final common option setting selected after IPA object files were found to be in conflict.

In the message text:

&1 is an option name.

User response

Ensure that the final option setting is appropriate. The warning message can be removed by recompiling one or both source files with the same option setting.

CCN2242   **IPA object files "&1" and "&2" contain code targeted for different machine architectures.**

Explanation

The IPA object files were compiled with conflicting machine architectures. A final common machine architecture will be selected.

In the message text:

&1 and &2 are object file names.

User response

Ensure that the final machine architecture is appropriate. The warning message can be removed by recompiling one or both source files so that consistent ARCH options that specify the same machine architecture are used.

CCN2243   **The "&1" machine architecture will be used.**

Explanation

This is the final machine architecture selected after IPA object files were found to be in conflict.

In the message text:

&1 is a machine architecture id.

User response

Ensure that the final machine architecture is appropriate. The warning message can be removed by recompiling one or both source files so that consistent ARCH options that specify the same machine architecture are used.

CCN2244   **IPA object files "&1" and "&2" contain code targeted for different operating environments.**

Explanation

The IPA object files were compiled using conflicting operating environments. A final common operating environment will be selected.

In the message text:

&1 and &2 are object file names.

User response

Ensure that the final target operating environment is appropriate. The warning message can be removed by recompiling one or both source files for the same operating environment.

CCN2245   **The "&1" operating environment will be used.**

Explanation

This is the final operating environment selected after IPA object files were found to be in conflict.

In the message text:

&1 is an operating environment id.

User response

Ensure that the final target operating environment is appropriate. The warning message can be removed by recompiling one or both source files for the same operating environment.

CCN2246	IPA object files "&1" and "&2" were generated from different source languages.

Explanation

The IPA object files were produced by compilers for different languages. The IPA object has been transformed as required to handle this situation.

In the message text:

&1 and &2 are object file names.

User response

No user response is required.

CCN2247	IPA object files "&1" and "&2" were generated by different compiler versions.

Explanation

The IPA object files were produced by different versions of the compiler. The older IPA object has been transformed to the later version.

In the message text:

&1 and &2 are object file names.

User response

No user response is required.

CCN2248	The code page for one or more IPA object files differs from the code page "&1", used during IPA Link processing.

Explanation

IPA object files contain code page identification if the LOCALE option is active when they are originally compiled. During IPA Link processing with the LOCALE option active, one or more IPA object files were encountered that had a code page (specified via the LOCALE option) which differs from that used during IPA Link processing. Character data will remain in the code page in which it was originally compiled.

In the message text:

&1 is a code page name.

User response

No user response is required.

CCN2250	Option "&1" not available because one or more IPA object files were compiled with option "&2".

Explanation

The specified option is not available during code generation for the current partition, because one or more IPA object files contain insufficient information to support it. A final common option will be selected.

In the message text:

&1 and &2 are option names.

User response

No recovery is required. However, greater optimization potential may occur if one or both of the module objects were recompiled to eliminate the option setting conflict.

CCN2260	Subprogram specified exceeds size limit: &1

Explanation

The ACU for the subprogram exceeds the LIMIT specified in the INLINE suboption.

In the message text:

&1 is the Subprogram name.

User response

Increase LIMIT if it is feasible to do so.

CCN2261	Subprogram specified is (or grows) too large to be inlined: &1

Explanation

This occurs when a subprogram is too large to be inlined into another subprogram.

In the message text:

&1 is the subprogram name.

User response

Use #pragma inline if it is feasible to do so.

CCN2262	Some calls to subprogram specified cannot be inlined: &1

Explanation

At least one call is either directly recursive, or the wrong number of parameters were specified.

In the message text:

&1 is the subprogram name.

User response

Check all calls to the subprogram specified and make sure that the number of parameters match the subprogram definition.

CCN2263	Automatic storage for subprogram specified increased to over &1 bytes: &2

Explanation

The size of automatic storage for subprogram increased by at least 4 KB due to inlining.

In the message text:

&1 is the automatic storage limit. &2 is the subprogram name.

User response

If feasible to do so, prevent the inlining of subprograms that have large auto storage.

CCN2265	Inlining of specified subprogram failed due to the presence of a global label: &1

Explanation

At least one call could not be inlined due to the presence of a global label.

In the message text:

&1 is the subprogram name.

User response

Minimize the use of global labels in your application. Their presence will inhibit global inlining.

CCN2266	Inlining of specified subprogram failed due to the presence of a C++ exception handler: &1

Explanation

At least one call could not be inlined due to the presence of a C++ exception handler.

In the message text:

&1 is the subprogram name.

User response

Minimize the use of C++ exception handlers in your application. Their presence will inhibit global inlining.

CCN2267	Inlining of specified subprogram failed due to the presence of variable arguments: &1

Explanation

At least one call could not be inlined due to the presence of variable arguments.

In the message text:

&1 is the subprogram name.

User response

No user response is required.

CCN2268	Inlining of subprogram "&1" into subprogram "&2" failed due to a conflict in options settings.

Explanation

The specified call could not be inlined due to incompatible options settings for the IPA object files that contain the two programs.

In the message text:

&1 and &2 are subprogram names.

User response

Use compatible options during the IPA Compile step.

CCN2269	Inlining of subprogram "&1" into subprogram "&2" failed due to a type mismatch in argument "&3".

Explanation

The specified call could not be inlined due to incompatible types for the specified argument number, where "&1" is the first argument.

In the message text:

&1 and &2 are subprogram names. &3 is the parameter index

User response

Correct the program to use compatible types for all arguments.

CCN2270   **Subprogram "&1" has been inlined into subprogram "&2". One or more unexpected extra parameters were ignored.**

Explanation

The specified call was inlined, but one or more parameters on the call were not required and were ignored.

In the message text:

&1 and &2 are subprogram names.

User response

Eliminate the extra parameters.

CCN2271   **Subprogram "&1" has been inlined into subprogram "&2". One or more arguments were not supplied, so the values are undefined.**

Explanation

The specified call was inlined, but one or more parameters were omitted on the call. Values for these arguments are indeterminate, so the operation of the subprogram is undefined.

In the message text:

&1 and &2 are subprogram names.

User response

Specify all parameters actually required by the called subprogram.

CCN2280   **A type mismatch was detected for symbol "&1".**

Explanation

An instance of the specified subprogram was found where one or more parameters were of an unexpected type.

In the message text:

&1 is a subprogram name.

User response

Correct the program to use parameter types compatible with the function definition.

CCN2281   **Function return types "&1" and "&2" for subprogram "&3" do not match.**

Explanation

An instance of the specified subprogram was found with an unexpected type for the function return value.

In the message text:

&1 and &2 are return type names. &3 is a subprogram name.

User response

Correct the program to use a return type compatible with the function definition.

CCN2282   **Subprogram "&1" has the wrong number of formal parameters.**

Explanation

The number of formal parameters for the definition of the given subprogram does not match the number of formal parameters for the declaration of the subprogram.

In the message text:

&1 is a subprogram name.

User response

Correct the program to use a consistent number of formal parameters for the subprogram.

CCN2283   **A linkage mismatch was detected for symbol "&1".**

Explanation

An instance of the specified subprogram was found which uses a linkage incompatible with the calling function.

In the message text:

&1 is a symbol name.

User response

Correct the program to ensure consistent linkage across all objects.

CCN2299   **Some optimizations may be inhibited.**

Explanation

During optimization of the IPA object, a problem was encountered that prevent the use of all available optimization techniques. These specific problems are identified in separate messages.

User response

Correct the problem which inhibits optimization.

CCN2300	Export symbol "&1" not found.

Explanation

An "export" directive entry for the specified symbol was present in the IPA Link control file, but no symbol by this name is present in the application program.

In the message text:

&1 is a symbol name.

User response

Correct the IPA Link control file directive.

CCN2301	External subprogram "&1" not found. Could not mark as "pure".

Explanation

A "pure" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2302	External subprogram "&1" not found. Could not mark as "isolated".

Explanation

A "isolated" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2303	External subprogram "&1" not found. Could not mark as "safe".

Explanation

A "safe" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2304	External subprogram "&1" not found. Could not mark as "unknown".

Explanation

An "unknown" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2305	External subprogram "&1" not found. Could not mark as "low frequency".

Explanation

A "lowfreq" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2306	External subprogram "&1" not found. Could not mark as "an exit".

Explanation

A "exits" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2307	External symbol "&1" not found. Could not mark as "retain".

Explanation

A "retain" directive entry for the specified symbol was present in the IPA Link control file, but no symbol by this name is present in the application program.

In the message text:

&1 is a symbol name.

User response

Correct the IPA Link control file directive.

CCN2308	Regular expression "&1" error: &2.

Explanation

The regular expression is incorrectly specified.

In the message text:

&1 is a regular expression.

User response

Correct the regular expression "&1".

CCN2310	External subprogram "&1" not found. Could not mark as "inline".

Explanation

An "inline" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2311	EXternal subprogram "&1" not found. Could not mark as "do not inline".

Explanation

A "noinline" directive entry for the specified subprogram was present in the IPA Link control file, but no subprogram by this name is present in the application program.

In the message text:

&1 is a subprogram name.

User response

Correct the IPA Link control file directive.

CCN2312	Could not inline calls from "&1" to "&2" as neither external subprogram was found.

Explanation

An "inline" directive entry for calls between the specified subprograms was present in the IPA Link control file, but no subprograms by these names are present in the application program.

In the message text:

&1 and &2 are subprogram names.

User response

Correct the IPA Link control file directive.

CCN2313	Could not inhibit inlining calls from "&1" to "&2" as neither external subprogram was found.

Explanation

A "noinline" directive entry for calls between the specified subprograms was present in the IPA Link control file, but no subprograms by these names are present in the application program.

In the message text:

&1 and &2 are subprogram names.

User response

Correct the IPA Link control file directive.

CCN2314	Could not inline calls from "&1" to "&2" as external subprogram "&3" was not found.

Explanation

An "inline" directive entry for calls between the specified subprograms was present in the IPA Link control file, but no subprogram with the specified name is present in the application program.

In the message text:

&1, &2 and &3 are subprogram names.

User response

Correct the IPA Link control file directive.

CCN2315	Could not inhibit inlining calls from "&1" to "&2" as external subprogram "&3" was not found.

Explanation

A "noinline" directive entry for calls between the specified subprograms was present in the IPA Link control file, but no subprogram with the specified name is present in the application program.

In the message text:

&1, &2 and &3 are subprogram names.

User response

Correct the IPA Link control file directive.

CCN2316	Could not find any calls from "&1" to "&2" to inline.

Explanation

An "inline" directive entry for calls between the specified subprograms was present in the IPA Link control file, but no such calls are present in the application program.

In the message text:

&1 and &2 are subprogram names.

User response

Delete the IPA Link control file directive.

CCN2317	Could not find any calls from "&1" to "&2" to inhibit from inlining.

Explanation

A "noinline" directive entry for calls between the specified subprograms was present in the IPA Link control file, but no such calls are present in the application program.

In the message text:

&1 and &2 are subprogram names.

User response

Delete the IPA Link control file directive.

CCN2320	The minimum size of partition &1 exceeds the partition size limit.

Explanation

The program information which must be contained within the current partition is larger than the current partition size limit. This may be because the partition contains a single large subprogram.

In the message text:

&1 is the number of the current partition.

User response

Use the IPA Link "partition" directive to specify a larger partition size limit.

CCN2340	Code generation was not performed due to previously detected errors. Object file not created.

Explanation

The completion of the IPA Link step is not possible due to errors that were previously detected. The generation of code and data from the IPA object information will not be performed, and no object file will be generated.

User response

Eliminate the cause of the error conditions.

CCN2341	Code generation for partition &1 terminated due to previous errors.

Explanation

The generation of object code and data for the current partition has been terminated due to error conditions detected during processing. Processing continues to allow further errors to be detected, but an incomplete object file will be generated.

In the message text:

&1 is the number of the current partition.

User response

Eliminate the cause of the error conditions.

CCN2342 **Code generation for partition &1 bypassed due to previous errors.**

Explanation

The generation of object code and data for the current partition has been bypassed due to error conditions detected when processing a previous partition. Processing continues to allow further errors to be detected, but an incomplete object file will be generated.

In the message text:

&1 is the number of the current partition.

User response

Eliminate the cause of the error conditions.

CCN2345 **An error occurred during code generation. The code generation return code was &1.**

Explanation

During the generation of code for the current partition, an error was detected. One or more messages may be issued when this occurs.

In the message text:

&1 is the code generation return code.

User response

Refer to the responses for these messages, and perform the suggested error recovery actions.

CCN2400 **File "&1" not found.**

Explanation

The compiler could not locate the specified file.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2401 **Object file "&1" not found.**

Explanation

The compiler could not locate the specified object file.

In the message text:

&1 is an object file name.

User response

Ensure the file name is correct. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2404 **IPA Link control file "&1" not found.**

Explanation

The compiler could not locate the specified IPA Link control file.

In the message text:

&1 is an IPA Link control file name.

User response

Ensure the file name is correct. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2406 **Load Module library member "&1" not found.**

Explanation

The compiler could not locate the specified member of the Load Module library.

In the message text:

&1 is a Load Module library member name.

User response

Ensure the member name and Load Module library names are correct. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2407 **File "&1" not found.**

Explanation

The compiler could not locate the specified file.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2420 **File "&1" has invalid format.**

Explanation

The specified file was located, but did not have the correct format.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Correct the file as necessary and repeat the step.

CCN2425   File "&1" has invalid attributes.

Explanation

The specified file was located, but did not have the correct attributes.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Correct the file as necessary and repeat the step.

CCN2430   File "&1" is not allocated.

Explanation

The specified file is not allocated, and is unavailable for processing.

In the message text:

&1 is a file name.

User response

Ensure the file allocation specification is correct. Correct the file allocation as necessary and repeat the step.

CCN2431   File "&1" is not allocated. Autocall will not be performed.

Explanation

The specified file is not allocated, and is unavailable for autocall processing.

In the message text:

&1 is a file name.

User response

Ensure the file allocation specification is correct. Correct the file allocation as necessary and repeat the step.

CCN2440   Unable to open file "&1", for read.

Explanation

The compiler could not open the specified file. This file was being opened with the intent of reading the file contents.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Ensure that the correct file is being read and has not been damaged. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2441   Unable to open file "&1", for write.

Explanation

The compiler could not open the specified file. This file was being opened with the intent of writing new information.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Ensure that the correct file is specified. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2442   An error occurred while reading file "&1".

Explanation

The compiler detected an error while reading from the specified file.

In the message text:

&1 is a file name.

User response

Ensure that the correct file is being read and has not been damaged.

CCN2443   An error occurred while writing to file "&1".

Explanation

The compiler detected an error while writing to the specified file.

In the message text:

&1 is a file name.

User response

Ensure that the correct file is specified.

CCN2445	Unable to close file "&1", after write.

Explanation

The compiler could not close the specified file after writing new information.

In the message text:

&1 is a file name.

User response

Ensure that the correct file is specified, and that there is sufficient free space.

CCN2446	File "&1" is empty.

Explanation

The compiler opened the specified file, but it was empty when an attempt was made to read the file contents.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Ensure that the correct file is being read and has not been damaged.

CCN2447	Premature end occurred while reading file "&1".

Explanation

The compiler opened the specified file and began processing the file contents. The end of file was reached before all data was processed. Processing continues with the next file.

In the message text:

&1 is a file name.

User response

Ensure that the correct file is being read and has not been damaged.

CCN2448	An error occurred while writing to file "&1", possibly due to lack of space.

Explanation

The compiler detected an error while writing to the specified file.

In the message text:

&1 is a file name.

User response

Ensure that the correct file is specified, and sufficient storage has been allocated for it.

CCN2451	Unable to create temporary file "&1".

Explanation

The compiler could not create the specified temporary file.

In the message text:

&1 is a file name.

User response

The file may be locked by another process or access may be denied because of insufficient permission.

CCN2460	Listing file "&1" is full.

Explanation

The compiler detected that there is insufficient free space to continue writing to the listing file. Compilation continues, without further updates to the listing file.

In the message text:

&1 is the listing file name.

User response

Ensure that the correct listing file is specified, and that there is sufficient free space.

CCN2461	Listing file "&1" closed prematurely.

Explanation

The compiler detected an error while writing to the listing file. Compilation continues, without further updates to the listing file.

In the message text:

&1 is the listing file name.

User response

Ensure that the correct listing file is specified.

CCN2462	Unable to write to temporary file "&1".

Explanation

The compiler detected an error while writing to the temporary file.

In the message text:

&1 is the temporary file name.

User response

Ensure there is enough disk space.

CCN2463	Unable to create a temporary file.

Explanation

The compiler could not create a temporary file.

User response

Check the system documentation on creating temporary files.

CCN2464	Profiling data matching that from the PDF1 phase could not be found for function "&1" in file "&2". Source files and/or compilation options may differ between the PDF1 and PDF2 phases.

Explanation

The structure or the control flow of the function has changed from the PDF1 phase to the PDF2 phase.

User response

Ensure that the source files and compilation options match between the PDF1 and PDF2 phases.

CCN2465	Data for function "&1" does not match the profiling data found in file "&2" from the PDF1 phase. Source files and/or compilation

options may differ between the PDF1 and PDF2 phases.

Explanation

The function has changed in a minor way from the PDF1 phase to the PDF2 phase.

User response

Ensure that the source files and compilation options match between the PDF1 and PDF2 phases.

CCN2466	Profiling data for function "&1" is not found in file "&2" from the PDF1 phase. Source files may differ between the PDF1 and PDF2 phases, or the function may not be called.

Explanation

The compiler cannot find any data for the function in the pdf profile generated by the training run.

User response

Ensure that the source files and compilation options match between the PDF1 and PDF2 phases and that the function is called.

CCN2467	Unable to create pdf file because the specified file name is too long.

Explanation

The length for the PDF file name exceeds 256 characters.

User response

Ensure PDF file name length is less than 256 characters.

CCN2468	Unable to open file "&1", for read.

Explanation

The compiler could not open the specified file. This file was being opened with the intent of reading the file contents.

In the message text:

&1 is a file name.

User response

Ensure the file name is correct. Ensure that the correct file is being read and has not been damaged. If the

file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2472   **Cache miss profiling is not supported on this platform.**

Explanation

The compiler will not profile cache misses as part of PDF.

User response

Ensure that the target machine supports cache miss profiling (PMAPI).

CCN2474   **New definition of function "&1" is inconsistent with profiling data found in file "&2" from the PDF1 phase. Source files and/or compilation options may differ between the PDF1 and PDF2 phases. PDF Information will be ignored for function "&3".**

Explanation

The function has changed in a minor way from the PDF1 phase to the PDF2 phase.

User response

Ensure that the source files and compilation options match between the PDF1 and PDF2 phases.

CCN2475   **Profiling data found in file "&1" from the PDF1 phase is &2% relevant.**

Explanation

In the message text:

&1 is the PDF filename. &2 is a percentage ratio of profiling data that can be used in PDF2.

User response

No user response is required.

CCN2476   **An error occurred when writing to PDF or PDF map file "&1".**

Explanation

The compiler could not write to the specified PDF or PDF map file(s).

User response

Ensure the file name and/or path specified in pdfname or PDFDIR is correct. If the file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission.

CCN2477   **One or more input object files were not compiled with PDF.**

Explanation

The compiler found object files that were not compiled with IPA(PDF1) or IPA(PDF2) on the IPA Compile step.

User response

For additional profiling, ensure that all input object files have been compiled with IPA(PDF1) or IPA(PDF2) on the IPA Compile step.

CCN2490   **COMPILER LIMIT EXCEEDED: Insufficient virtual storage.**

Explanation

The compiler ran out of memory attempting to compile the file. This sometimes happens with large files or programs with large functions. Note that very large programs limit the amount of optimization that can be done.

User response

Redefine your virtual storage to a larger size. If sufficient storage is not available, you can try various approaches, such as shut down any large processes that are running, ensure your swap path is large enough, try recompiling the program with a lower level of optimization or without interprocedural analysis.

CCN2492   **INTERNAL COMPILER ERROR: Error &1 in Procedure &2.**

Explanation

An internal compiler error occurred during compilation.

User response

Contact your Service Representative.
For more information, visit Internal Compiler Error (www.ibm.com/support/docview.wss?uid=swg21110810).

CCN2493   **INTERNAL COMPILER ERROR: &1.**

Explanation

An internal compiler error occurred during compilation.

User response

Contact your Service Representative.
For more information, visit Internal
Compiler Error (www.ibm.com/support/docview.wss?
uid=swg21110810).

CCN2497	File &1 call failed. System error description: "&2".

Explanation

The specified file operation failed. The system error description describes the reason for the failure.

User response

Based on the system error description, change the environment in order to facilitate the given file system call. For example, if the process is not authorized to perform the given action, then change the appropriate permissions.

CCN2498	The listing destination when SPLITLIST is specified must be either a PDS, PDSE or z/OS UNIX System Services file system directory. Processing is terminated.

Explanation

The listing destination is not a PDS, PDSE or z/OS UNIX System Services file system directory.

User response

Change the destination to the name of a PDS, PDSE or z/OS UNIX System Services file system directory or remove the SPLITLIST option from the list of command line options passed to the IPA Link phase.

CCN2506	"&1", line &2: "&3" is used before it is set.

Explanation

An uninitialized variable detected by the compiler.

In the message text:

&1 is the filename. &2 is the line number. &3 is the variable.

User response

Correct the error on the program source.

CCN2507	"&1", line &2: "&3" might be used before it is set.

Explanation

A potential uninitialized variable detected by the compiler.

In the message text:

&1 is the filename. &2 is the line number. &3 is the variable.

User response

Correct the error on the program source.

CCN3001	INTERNAL COMPILER ERROR: Procedure &1.

Explanation

An internal compiler error occurred during compilation.

In the message text:

&1 is a procedure name.

For more information visit Internal
Compiler Error (www.ibm.com/support/docview.wss?
uid=swg21110810).

User response

Contact your service representative.

CCN3002	COMPILER ERROR: Feature not implemented: &1.

Explanation

An error occurred during compilation.

In the message text:

&1 is a feature name.

User response

See the C/C++ Language Reference for a description of supported features.

CCN3003	Width of a bit field of type "&1" cannot exceed &2.

Explanation

The length of the bit field must not exceed the maximum bit size of the bit field's type.

In the message text:

&1 is a type name, &2 is an integer (number of bits).

User response

Define the bit field length to be less than or equal to the maximum bit size of the bit field type.

CCN3004	pragma must appear before use of identifier &1.

Explanation

The identifier is modified by the pragma after the pragma is seen.

In the message text:

&1 is an identifier.

User response

Move the pragma so that it appears before the identifier is used.

CCN3005	Error in message set &1, unable to retrieve message &2.

Explanation

Message cannot be retrieved from the message catalog.

In the message text:

&1 is a message catalog set, &2 is a message number.

User response

Check the installation procedure to see if the message catalog has been properly installed.

CCN3006	Label &1 is undefined.

Explanation

A label must be visible in the current function scope if it is used in an expression.

In the message text:

&1 is a label.

User response

Declare a label with that name in the current function scope.

CCN3007	"&1" is undefined.

Explanation

A C identifier must be declared before it is used in an expression.

In the message text:

&1 is an identifier.

User response

Declare an identifier with that name in the current scope or in a higher scope.

CCN3008	The argument is not valid for the pragma directive.

Explanation

pragma does not recognize the argument.

User response

Remove the argument or change its format.

CCN3009	Bit field &1 must be of type signed int, unsigned int or int.

Explanation

The type of the bit field is not a signed int, unsigned int, or an int.

In the message text:

&1 is an identifier.

User response

Define the bit field with a type signed int or unsigned int.

CCN3010	Macro &1 invoked with a null argument for parameter &2.

Explanation

No argument was specified for the parameter.

In the message text:

&1 is a macro name, &2 is a parameter number.

User response

Specify arguments for all macro parameters.

CCN3012	Operand of bitwise complement must be an integral type.

Explanation

The operand of the bitwise complement operator does not have an integral type. Valid integral types include: signed and unsigned char; signed and unsigned short, long, and int; and enum.

User response

Change the type of the operand, or use a different operand.

CCN3013	Operand of unary + or - operator must be an arithmetic type.

Explanation

The operand of the unary + or - operator does not have an arithmetic type. Valid arithmetic types include: signed and unsigned char; signed and unsigned short, long, and int; enum, float, double, and long double.

User response

Change the type of the operand, or use a different operand.

CCN3014	Operand of logical negation must be a scalar type.

Explanation

The operand of the logical negation operator (!) does not have a scalar type. Valid scalar types include: signed and unsigned char; signed and unsigned short, long, and int; enum, float, double, long double, and pointers.

User response

Change the type of the operand, or use a different operand.

CCN3017	Operand of address operator must be an lvalue or function designator.

Explanation

The operand of the address operator (unary &) is not valid. The operand must be either a function designator or an lvalue that designates an object that is not a bit field and is not declared with register storage class.

User response

Change the operand.

CCN3018	Operand of indirection operator must be a pointer expression.

Explanation

The operand of the indirection operator (unary *) is not a pointer.

User response

Change the operand to a pointer.

CCN3019	Expecting an array or a pointer to object type.

Explanation

Index operator ([]) operates only on arrays or pointer to objects.

User response

Change the operand.

CCN3020	Expression must be an integral type.

Explanation

The expression does not evaluate to an integral type. Valid integral types include: signed, unsigned and plain char, signed and unsigned short, int, long, and enum.

User response

Change the type of the operand.

CCN3021	Expecting struct or union.

Explanation

The left hand operand of the dot operator (.) must have a struct or union type.

User response

Change the operand.

CCN3022	"&1" is not a member of "&2".

Explanation

The specified member does not belong to the structure or union given. One of the following situations has occurred:

1. The right-hand operand of the dot (.) operator is not a member of the structure or union specified on the left-hand side of the operator.

2. The right hand operand of the arrow (->) operator is not a member of the structure or union pointed to by the pointer on the left hand side of the operator.

In the message text:

&1 is the name of a member, &2 is a type or type name.

User response

Change the identifier.

CCN3023	Expecting function or pointer to function.

Explanation

The expression is followed by an argument list but does not evaluate to a function designator.

User response

Change the expression to be a function or a pointer to a function.

CCN3024	The operand of the __alignof__ operator is not valid.

Explanation

The __alignof__ operator cannot be used with incomplete functions, incomplete types, or arrays of unknown size. The __alignof__ operator cannot be applied to an expression that has a function type or an incomplete type, or to the parenthesized name of such a type.

User response

Change the operand.

CCN3025	Operand must be a modifiable lvalue.

Explanation

A modifiable lvalue is an expression representing an object that can be changed.

User response

Change the operand.

CCN3026	Number of initializers cannot be greater than the number of aggregate members.

Explanation

Too many initializers were found in the initializer list for the indicated declaration.

User response

Check the number of initializers and change it to correspond to the number of declared members. Make sure the closing brace at the end of the initializer list is positioned correctly.

CCN3027	Function &1 cannot be initialized.

Explanation

An attempt was made to assign an initial value to a function identifier. You can not assign a value to a function identifier.

In the message text:

&1 is a function name.

User response

Remove the assignment operator and the initializer.

CCN3028	Storage class "&1" cannot be used with external data.

Explanation

The storage class is not appropriate for this declaration. Restrictions include: 1) Storage class specifier is not allowed for aggregate members, casts, sizeof or offsetof declarations. 2) Declarations at file scope cannot have a "register" or "auto" storage class.

In the message text:

&1 is a storage class specifier.

User response

Remove the storage class specifier.

CCN3029	The pragma is ignored. Identifiers are already disjoint.

Explanation

The identifiers that are specified in the pragma are already known to be disjoint so the pragma is ignored.

User response

Nothing, or remove the pragma as it is redundant.

CCN3030	Identifier &1 cannot be redeclared.

Explanation

The identifier has already been declared.

In the message text:

&1 is an identifier.

User response

Remove one of the declarations.

CCN3031	All dimensions except the first must be specified for a multidimensional array.

Explanation

Only the first dimension of an initialized array can be unspecified. All the other dimensions must be specified on the declaration.

User response

Specify all the other dimensions in the array declaration.

CCN3032	Elements of an array cannot be functions.

Explanation

An array must be composed of elements that are an object type. Functions are not object types and thus cannot be elements of an array.

User response

Use a pointer to the function, or change the type of the element.

CCN3033	Function &1 is not valid. Function cannot return a function.

Explanation

A function cannot have a return type of function.

In the message text:

&1 is a function name.

User response

Return a pointer to the function or specify a different return type.

CCN3034	Function &1 is not valid. Function cannot return an array.

Explanation

A function cannot return an array and the specified return type of the function is an array.

In the message text:

&1 is a function name.

User response

Return a pointer to the array or specify a different return type.

CCN3035	Storage class "&1" cannot be used with functions.

Explanation

A function can only have a storage class of extern or static.

In the message text:

&1 is a storage class specifier.

User response

Remove the storage class specifier for the function identifier, or change it to either extern or static.

CCN3036	Range error.

Explanation

The value is outside of the valid range.

User response

Change value to be within the required limits.

CCN3037	Member of struct or union cannot be a function.

Explanation

Members of structs or unions must have object type. Functions do not have object type and cannot be members of a struct or union.

User response

Use a pointer to the function or remove the function from the member list.

CCN3039	Expecting a parameter after # operator.

Explanation

The # preprocessor operator can only be applied to a macro parameter.

User response

Place a parameter after the # token, or remove the token.

CCN3041   **The invocation of macro &1 contains fewer arguments than are required by the macro definition.**

Explanation

The number of arguments supplied to the macro must match the number of parameters in the macro definition. There are not enough arguments supplied.

In the message text:

&1 is a macro name.

User response

Complete the specification of the macro argument list.

CCN3043   **The operand of the sizeof operator is not valid.**

Explanation

Sizeof operator cannot be used with functions, void types, bit fields, incomplete types, or arrays of unknown size. The sizeof operator cannot be applied to an expression that has a function type or an incomplete type, to the parenthesized name of such a type, or to an lvalue that designates a bit field object.

User response

Change the operand.

CCN3044   **Expression must be a non-negative integer constant.**

Explanation

The supplied expression must evaluate to a non-negative integer constant.

User response

Change the constant expression to yield a non-negative value.

CCN3045   **Undeclared identifier &1.**

Explanation

You must declare a C identifier before you use it in an expression.

In the message text:

&1 is an identifier.

User response

Declare an identifier with that name in the current scope or in a higher scope.

CCN3046   **Syntax error.**

Explanation

See the C/C++ Language Reference for a complete description of C syntax rules.

User response

Correct the syntax error and compile again.

CCN3047   **Incorrect hexadecimal escape sequence \x. \ ignored.**

Explanation

\x is used to indicate an hexadecimal escape sequence but the sequence immediately following is not a valid hexadecimal number.

User response

Change the sequence to a valid hexadecimal number.

CCN3048   **Unable to initialize source conversion from code page &1 to code page &2.**

Explanation

An error occurred when attempting to convert source between the specified code pages.

In the message text:

&1 and &2 are code page names.

User response

Ensure the code pages are correct and that conversion between these code pages is supported.

CCN3049   **The object &1 has a size &2 which exceeds the compiler limit &3.**

Explanation

The size of the object is too large for the compiler to represent internally.

In the message text:

&1 is name of the variable, &2 is the storage size of the variable, &3 is the maximum storage size allowed by the compiler.

User response

Reduce the size of the object.

CCN3050   **Return type "&1" in redeclaration is not compatible with the previous return type "&2".**

Explanation

The second declaration of the function declares a different return type from the first. The declaration must be identical. When you redeclare a function, the return type and parameter types must be the same in both declarations.

In the message text:

&1 and &2 are types.

User response

Change the declaration of one or both functions so that their return types are compatible.

CCN3051   **Case expression must be a valid integral constant.**

Explanation

The expression in the case statement must be a constant integral expression. Valid integral expressions are: char, signed and unsigned int, and enum.

User response

Change the expression.

CCN3052   **Duplicate case label for value &1. Labels must be unique.**

Explanation

Two case labels in the same switch statement cannot evaluate to the same integer value.

In the message text:

&1 is a case label value.

User response

Change one of the labels.

CCN3053   **Default label cannot be placed outside a switch statement.**

Explanation

A statement is labeled with default, which can only be used as a statement label within a switch statement.

User response

Remove the default case label, or place it inside a switch statement. Check for misplaced braces on a previous switch statement.

CCN3054   **Switch statement cannot contain more than one default label.**

Explanation

Only one default label is allowed within a switch statement. Nested switch statements may each have one default label. This error may have been caused by a default label that is not properly placed within a nested switch statement.

User response

Remove one of the default labels or check for misplaced braces on nested switch statements..

CCN3055   **Case label cannot be placed outside a switch statement.**

Explanation

Case labels are only allowed within a switch statement.

User response

Remove the case label, or place it within a switch statement group. Check for misplaced braces on the previous switch statement.

CCN3056   **Break statement cannot be placed outside a while, do, for, or switch statement.**

Explanation

Break statements are only allowed within a while, do, for, or switch statement.

User response

Remove the break statement or place it inside a while, do, for or switch statement. Check for misplaced braces on a previous statement.

CCN3057   **Continue cannot be placed outside a while, do, or for statement.**

Explanation

Continue is only valid as, or within, a loop body.

User response

Remove the continue statement or place it inside a while, do or for loop. Check for misplaced braces on a previous loop.

CCN3058   **Label &1 has already been defined on line &2 of "&3".**

Explanation

You already used the label to identify a section of code in the file indicated. You cannot redefine a label.

In the message text:

&1 is a label, &2 is a line number, &3 is a file name.

User response

Change the name of one of the labels.

CCN3059   **Comment that started on line &1 must end before the end of file.**

Explanation

A comment that was not terminated has been detected. The comment started on the line indicated.

In the message text:

&1 is a line number.

User response

End the comment before the file ends.

CCN3062   **Escape sequence &1 is out of the range 0-&2. Value is truncated.**

Explanation

Character constants specified in an escape sequence exceeded the decimal value of 255, or the octal equivalent of 377, or the hexadecimal equivalent of FF.

In the message text:

&1 is an escape sequence, &2 is an integer.

User response

Change the escape sequence so that the value does not exceed the maximum value.

CCN3067   **A struct or union can only be assigned to a compatible type.**

Explanation

The assignment is invalid between the given aggregate types.

User response

Change the operands so that they have the same type.

CCN3068   **Operation between types "&1" and "&2" is not allowed.**

Explanation

The operation specified is not valid between the operands having the given types.

In the message text:

&1 and &2 are both type names.

User response

Either change the operator or the operands.

CCN3070   **Register is the only storage class that can be used with parameters.**

Explanation

Parameters can have either no storage class specifier or the register storage class specifier.

User response

Remove the storage class specified in the parameter declaration or use the register storage class.

CCN3073   **Empty character constant.**

Explanation

An empty character constant is not valid. There must be at least one character between the single quotation marks.

User response

Put at least one character inside the pair of single quotation marks.

CCN3076   **Character constant &1 has more than 4 characters. No more than rightmost 4 characters are used.**

Explanation

A character constant can only have up to four bytes.

In the message text:

&1 is a character constant.

User response

Change the character constant to contain four bytes or less.

CCN3077	The wchar_t value &1 is not valid.

Explanation

The value is not a valid wchar_t value. See the C/C++ Language Reference for information on wide characters.

In the message text:

&1 is a wchar_t value.

User response

Change character to a valid wchar_t. See the C/C++ Language Reference for information about the wchar_t type.

CCN3078	#&1 directive has no effect.

Explanation

A preprocessor directive has been specified that has no effect.

In the message text:

&1 is a preprocessor directive.

User response

Remove the preprocessor directive.

CCN3085	Predefined macro &1 cannot be undefined.

Explanation

The macro is predefined. You cannot undefine predefined macros.

In the message text:

&1 is a macro name.

User response

Remove the statement that undefines the macro.

CCN3095	Unexpected parameter &1.

Explanation

A parameter was declared in the parameter declaration list of the K&R function definition. The parameter did not appear in the parameter identifier list. It is also possible that the K&R function definition had more parameters than the function prototype.

In the message text:

&1 is a parameter name.

User response

Change the number of parameters.

CCN3098	Missing argument(s).

Explanation

The function call contains fewer arguments than specified in the parameter list of the function prototype.

User response

Make sure the function call has the same number of arguments as the function prototype has parameters.

CCN3099	Unexpected argument.

Explanation

The function call contains more arguments than specified in the parameter list of the function prototype.

User response

Change the number of arguments in the function call or change the function prototype.

CCN3103	Tag &1 requires a complete definition before it is used.

Explanation

Only pointer declarations can include incomplete types. A struct or union tag is undefined if the list describing the name and type of its members has not been specified.

In the message text:

&1 is a struct or union tag.

User response

Define the tag before it is used in the declaration of an identifier or complete the declaration.

CCN3104	The value of an enumeration constant must be an integral constant expression.

Explanation

If an enum constant is initialized in the definition of an enum tag, the initial value of the constant must be an

integral expression that has a value representable as an int.

User response

Remove the initial value, or ensure that the initial value is an integral constant expression with a value representable as an int.

CCN3108	Bit fields with zero width must be unnamed bit fields.

Explanation

A named bit field must have a positive length; a zero length bit field is used for alignment only and must not be named.

User response

Redefine the bit field with a length greater than zero or remove the name of the bit field.

CCN3112	Duplicate type qualifier "&1" ignored.

Explanation

The indicated qualifier appears more than once in the type declaration.

In the message text:

&1 is a type qualifier.

User response

Remove one of the duplicate qualifiers.

CCN3115	Duplicate type specifier "&1" ignored.

Explanation

A duplicate type specifier appears in the type declaration.

In the message text:

&1 is a type specifier.

User response

Remove one of the duplicate type specifiers.

CCN3117	Operand must be a scalar type.

Explanation

Valid scalar types include: signed and unsigned char; signed and unsigned short, long, and int; enum, float, double, long double, and pointers.

User response

Change the type of the operand, or use a different operator.

CCN3119	Duplicate storage class specifier &1 ignored.

Explanation

A duplicate storage class specifier appears in the declaration.

In the message text:

&1 is a storage class specifier.

User response

Remove one of the duplicate storage class specifiers.

CCN3120	Function cannot return a &1 qualified type.

Explanation

The const or volatile qualifier cannot be used to qualify a function's return type.

In the message text:

&1 is a storage class specifier.

User response

Remove the qualifier or return a pointer to the qualified type.

CCN3122	Expecting pointer to struct or union.

Explanation

The left hand operand of the arrow operator (->) must have type pointer to structure or pointer to union.

User response

Change the operand.

CCN3127	The second and third operands of the conditional operator must have compatible struct or union types.

Explanation

If one operand in the conditional expression has type struct or union, the other operand must also have type struct or union.

User response

Make the operands compatible.

CCN3131   **Explicit dimension specification or initializer required for an auto or static array.**

Explanation

For arrays of automatic or static storage class, all dimensions of the array must be specified in the declaration. If the declaration provides an initialization, the first dimensions may be unspecified because the initialization will determine the size needed.

User response

Specify all of the dimensions in the array declaration.

CCN3134   **Array bound is too large.**

Explanation

The size of the array is too large for the compiler to represent internally.

User response

Reduce the size of the array.

CCN3137   **Declaration must declare at least one declarator, tag, or the members of an enumeration.**

Explanation

The declaration specifier was the only component of the declaration. eg. int ;

User response

Specify at least one declarator, tag, or member of an enumeration.

CCN3152   **A register array may only be used as the operand to sizeof.**

Explanation

The only operator that can be applied to an array declared with storage class specifier register is sizeof.

User response

Remove the operation or remove the register storage class specifier.

CCN3155   **Option &1 requires suboption(s).**

Explanation

The option is not completely specified; a suboption is required.

In the message text:

&1 is an option name.

User response

Add a suboption.

CCN3159   **Bit field type specified for &1 is not valid. Type &2 assumed.**

Explanation

The type of a bit field must be a (possibly qualified) version of int, signed int or unsigned int.

In the message text:

&1 is an identifier, &2 is a type specifier (signed or unsigned).

User response

Define the bit field with a type signed int or unsigned int.

CCN3160   **Object &1 cannot be declared as type void.**

Explanation

The type void can only be used as the return type or parameter list of a function, or with a pointer. No other object can be of type void.

In the message text:

&1 is an identifier name.

User response

Ensure that the declaration uses type void correctly.

CCN3162   **No definition was found for function &1. Storage class changed to extern.**

Explanation

A static function was declared and referenced in this file. The definition of the function was not found before the end of the file. When a function is declared to be static, the function definition must appear in the same file.

In the message text:

&1 is a function name.

User response

Change the storage class to extern or provide a function definition in this file.

CCN3164   **Expression must be a scalar type.**

Explanation

Valid scalar types include: signed and unsigned char; signed and unsigned short, long, and int; enum, float, double, long double, and pointers.

User response

Change the expression.

CCN3166   **Definition of function &1 requires parentheses.**

Explanation

The syntax of the declaration is not correct. The compiler assumes it is the declaration of a function in which the parentheses surrounding the parameters are missing.

In the message text:

&1 is a function name.

User response

Check the syntax of the declaration. Ensure the object name and type are properly specified. Check for incorrect spelling or missing parentheses.

CCN3167   **String literal is longer than target array. Literal is truncated on the right.**

Explanation

An attempt was made to initialize an array with a string that is too long. The largest possible prefix of the string has been placed in the array.

User response

Increase the size of the array. Make sure you include space for the terminating null character.

CCN3168   **Initializer must be enclosed in braces.**

Explanation

The initializer list for a declarator must be enclosed in braces.

User response

Check for misplaced or missing braces.

CCN3169   **Too many suboptions specified for option FLAG. Specify only two suboptions.**

Explanation

The FLAG option takes two suboptions separated by ':'. The suboptions indicate the level of errors to be reported in the source listing and in stderr.

User response

Only specify two suboptions to the FLAG option.

CCN3170   **Parameter &1 has already been defined on line &2 of "&3".**

Explanation

A parameter can only be defined once but more than one definition for the parameter has been specified. Parameters names must be unique.

In the message text:

&1 is a parameter name, &2 is a line number, &3 is a file name.

User response

Remove one of the parameter declarations or change the name of the identifier.

CCN3172   **Parameter type list for function &1 contains parameters without identifiers.**

Explanation

In a C function definition, all parameters must be named in the parameter list. The only exceptions are parameters of type void.

In the message text:

&1 is a function name.

User response

Name the parameter or remove it.

CCN3173   **Option &1 is not recognized.**

Explanation

An invalid option was specified.

In the message text:

&1 is an option name.

User response

Correct the spelling of the option.

CCN3174	Option &1 must be specified on the command line.

Explanation

The option can only be specified on the command line and is not valid as part of an options pragma.

In the message text:

&1 is an option name.

User response

Specify option on command line.

CCN3175	Option &1 must be specified on the command line or before the first C statement in the program.

Explanation

The option is specified in a pragma options after the first C token in the compilation unit. It must be specified before the first token.

In the message text:

&1 is an option name.

User response

Specify the option on the command line or move the pragma options before the first token.

CCN3176	Option &1 cannot take more than one suboption.

Explanation

More than one suboption was specified for an option that can only accept one suboption.

In the message text:

&1 is an option name.

User response

Remove the extra suboptions.

CCN3178	Unexpected argument for built-in function &1.

Explanation

The function call contains more arguments than specified in the parameter list of the built-in function.

In the message text:

&1 is a function name.

User response

Change the number of arguments in the function call.

CCN3180	Redeclaration of built-in function &1 ignored.

Explanation

Built-in functions are declared by the compiler and cannot be redeclared.

In the message text:

&1 is a function name.

User response

Remove the declaration.

CCN3181	Definition of built-in function &1 ignored.

Explanation

Built-in functions are defined by the compiler and cannot be redefined.

In the message text:

&1 is a function name.

User response

Remove the function definition.

CCN3182	Arguments missing for built-in function &1.

Explanation

The function call contains fewer arguments than specified in the parameter list of the built-in function.

In the message text:

&1 is a function name.

User response

Change the number of arguments in the function call.

CCN3183	Built-In function &1 cannot change a read-only string literal.

Explanation

Read-only strings cannot be modified.

In the message text:

&1 is a function name.

User response

Modify a copy of the string or change the string's read-only status.

CCN3184	Too few suboptions specified for option FLAG. Specify two suboptions.

Explanation

The FLAG option takes two suboptions separated by ':'. The suboptions indicate the level of errors to be reported in the source listing and in stderr.

User response

Specify two suboptions to the FLAG option.

CCN3185	#line number &1 must be greater than zero.

Explanation

The #line directive tells the compiler to treat the following source lines as starting from the specified line. This number must be a non-negative offset from the beginning of the file.

In the message text:

&1 is an integer.

User response

Change the line number to a non-negative integer.

CCN3186	String literal must be ended before the end of line.

Explanation

String literals must end before the end of the line. To create a string literal longer than one line, use the line continuation sequence (a backslash (\) at the end of the line), or concatenate adjacent string literal.

User response

End the string with a quotation mark before the end of the line or use the continuation sequence.

CCN3188	Reserved name &1 cannot be defined as a macro name.

Explanation

The name is reserved for the compiler's use.

In the message text:

&1 is a reserved name.

User response

Choose another name.

CCN3189	Floating-point constant &1 is not valid.

Explanation

See the C/C++ Language Reference for a description of a floating-point constant.

In the message text:

&1 is the floating-point literal.

User response

Ensure that the floating-point constant does not contain any characters that are not valid.

CCN3190	Automatic constant &1 does not have a value. Zero is being assumed.

Explanation

Const qualified variable declarations should contain an initializer. Otherwise you cannot assign the variable a value.

In the message text:

&1 is a variable name.

User response

Initialize the const variable when you declare it.

CCN3191	The character &1 is not a valid C source character.

Explanation

Refer to the C/C++ Language Reference for information on valid characters.

In the message text:

&1 is a character.

User response

Change the character.

CCN3192   **Cannot take address of built-in function &1.**

Explanation

You cannot take the address of a built-in function or declare a pointer to a built-in function.

In the message text:

&1 is a function name.

User response

Remove the operation that takes the address of the built-in function.

CCN3193   **The size of this type is zero.**

Explanation

You cannot take the address of an array of size zero.

User response

Remove the operation that takes the address of the zero-sized array.

CCN3194   **Incomplete type is not allowed.**

Explanation

Except for pointers, you cannot declare an object of incomplete type.

User response

Complete the type declaration.

CCN3195   **Integral constant expression with a value greater than zero is required.**

Explanation

The size of an array must be an expression that evaluates to a compile-time integer constant that is larger than zero.

User response

Change the expression.

CCN3196   **Initialization between types "&1" and "&2" is not allowed.**

Explanation

An attempt was made to initialize a variable with an incompatible type.

In the message text:

&1 and &2 are both type names.

User response

Ensure types are compatible.

CCN3197   **Expecting header file name in #include directive.**

Explanation

There was no header file name after the #include directive.

User response

Specify the header file name. Enclose system header names in angle brackets and user header names in double quotation marks.

CCN3198   **#if, #else, #elif, #ifdef, #ifndef block must be ended with #endif.**

Explanation

Every #if, #ifdef, and #ifndef must have a corresponding #endif.

User response

End the conditional preprocessor statements with a #endif.

CCN3199   **#&1 directive requires a macro name.**

Explanation

There must be a macro name after every #define, #undef, #ifdef or #ifndef.

In the message text:

&1 is a preprocessor directive.

User response

Ensure that a macro name follows the #define, #undef, #ifdef, or #ifndef preprocessor directive.

CCN3200   **#elif can only appear within a #if, #elif, #ifdef, or #ifndef block.**

Explanation

#elif is only valid within a conditional preprocessor block.

User response

Remove the #elif statement, or place it within a conditional preprocessor block.

CCN3201	#else can only appear within a #if, #elif, #ifdef or #ifndef block.

Explanation

#else is only valid within a conditional preprocessor block.

User response

Remove the #else statement, or place it within a conditional preprocessor block.

CCN3202	#endif can only appear at the end of a #if, #elif, #ifdef or #ifndef block.

Explanation

Every #endif must have a corresponding #if, #ifdef, or #ifndef.

User response

Remove the #endif statement, or place it after a conditional preprocessor block.

CCN3204	Unexpected end of file.

Explanation

The end of the source file has been encountered prematurely.

User response

Check for misplaced braces.

CCN3205	&1

Explanation

The #error directive was encountered. Compilation terminated.

In the message text:

&1 is text following the #error directive.

User response

Recompile with correct macro definitions.

CCN3206	Suffix of integer constant &1 is not valid.

Explanation

Valid integer suffixes are u or U for unsigned, or l or L for long. Unsuffixed constants are given the smallest data type that can hold the value. Refer to the C/C++ Language Reference.

In the message text:

&1 is an integer constant.

User response

Change or remove the suffix.

CCN3207	Integer constant &1 out of range.

Explanation

The specified constant is too large to be represented by an unsigned long int.

In the message text:

&1 is an integer constant.

User response

The constant integer must have a value less than UINT_MAX defined in <limits.h>.

CCN3208	Compilation ended due to an I/O error.

Explanation

A file read or write error occurred.

User response

Ensure that you have read access to all source files, and read and write access to the TMP directory. You also need write access to the object output directory.

CCN3209	Character constants must end before the end of a line.

Explanation

Character literals must be terminated before the end of the line.

User response

End the character literal before the end of the line. Check for misplaced quotation marks.

CCN3210	The ## operator requires two operands.

Explanation

The ## operator must be preceded and followed by valid tokens in the macro replacement list. Refer to the C/C++ Language Reference for information on the ## operator.

User response

Provide both operands for the ## operator.

CCN3211	Parameter list must be empty, or consist of one or more identifiers separated by commas.

Explanation

The macro parameter list must be empty, contain a single identifier, or contain a list of identifiers separated by commas.

User response

Correct the parameter list.

CCN3212	Duplicate parameter &2 in definition of macro &1.

Explanation

The identifiers in the macro parameter list must be unique.

In the message text:

&1 is a macro name, &2 is a parameter name.

User response

Change the identifier name in the parameter list.

CCN3213	Macro name &1 cannot be redefined.

Explanation

You can define a macro multiple times only if the definitions are identical except for white space separating the tokens.

In the message text:

&1 is a macro name.

User response

Change the macro definition to be identical to the preceding one, or remove it.

CCN3215	Too many arguments specified for macro &1.

Explanation

The number of arguments specified in the macro invocation is different from the number of parameters specified in the macro definition.

In the message text:

&1 is a macro name.

User response

Make the number of arguments consistent with the macro definition.

CCN3218	Unknown preprocessing directive #&1.

Explanation

An unrecognized preprocessing directive has been encountered.

In the message text:

&1 is a preprocessor directive.

User response

Check the spelling and syntax or remove the directive.

CCN3219	The #line value &1 is outside the range 1 to &2.

Explanation

The value for a #line directive must not exceed &2.

In the message text:

&1 and &2 are integers.

User response

Ensure that the #line value does not exceed &2.

CCN3220	#line value &1 must contain only decimal digits.

Explanation

A nonnumerical character was encountered in the #line value.

In the message text:

&1 is an integer.

User response

Check the syntax of the value given.

CCN3221	Initializer must be a valid constant expression.

Explanation

The initializers for objects of static storage duration, for elements of an array, or for members of a structure or union must be valid constant expressions.

User response

Remove the initialization or change the indicated initializer to a valid constant expression.

CCN3224   **Incorrect pragma ignored.**

Explanation

An unrecognized pragma directive was encountered. See the C/C++ Language Reference for the list of valid pragma directives.

User response

Change or remove the pragma directive.

CCN3226   **The ":" operator is not allowed between "&1" and "&2".**

Explanation

The operands must be of compatible type.

In the message text:

&1 and &2 are type names.

User response

Change the type of the operands.

CCN3229   **File is empty.**

Explanation

The source file contains no code.

User response

Check that the file name and path are correct. Add source code to the file.

CCN3231   **Error occurred while opening preprocessor output file.**

Explanation

The preprocessor was unsuccessful in attempting to open the output file.

User response

Ensure you have write access to the file.

CCN3232   **Divisor for modulus or division operator cannot be zero.**

Explanation

The value of the divisor expression cannot be zero.

User response

Change the expression used as the divisor.

CCN3234   **Expecting a new-line character on #&1 directive.**

Explanation

A character sequence was encountered when the preprocessor required a new-line character.

In the message text:

&1 is a preprocessor directive.

User response

Add a new-line character.

CCN3235   **Incorrect escape sequence &1. \ ignored.**

Explanation

An escape sequence that is not valid has been encountered in a string literal or a character literal. It is replaced by the character following the backslash (\).

In the message text:

&1 is an escape sequence.

User response

Change or remove the escape sequence.

CCN3236   **Macro name &1 has been redefined.**

Explanation

An attempt is being made to redefine the macro.

In the message text:

&1 is a macro name.

User response

Change the name of the macro being defined.

CCN3238   **Function argument cannot be type void.**

Explanation

The void type cannot appear in the argument list of a function call. The void type can appear in a parameter list only if it is a non-variable argument function. It is the only parameter in the list, and it is unnamed.

User response

Correct the argument or remove the argument.

CCN3242	An object with external linkage declared at block scope cannot be initialized.

Explanation

You cannot declare a variable at block scope with the storage class extern and give it an explicit initializer.

User response

Initialize the external object in the external declaration.

CCN3243	Value of enumeration constant must be in the range of &1.

Explanation

If an enum constant is initialized in the definition of an enum tag, the initial value must be a constant expression with a representable value of type specified in the message.

In the message text:

&1 is a type name.

User response

Remove the initial value, or ensure that it is a constant expression with a representable value of type specified in the message.

CCN3244	External variable &1 cannot be redefined.

Explanation

An attempt was made to redefine an external variable.

In the message text:

&1 is an identifier.

User response

Remove the redefinition.

CCN3245	Incompatible sign adjective "&1".

Explanation

Adjectives "signed" and "unsigned" can only modify integer type specifiers.

In the message text:

&1 is a type specifier.

User response

Either remove the sign adjective or use a different type specifier.

CCN3246	Incompatible length adjective "&1".

Explanation

Length adjectives short and long can only be applied to particular scalar types. See the C/C++ Language Reference for valid types.

In the message text:

&1 is a type specifier.

User response

Either remove the length adjective or use a different type specifier.

CCN3247	Incompatible type specifier "&1".

Explanation

The type specifier is not compatible with the type adjectives used. See the C/C++ Language Reference for valid combinations of type specifiers and adjectives.

In the message text:

&1 is a type specifier.

User response

Either remove the adjective or use a different type specifier.

CCN3248	More than one storage class specifier &1.

Explanation

A C declaration must only have one storage class specifier.

In the message text:

&1 is a storage class specifier.

User response

Ensure only one storage class is specified.

CCN3249	Identifier contains a $ character.

Explanation

You cannot use the $ character in an identifier. An identifier can contain alphanumeric characters and underscores. An identifier must start with either an underscore or alphabetic character.

User response

Remove the $ character.

CCN3250	Floating-point constant &1 out of range.

Explanation

The compiler detected a floating-point overflow either in scanning a floating-point constant, or in performing constant arithmetic folding.

In the message text:

&1 is a floating-point constant

User response

Change the floating-point constant so that it does not exceed the maximum value.

CCN3251	Static function &1 is undefined.

Explanation

A static function was declared and referenced in this file. The definition of the function was not found before the end of the file. When a function is declared to be static, the function definition must appear in the same file.

In the message text:

&1 is a function name.

User response

Define the function in the file or remove the static storage class.

CCN3255	pragma &1 is out of sequence.

Explanation

The pragma directive was out of sequence. See the C/C++ Language Reference for the restrictions on placement.

In the message text:

&1 is a pragma name.

User response

Change or remove the pragma directive.

CCN3258	Hexadecimal integer constant &1 is not valid.

Explanation

An invalid hexadecimal integer constant was specified. See the C/C++ Language Reference for details on specifying hexadecimal characters.

In the message text:

&1 is a hexadecimal integer constant.

User response

Change the value to a valid hexadecimal integer constant.

CCN3260	Octal integer constant &1 is not valid.

Explanation

An invalid octal integer constant was specified. See the C/C++ Language Reference for details on specifying octal characters.

In the message text:

&1 is an octal integer constant.

User response

Change the value to a valid octal integer constant.

CCN3261	Suboption &1 is not valid for option &2.

Explanation

An invalid suboption was specified for some option.

In the message text:

&1 is a suboption, &2 is an option name.

User response

Change the suboption.

CCN3262	pragma &1 must occur before first C statement in program. The pragma is ignored.

Explanation

This pragma must be specified before the first C token in the input (including header files).

In the message text:

&1 is a pragma name.

User response

Place the pragma directive in the file before any C code, or remove it.

CCN3263   **pragma strings directive can be specified only once per source file. pragma ignored.**

Explanation

This pragma specifies whether string literals are placed in read-only memory. It must appear only once and before any C code.

User response

Change the location of the directive and ensure that it appears only once in the translation unit.

CCN3264   **pragma &1 directive can be specified only once per source file.**

Explanation

There can only be one pragma &1 per source file.

In the message text:

&1 is a pragma name.

User response

Ensure that it occurs only once in the translation unit.

CCN3266   **Parameter(s) for pragma are out of range.**

Explanation

The pragma parameters were invalid. See the C/C++ Language Reference for details on valid pragma parameters.

User response

Change the parameter.

CCN3267   **Unrecognized pragma ignored.**

Explanation

An invalid pragma was encountered and ignored.

User response

Ensure that the pragma name is spelled correctly. A pragma with equivalent function, but a different name may exist. See the C/C++ Language Reference for a list of pragma directives.

CCN3268   **Macro &1 invoked with an incomplete argument for parameter &2.**

Explanation

The parameter for the macro invocation must have a complete argument.

In the message text:

&1 is a macro name, &2 is a parameter name.

User response

Complete the specification of the macro argument list. Check for missing commas.

CCN3271   **The indirection operator cannot be applied to a void pointer.**

Explanation

The indirection operator requires a pointer to a complete type. A void pointer is an incomplete type that can never be completed.

User response

Cast the pointer to a type other than void before this operation.

CCN3272   **Identifier not allowed in cast or sizeof declarations.**

Explanation

Only abstract declarators can appear in cast or sizeof expressions.

User response

Remove the identifier from the cast or sizeof expression and replace it with an abstract declarator.

CCN3273   **Missing type in declaration of &1.**

Explanation

A declaration was made without a type specifier.

In the message text:

&1 is an identifier.

User response

Insert a type specifier into the declaration.

CCN3274   **Missing declarator in member declaration.**

Explanation

An aggregate member declaration must specify a name. A type cannot be followed by a semicolon.

User response

Declare the member with a name.

CCN3275   **Unexpected text &1 encountered.**

Explanation

A syntax error has occurred. This message lists the tokens that were discarded by the parser when it tried to recover from the syntax error.

In the message text:

&1 is a token.

User response

Correct the syntax error and compile again.

CCN3276   **Syntax error: possible missing &1?**

Explanation

A syntax error has occurred. This message lists the token that the parser expected and did not find.

In the message text:

&1 is a token.

User response

Correct the syntax error and compile again.

CCN3277   **Syntax error: possible missing &1 or &2?**

Explanation

A syntax error has occurred. This message lists the tokens that the parser expected and did not find.

In the message text:

&1 and &2 are tokens.

User response

Correct the syntax error and compile again.

CCN3278   **The structure definition must specify a member list.**

Explanation

The declaration of a struct or a union that includes an empty member list enclosed between braces is not a valid struct or union definition.

User response

Specify the members of the struct or union in the definition or remove the empty braces to make it a simple struct or union tag declaration.

CCN3279   **A function declarator cannot have a parameter identifier list if it is not a function definition.**

Explanation

A function declarator that is not also a function definition may not have a K&R style parameter identifier list. An example is the "x,y" in "int (*fred(a,b)) (x,y) {}".

User response

Remove the parameter identifier list.

CCN3280   **Function argument assignment between types "&1" and "&2" is not allowed.**

Explanation

The type of the argument in the function call should match the corresponding parameter type in the function declaration.

In the message text:

&1 and &2 are types.

User response

Cast the argument to a different type, change the type or change the function prototype.

CCN3281   **Prefix and postfix increment and decrement operators cannot be applied to "&1".**

Explanation

Increment and decrement operators cannot operate on pointers to function or pointers to void.

In the message text:

&1 is a type.

User response

Change the pointer to point to an object type.

CCN3282	The type of the parameters must be specified in a prototype.

Explanation

A prototype specifies the number and the type of the parameters that a function requires. A prototype that does not specify the type of the parameters is not correct, for example,

```
fred(a,b);
```

User response

Specify the type of the parameters in the function prototype.

CCN3283	Functions cannot be declared &1 at block scope, &2 is ignored.

Explanation

Functions declared at block scope can only have extern as an explicit storage class specifier and cannot be inline.

In the message text:

&1 and &2 is a storage class specifier or the inline specifier.

User response

Place the declaration of the function at file scope, or remove the storage class specifier or the inline specifier.

CCN3285	The indirection operator cannot be applied to a pointer to an incomplete struct or union.

Explanation

A structure or union type is completed when the definition of its tag is specified. A struct or union tag is defined when the list describing the name and type of its members is specified.

User response

Complete the struct or union definition.

CCN3286	A struct or union with no named members cannot be explicitly initialized.

Explanation

Only aggregates containing named members can be explicitly initialized.

User response

Name the members of the struct or union.

CCN3287	The parameter list on the definition of macro &1 is not complete.

Explanation

There is a problem with the parameter list in the definition of the macro.

In the message text:

&1 is a macro name.

User response

Complete the parameter list. Look for misplaced or extra commas.

CCN3288	Expecting file name or new-line character on #line directive.

Explanation

The #line directive requires a line number argument as its first parameter and a file name as an optional second parameter. No other arguments are allowed. A new-line character must be present after the argument list.

User response

Change the directive syntax.

CCN3289	Macro &1 redefined with identical definition.

Explanation

Identical macro redefinitions are allowed but not necessary. The amount of white space separating the tokens has no bearing on whether macros are considered identical.

In the message text:

&1 is a macro name.

User response

Remove the redefinition.

CCN3290	Unknown macro name &1 on #undef directive.

Explanation

An attempt is being made to undefine a macro that has not been previously defined.

In the message text:

&1 is a macro name.

User response

Check the spelling of the macro name or remove the #undef directive.

CCN3291	Expecting decimal constant on #line directive.

Explanation

The value for a #line directive must be a decimal constant.

User response

Specify a line number on the #line directive.

CCN3292	Multibyte character literal not allowed on #&1 directive.

Explanation

The directive does not allow a multibyte character literal.

In the message text:

&1 is a preprocessor directive.

User response

Remove the multibyte character literal.

CCN3293	Identifier &1 assigned default value of zero on &2 directive.

Explanation

The indicated identifier in an #if or #elif expression was assigned the default value of zero. The identifier may have been intended to be expanded as a macro.

In the message text:

&1 is an identifier, &2 is a preprocessor directive.

User response

Add a #define for the macro before using it in a preprocessor conditional.

CCN3294	Syntax error in expression on #&1 directive.

Explanation

The expression for a preprocessor directive contains a syntax error.

In the message text:

&1 is a preprocessor directive.

User response

Replace the expression that controls the directive with a constant integral expression.

CCN3295	File ended with a continuation sequence.

Explanation

The file ended unexpectedly with a backslash character followed by a new-line character.

User response

Remove the continuation character from the last line of the file, or add code after the continuation character.

CCN3296	#include file &1 not found.

Explanation

The file specified on the #include directive could not be found. See the C/C++ Language Reference for file search order.

In the message text:

&1 is a file name.

User response

Ensure the #include file name and the search path are correct.

CCN3297	Unable to open input file &1. (&2)

Explanation

The compiler was unable to open the input file.

In the message text:

&1 is a file name, &2 is an additional system error message.

User response

Ensure the file exists and that the compiler can access it.

CCN3298	Unable to read input file &1. (&2)

Explanation

The compiler was unable to read the input file.

In the message text:

&1 is a file name, &2 is an additional system error message.

User response

Ensure the file exists and that the compiler can access it.

CCN3299	Maximum #include nesting depth of &1 has been exceeded.

Explanation

The included files have been nested too deeply.

In the message text:

&1 is an integer.

User response

Reduce the number of nested include files.

CCN3300	Insufficient storage available.

Explanation

The compiler ran out of memory trying to compile the file. This sometimes happens with large files or programs with large functions. Note that very large programs limit the amount of optimization that can be done.

User response

Increase your region size on MVS, or your virtual storage on VM. You can also divide the file into several small sections or shorten the function.

CCN3301	Redeclaration cannot specify fewer parameters than previous declaration.

Explanation

The function definition has fewer parameters than the prototype.

User response

Modify one of the function declarations so that the number and types of the parameters match.

CCN3302	The declarations of the function &1 must be consistent in their use of the ellipsis.

Explanation

The prototyped redeclaration of the function is not correct. Fewer parameters appear before the ellipsis in this function redeclaration than the previous declaration.

In the message text:

&1 is a function name.

User response

Ensure that the redeclaration is consistent with the previous declaration.

CCN3303	The type of the parameter &1 cannot conflict with the previous declaration of function &2.

Explanation

Nonprototype function declarations, popularly known as K&R prototypes, specify only the function return type. The function parentheses are empty; no information about the parameters is given.

Nonprototype function definitions specify a list of parameter names appearing between the function parentheses followed by a list of declarations (located between the parentheses and the opening left brace of the function) that indicates the type of the parameters. A nonprototype function definition is also known as a K&R function definition.

A prototype function declaration or definition specifies the type and the number of the parameters in the parameter declaration list that appears inside the function parentheses. A prototype function declaration is better known as an ANSI prototype, and a prototype function definition is better known as an ANSI function definition.

When the nonprototype function declarations/definitions are mixed with prototype declarations, the type of each prototype parameter must be compatible with the type that results from the application of the default argument promotions.

Most types are already compatible with their default argument promotions. The only ones that aren't are `char`, `short`, and `float`. Their promoted versions are, respectively, `int`, `int`, and `double`.

This message can occur in several situations. The most common is when mixing ANSI prototypes with K&R function definitions. If a function is defined using a K&R-style header, then its prototype, if present, must specify widened versions of the parameter types. Here is an example.

```
int fn(short); int fn(x) short x; {}
```

This is not valid because the function has a K&R-style definition and the prototype does not specify the widened version of the parameter. To be correct, the prototype should be

```
int fn(int);
```

because `int` is the widened version of `short`.

Another possible solution is to change the function definition to use ANSI syntax. This particular example would be changed to

```
int fn(short); int fn(short x) {}
```

This second solution is preferable, but either solution is equally valid.

In the message text:

&1 is a parameter name, &2 is a function name.

User response

Give a promoted type to the parameter in the prototype function declaration.

CCN3304   **No function prototype given for "&1".**

Explanation

A prototype declaration of the function specifying the number and type of the parameters was not found before the function was used. Errors may occur if the function call does not respect the function definition.

In the message text:

&1 is a function name.

User response

Add an appropriate function prototype before calling the function.

CCN3306   **Subscript operator requires an array operand in the offsetof macro.**

Explanation

A subscript was specified in the offsetof macro but the operand is not an array.

User response

Either change the operand to be an array type or remove the subscript operator.

CCN3307   **Array index must be a constant expression in the offsetof macro.**

Explanation

The offsetof macro is evaluated at compile time. Thus all arguments must be constant expressions.

User response

Change the expression.

CCN3308   **Operand of the offsetof macro must be a struct or a union.**

Explanation

The first operand of the offsetof macro must be a structure or union type.

User response

Change the operand.

CCN3309   **The offsetof macro cannot be used with an incomplete struct or union.**

Explanation

An incomplete struct or union is not a valid argument to the offsetof macro. A structure or union type is completed when the definition of its tag is specified.

User response

Ensure the struct or union is a complete type.

CCN3310   **The type "&1 &2" was introduced in a parameter list, and will go out of scope at the end of the function declaration or definition.**

Explanation

The tag will be added to parameter scope in ANSI mode. Thus it will go out of scope at the end of the declaration or function definition. In extended mode, the tag is added to the closest enclosing block scope.

In the message text:

&1 and &2 together form a type name.

User response

If the tag is needed for declarations outside its scope, move the tag declaration outside of parameter scope.

CCN3311   **Wide character constant &1 has more than one character. Last character is used.**

Explanation

All but the last character in the constant will be discarded.

In the message text:

&1 is a wide character constant.

User response

Remove all but one character or change the character constant into a string literal.

CCN3312	Compiler internal name &1 has been defined as a macro.

Explanation

Do not redefine internal compiler names.

In the message text:

&1 is a compiler internal name.

User response

Remove the macro definition or change the name of the macro being defined.

CCN3313	Compiler internal name &1 has been undefined as a macro.

Explanation

Do not redefine internal compiler names.

In the message text:

&1 is a compiler internal name.

User response

Remove the macro undefinition.

CCN3314	The tag of this expression's type has gone out of scope.

Explanation

The tag used in the type declaration of the object has gone out of scope, however the object is still referenced in the expression.

User response

Either remove the reference to the object or move the tag's definition to a scope that encloses both the referenced object and the object's declaration.

CCN3320	Operation is not allowed because the size of &1 is unknown.

Explanation

The operand must be a complete type for the compiler to determine its size.

In the message text:

&1 is a type.

User response

Provide a complete type definition.

CCN3321	You can specify an initializer only for the first named member of a union.

Explanation

There can only be an initializer for the first named member of a union.

User response

Remove all union initializers other than the one attached to the first named member.

CCN3322	Illegal multibyte character &1.

Explanation

The multibyte character specified is not valid.

In the message text:

&1 is a multibyte character.

User response

Correct the multibyte character.

CCN3323	"double" should be used instead of "long float".

Explanation

The type long float is not valid; it is treated as a double.

User response

Remove the long type specifier or use double instead of float.

CCN3324	"&1" cannot be converted to "&2".

Explanation

The cast between the two types is not allowed.

In the message text:

&1 is the type being converted from. &2 is the type being converted to.

User response

Remove the cast.

CCN3327	An error occurred while opening the listing file, &1.

Explanation

The compiler was unable to open the listing file.

In the message text:

&1 is a file name.

User response

Ensure the file exists and that the compiler can access it.

CCN3328	""&1" is not a valid hex digit."

Explanation

Valid hex digits are the letters A,B,C,D,E,F,0,1,2,3,4,5,6,7,8,9.

In the message text:

&1 is a character.

User response

Change the digit.

CCN3329	Byte string must have an even length.

Explanation

The byte string for a pragma mcfunc must be of even length.

User response

Ensure that the machine code string is of even length.

CCN3332	Option &1 is ignored because option &2 is not specified.

Explanation

The option &1 is only valid when used in conjunction with &2.

In the message text:

&1 and &2 are both option names.

User response

Compile with &2.

CCN3334	Identifier &1 has already been defined on line &2 of "&3".

Explanation

There is more than one definition of an identifier.

In the message text:

&1 is an identifier, &2 is a line number, &3 is a file name.

User response

Remove one of the definitions or change the name of the identifier.

CCN3335	Parameter identifier list contains multiple occurrences of &1.

Explanation

Identifier names in a parameter list must be unique.

In the message text:

&1 is a parameter name.

User response

Change the name of the identifier or remove the parameter.

CCN3339	A character string literal cannot be concatenated with a wide string literal.

Explanation

A string that has a prefix L cannot be concatenated with a string that is not prefixed. Concatenation requires that both strings be of the same type.

User response

Check the syntax of the value given.

CCN3341	#include header must be ended before the end of the line.

Explanation

A #include directive was specified across two or more lines.

User response

Ensure that the #include directive and its arguments are contained on a single line.

CCN3342   **""/*" detected in comment."**

Explanation

You can ignore this message if you intended "/*" to be part of the comment. If you intended it to start a new comment, move it out of the enclosing comment.

User response

Remove "/*" or ensure that "/*" was intended in the comment.

CCN3343   **Redeclaration of &1 differs from previous declaration on line &2 of "&3".**

Explanation

The redeclaration is not compatible with the previous declaration.

In the message text:

&1 is an identifier, &2 is a line number, and &3 is a file name.

User response

Either remove one declaration or make the types compatible.

CCN3344   **Member &1 has already been defined on line &2 of "&3".**

Explanation

Member names must be unique within the same aggregate.

In the message text:

&1 is an identifier, &2 is a line number, and &3 is a file name.

User response

Change the name.

CCN3345   **The data in precompiled header file &1 does not have the correct format.**

Explanation

The precompiled header file may have become corrupt and is ignored.

In the message text:

&1 is a file name.

User response

Regenerate the precompiled header files.

CCN3346   **Unable to open precompiled header file &1 for input. The original header will be used.**

Explanation

The compiler was unable to open the precompiled header file for reading and will use the original header.

In the message text:

&1 is a file name.

User response

Regenerate the precompiled header files.

CCN3347   **Precompiled header file &1 was created by a more recent release of the compiler. The original header will be used.**

Explanation

The compiler cannot understand the format of the precompiled header, since it was generated using a more recent version of the compiler. The original text version of the header will be used.

In the message text:

&1 is a file name.

User response

Regenerate the precompiled header files.

CCN3348   **Unable to write to precompiled header file &1.**

Explanation

The compiler was unable to write to the precompiled header files.

In the message text:

&1 is a file name.

User response

Ensure that the compiler has write access to the precompiled header files.

CCN3350   **Error writing to intermediate files. &1.**

Explanation

An error occurred during compilation. Ensure the compiler has write access to the work files and that there is enough free space.

In the message text:

&1 is an error message.

User response

Recompile compilation unit.

CCN3351   **Error opening intermediate files.**

Explanation

An error occurred during compilation. Ensure the compiler has write access to the work files and that there is enough free space.

User response

Recompile compilation unit.

CCN3352   **Incompatible specifications for options arch and tune.**

Explanation

The values specified for tune option cannot be smaller than that of arch.

User response

Change option values.

CCN3356   **Compilation unit is empty.**

Explanation

There is no code in the compilation unit.

User response

Ensure the correct source file is specified. Recompile.

CCN3357   **Unable to generate prototype for "&1" because one or more enum, struct, or union specifiers did not have a tag.**

Explanation

A prototype could not be generated for the function because the enum, struct or union declaration did not have a tag.

In the message text:

&1 is a function name.

User response

Specify a tag.

CCN3358   **"&1" is defined on line &2 of &3.**

Explanation

This message indicates where a previous definition is located.

In the message text:

&1 is an identifier name. &2 is a line number. &3 is a file name.

User response

Remove one of the definitions or change the name of the identifier.

CCN3359   **Automatic variable &1 contains a const member and is not initialized. It will be initialized to zero.**

Explanation

An automatic variable that has a const member is not initialized. The compiler is using zero as the initializer.

In the message text:

&1 is an identifier name.

User response

Initialize the const member.

CCN3360   **Same pragma &1 has already been specified for object "&2"; this specification is ignored.**

Explanation

The repetition of the pragma is redundant and is ignored.

In the message text:

&1 is the name of the pragma, &2 is an identifier name.

User response

Remove the duplicate pragma.

CCN3361   **A different pragma &1 has already been specified for object "&2", this specification is ignored.**

Explanation

A previous pragma for the object is taking precedence over this pragma.

In the message text:

&1 is the name of the pragma, &2 is an identifier name.

User response

Remove one of the pragma directives.

CCN3362	Identifier "&1" was referenced in pragma &2, but was never actually declared.

Explanation

A pragma refers to an identifier that has not been declared.

In the message text:

&1 is an identifier name, &2 is the name of the pragma.

User response

Declare the identifier or remove the pragma.

CCN3363	Packing boundary must be specified as one of 1, 2, 4, 8 or 16.

Explanation

Objects must be packed on 1, 2, 4, 8 or 16 byte boundaries.

User response

Change the packing specifier.

CCN3364	main must have C calling convention.

Explanation

An inappropriate linkage has been specified for the main function. This function is the starting point of the program so only C linkage is allowed.

User response

Change the calling convention of main.

CCN3366	Declaration cannot specify multiple calling convention specifiers.

Explanation

A declaration can specify only one calling convention. Valid calling conventions include: OS, COBOL, PLI, FORTRAN

User response

Remove extra calling convention specifiers.

CCN3367	Only functions or typedefs of functions can be given a calling convention.

Explanation

A calling convention protocol keyword has been applied to an identifier that is not a function type or a typedef to a function type.

User response

Check that correct identifier is specified or remove pragma.

CCN3369	The function cannot be redeclared with a different calling convention.

Explanation

The redeclaration of this function cannot have a different calling convention than the previous declaration. The function could have been given a calling convention through a typedef, or via a previous declaration.

User response

Make sure all declarations of the function specify the same calling convention.

CCN3374	Pointer types "&1" and "&2" are not compatible.

Explanation

The types pointed to by the two pointers are not compatible.

In the message text:

&1 and &2 are types.

User response

Change the types to be compatible.

CCN3376	Redeclaration of &1 has a different number of fixed parameters than the previous declaration.

Explanation

The number of fixed parameters in the redeclaration of the function does not match the original number of fixed parameters.

In the message text:

&1 is a function name.

User response

Change the declarations to have the same number of parameters, or rename or remove one of the declarations.

CCN3377	The type "&1" of parameter &2 differs from the previous type "&3".

Explanation

The type of the corresponding parameter in the previous function declaration is not compatible.

In the message text:

&1 is a type, &2 is a parameter name, &3 is a type.

User response

Change the parameter declaration or rename the function declaration.

CCN3378	Prototype for function &1 cannot contain "..." when mixed with a nonprototype declaration.

Explanation

A function prototype and a nonprototype declaration can not be compatible if one contains "...".

User response

Convert nonprototype declaration to a prototyped one or remove the "...".

CCN3379	Prototype for function &1 must contain only promoted types if prototype and nonprototype declarations are mixed.

Explanation

Nonprototype declarations have their parameters automatically promoted. Integral widening conversions are applied to integral types and float is converted into double.

In the message text:

&1 is a function name.

User response

Promote the parameter types in the prototyped declaration.

CCN3380	Parameter &1 has type "&2" which promotes to "&3".

Explanation

Nonprototype declarations have their parameters automatically promoted. Integral widening conversions are applied to integral types and float is converted into double.

In the message text:

&1 is a parameter name, &2 and &3 are types.

User response

Promote the parameter types in the prototyped declaration.

CCN3381	The type "&1" of parameter &2 in the prototype declaration is not compatible with the corresponding parameter type "&3" in the nonprototype declaration.

Explanation

The types of the parameters must be compatible.

In the message text:

&1 is a type, &2 is a parameter name, &3 is a type.

User response

Change the parameters so that they are compatible.

CCN3382	The type "&1" of identifier &2 differs from previous type "&3".

Explanation

The two types are not compatible.

In the message text:

&1 is a type, &2 is an identifier, &3 is a type.

User response

Change the parameter types so that they are compatible.

CCN3383	Expecting "&1" to be an external identifier.

Explanation

The identifier must have external linkage.

In the message text:

&1 is an identifier.

User response

Change the storage class to extern.

CCN3384	Expecting "&1" to be a function name.

Explanation

"&1" should be a function symbol.

In the message text:

&1 is an identifier.

User response

Specify a different name or change the type of the symbol.

CCN3387	The enum cannot be packed to the requested size. Change the enumeration value or change the pragma enum().

Explanation

Enums may be 1, 2, or 4 bytes in size.

User response

Change the enumeration value or change the pragma enum().

CCN3388	Value &1 specified in pragma &2 is out of range.

Explanation

Refer to the C/C++ Language Reference for more information about the valid values for the pragmas.

In the message text:

&1 is an integer, &2 is a pragma name.

User response

Specify a different value.

CCN3389	Some program text not scanned due to &1 option or pragma &2.

Explanation

MARGINS or SEQUENCE option, or pragma margins or sequence was used to limit the valid text region in a source file.

In the message text:

&1 is an option name, &2 is a pragma name.

User response

Remove the MARGINS or SEQUENCE option, or remove the pragma margins or sequence, or specify a more inclusive text region.

CCN3390	The function or variable &1 cannot be declared as an import in the same compilation unit in which it is defined.

Explanation

An object or function has both a definition and an import directive in this compilation unit. This creates a conflict, since the function or object can be defined either here or where it is exported from, but not in both places.

In the message text:

&1 is an identifier.

User response

Remove the pragma import directive or __import keyword or change the definition of the object or function into an extern declaration.

CCN3393	&1 value must contain only decimal digits.

Explanation

A nonnumerical character was encountered in the &1 value.

In the message text:

&1 is a pragma name.

User response

Check the syntax of the value given.

CCN3394	Ordinal value on pragma &1 is out of range.

Explanation

The specified ordinal number should be between 0 and 65535, inclusive.

In the message text:

&1 is a pragma name.

User response

Change the value accordingly

CCN3395	Variable &1 must be an external object or a function name for use with pragma import.

Explanation

The identifier specified by the pragma is not a function or external object.

In the message text:

&1 is an identifier.

User response

Declare the object with storage class "extern".

CCN3396	Option &1 is incompatible with option &2 and is ignored.

Explanation

The option is not compatible with another option so it is ignored.

In the message text:

&1 and &2 are option names.

User response

Remove one of the options.

CCN3397	Undefined function or variable &1 cannot have a pragma export.

Explanation

Only defined variables or functions can be specified as an export.

In the message text:

&1 is an identifier.

User response

Define the function or variable.

CCN3398	Bit field type specified for &1 is non-portable. The type should be signed int, unsigned int or int.

Explanation

The specification of the bit field type may cause problems with porting the code to another system.

In the message text:

&1 is an identifier.

User response

Change the type specifier.

CCN3399	The alignment of a structure/union is determined at the left brace of the definition.

Explanation

The alignment of an aggregate is constant throughout its definition.

User response

No response required.

CCN3400	pragma &1 must appear only once in any C file.

Explanation

The specified pragma can only be used once.

In the message text:

&1 is a pragma name.

User response

Remove all but one of the specified pragma directives.

CCN3401	Function &1 must be defined for pragma entry.

Explanation

The function must be defined for it to be specified using pragma entry.

In the message text:

&1 is a function name.

User response

Define the function.

CCN3402	&1 must be an externally-defined function for use with pragma entry.

Explanation

The identifier must be defined as a function with external linkage for it to be specified using pragma entry.

In the message text:

&1 is a function name.

User response

Define the function.

CCN3404	&1 can only qualify a pointer type.

Explanation

An attempt was made to qualify an identifier with &1 but the identifier is not a pointer.

In the message text:

&1 is a type qualifier.

User response

Remove the &1 type qualifier from the declaration or definition, or ensure it qualifies a pointer.

CCN3408	The linkage protocol is not supported on the target platform.

Explanation

An attempt to use an unsupported linkage protocol was made.

User response

Remove the linkage protocol keywords.

CCN3409	The static variable "&1" is defined but never referenced.

Explanation

A variable that is defined but never used probably serves no purpose.

In the message text:

&1 is an identifier.

User response

Remove the variable definition if you are not going to use the variable.

CCN3410	The automatic variable "&1" is defined but never referenced.

Explanation

A variable that is defined but never used likely serves no purpose.

In the message text:

&1 is an identifier.

User response

Remove the variable definition.

CCN3411	An array that is not an lvalue cannot be subscripted.

Explanation

A non-lvalue array is created when a function returns a structure that contains an array. This array cannot be dereferenced.

User response

Remove the subscript.

CCN3412	Referenced variable "&1", which was not initialized in its declaration.

Explanation

The variable referenced was not initialized in its declaration. At the point of the first reference, the variable might or might not have already been set to a value, depending on the code executed prior to the point of the first reference.

In the message text:

&1 is an identifier.

User response

This is an informational message to aid debugging. Either initialize the variable in its declaration, or trace the code carefully to make sure that it is set to a value prior to the first reference.

CCN3413	A goto statement is used.

Explanation

The use of goto statements may result in code that is more difficult to trace.

User response

Replace the goto statement with equivalent structured-programming constructs.

CCN3414   **The parameter "&1" is never referenced.**

Explanation

The parameter is passed to the function, but is not referenced anywhere within the function body.

In the message text:

&1 is a parameter name.

User response

Remove the parameter from the function prototype.

CCN3415   **The external function definition "&1" is never referenced.**

Explanation

A function that is defined but never used likely serves no purpose.

In the message text:

&1 is a function name.

User response

Remove the function definition, unless needed in another compilation unit.

CCN3416   **Taking the negative of the most negative value, '&1', of a signed type will cause truncation.**

Explanation

The negative of the most negative value cannot be represented as a positive value of the same type.

In the message text:

&1 is a numeric string.

User response

Change the value or use a larger data type.

CCN3417   **The function &1 is not defined but has pragma inline directive specified.**

Explanation

A pragma inline has been applied to an identifier which does not exist or does not correspond to a function.

In the message text:

&1 is a function name.

User response

Check that correct identifier is specified or remove the pragma.

CCN3418   **'&1' does not evaluate to a constant that fits in its signed type.**

Explanation

The expression evaluates to a number that is not within the range that can be stored by the target.

In the message text:

&1 is a numeric string.

User response

Change the expression so it evaluates to a value in the valid range.

CCN3419   **Converting &1 to type "&2" does not preserve its value.**

Explanation

The user cast converts &1 to a type that cannot contain the value of the original type.

In the message text:

&1 is an numeric string, &2 is a type.

User response

Change the cast.

CCN3420   **An unsigned comparison is performed between an unsigned value and a negative constant.**

Explanation

Comparing an unsigned value with a signed value may produce unexpected results.

User response

Type-cast the unsigned value to a signed type if a signed comparison is the comparison that you want, or type-cast the negative constant to an unsigned type if an unsigned comparison is the comparison that you want.

CCN3421   **The comparison is always true.**

Explanation

The type specifiers of the values being compared result in a constant result.

User response

Simplify or remove the conditional expression.

CCN3422   **The comparison is always false.**

Explanation

The type specifiers of the values being compared result in a constant result.

User response

Simplify or remove the conditional expression.

CCN3423   **The comparison may be rewritten as '&1'.**

Explanation

The type specifiers of the values being compared may allow the expression to be simplified.

In the message text:

&1 is a comparison expression.

User response

Simplify the comparison expression.

CCN3424   **The condition is always true.**

Explanation

Because the value of the conditional expression is constant, it may be possible to simplify or remove the conditional test.

User response

Change the conditional expression or remove the conditional test.

CCN3425   **The condition is always false.**

Explanation

Because the value of the conditional expression is constant, it may be possible to simplify or remove the conditional test.

User response

Change the conditional expression or remove the conditional test.

CCN3426   **An assignment expression is used as a condition. An equality comparison (==) may have been intended.**

Explanation

A single equal sign '=' is often mistakenly used as an equality comparison operator.

User response

Ensure an assignment operation was intended.

CCN3427   **A constant expression is used as a switch condition.**

Explanation

The same code path will be taken through every execution of the switch statement.

User response

Change the switch expression to be a non-constant value or remove the unused portions of the switch structure.

CCN3428   **The left-hand side of a shift expression is an unparenthesized arithmetic expression which has a higher precedence.**

Explanation

The left-hand expression is evaluated before the shift operator.

User response

Place parentheses around the left-hand expression to make the order of operations explicit.

CCN3429   **The right-hand side of a shift expression is an unparenthesized arithmetic expression which has a higher precedence.**

Explanation

The right-hand expression is evaluated before the shift operator.

User response

Place parentheses around the right-hand expression to make the order of operations explicit.

CCN3430   **The result of a comparison is either 0 or 1, and may not be appropriate as operand for another comparison operation.**

Explanation

The comparison expression may be malformed.

User response

Ensure that the resulting value from the comparison is appropriate for use in the following comparison.

CCN3431	The left-hand side of a bitwise &, |, or ^ expression is an unparenthesized relational, shift, or arithmetic expression which has a higher precedence.

Explanation

The left-hand expression is evaluated before the bitwise operator.

User response

Place parentheses around the left-hand expression to make the order of operations explicit.

CCN3432	The right-hand side of a bitwise &, |, or ^ expression is an unparenthesized relational, shift, or arithmetic expression which has a higher precedence.

Explanation

The right-hand expression is evaluated before the bitwise operator.

User response

Place parentheses around the right-hand expression to make the order of operations explicit.

CCN3433	The right-hand side of a bitwise shift expression should be positive and less than the width in bits of the promoted left operand.

Explanation

This expression may not be portable.

User response

Change the shift expression.

CCN3434	The left-hand side of a bitwise right shift expression has a signed promoted type.

Explanation

This expression may not be portable.

User response

Change the shift expression.

CCN3435	An expression statement should have some side effects because its value is discarded.

Explanation

If an expression statement has no side effects, then it may be possible to remove the statement with no change in program behavior.

User response

Change or remove the expression statement.

CCN3436	Left-hand side of comma expression should have side effects because its value is discarded.

Explanation

A comma expression evaluates to its right-hand operand.

User response

Change the expression.

CCN3437	The init or re-init expression of a for statement should have some side effects since its value is discarded.

Explanation

If the init and/or the re-init expression of a for statement have no side effects, the loop may not execute as intended.

User response

Change the init and/or re-init expressions.

CCN3438	The variable "&1" might be used before it is set.

Explanation

Because the variable has not been initialized, its value is undefined. The results of using an undefined variable are unpredictable.

In the message text:

&1 is an identifier.

User response

Add an initialization statement or change the expression.

CCN3439	Assigning enum type "&1" to enum type "&2" may not be correct.

Explanation

The values of the enumerated types may be incompatible.

In the message text:

&1 and &2 are enumerated type names.

User response

Change the types of the values being assigned.

CCN3440	Cannot assign an invalid enumerator value to enum type "&1".

Explanation

The value being assigned is not a member of the enumeration.

In the message text:

&1 is an enumerated type name.

User response

Change the value being assigned, or make it an enumeration member.

CCN3441	The macro definition will override the keyword "&1".

Explanation

Overriding a C keyword with a preprocessor macro may cause unexpected results.

In the message text:

&1 is a keyword.

User response

Change the name of the macro or remove it.

CCN3442	A trigraph sequence occurs in a character literal.

Explanation

The trigraph sequence will be converted. A literal interpretation may have been intended.

User response

Change the value of the character literal.

CCN3443	A trigraph sequence occurs in a string literal.

Explanation

The trigraph sequence will be converted. A literal interpretation may have been intended.

User response

Change the value of the string literal.

CCN3444	The opening brace is redundant.

Explanation

The initialization expression contains extra, possibly unnecessary, braces.

User response

Remove the extra braces.

CCN3445	The closing brace is redundant.

Explanation

The initialization expression contains extra, possibly unnecessary, braces.

User response

Remove the extra braces.

CCN3446	Array element(s) [&1] will be initialized with a default value of 0.

Explanation

Some array elements were not explicitly initialized. They will be assigned the default value.

In the message text:

&1 is an integer array element index, or a range of array element indices.

User response

Add initializations if necessary.

CCN3447   **The member(s) starting from "&1" will be initialized with a default value of 0.**

Explanation

Some members were not explicitly initialized. They will be assigned the default value.

In the message text:

&1 is an identifier.

User response

Add initializations if necessary.

CCN3448   **Assigning a packed struct to an unpacked struct, or vice versa, requires remapping.**

Explanation

Assignments between packed/unpacked structures may produce incorrect results.

User response

Change the type qualifiers of the values in the assignment.

CCN3449   **Missing return expression.**

Explanation

If a function has a non-void return type, then all return statements must have a return expression of the correct type.

User response

Add a return expression.

CCN3450   **Obsolete non-prototype-style function declaration.**

Explanation

The K&R-style function declaration is obsolete.

User response

Change the function declaration to the prototyped style.

CCN3451   **The target integral type cannot hold all possible values of the source integral type.**

Explanation

Data loss or truncation may occur because of the type conversions.

User response

Change the types of the values in the expression.

CCN3452   **Assigning a floating-point type to an integral type may result in truncation.**

Explanation

Data loss or truncation may occur because of the type conversions.

User response

Change the types of the values in the expression.

CCN3453   **Assigning a floating-point type to another floating-point type with less precision.**

Explanation

Data loss or truncation may occur because of the type conversions.

User response

Change the types of the values in the expression.

CCN3454   **&1 condition evaluates to &2.**

Explanation

This message traces preprocessor expression evaluation.

In the message text:

&1 is a condition, &2 is a value.

User response

No response required.

CCN3455   **defined(&1) evaluates to &2.**

Explanation

This message traces preprocessor #ifdef and #ifndef evaluation.

In the message text:

&1 is an identifier, &2 is a value.

User response

No response required.

CCN3456   Stop skipping tokens.

Explanation

This messages traces conditional compilation activity.

User response

No response required.

CCN3457   File &1 has already been included.

Explanation

This #include directive is redundant.

In the message text:

&1 is a file name.

User response

Remove the #include directive.

CCN3458   #line directive changing line to &1 and file to &2.

Explanation

This message traces #line directive evaluation.

In the message text:

&1 is a line number, &2 is a file name.

User response

No response required.

CCN3459   #line directive changing line to &1.

Explanation

This message traces #line directive evaluation.

In the message text:

&1 is a line number.

User response

No response required.

CCN3460   &1 nesting level is &2.

Explanation

This message traces conditional compilation activity.

In the message text:

&1 is a token, &2 is an integer.

User response

No response required.

CCN3461   Generating precompiled header file &1.

Explanation

This message traces precompiled header generation activity.

In the message text:

&1 is a file name.

User response

No response required.

CCN3462   Precompiled header file &1 is found but not used because it is not up to date.

Explanation

This message traces precompiled header file generation activity.

In the message text:

&1 is a file name.

User response

No response required.

CCN3463   Using precompiled header file &1.

Explanation

This message traces precompiled header file generation activity.

In the message text:

&1 is a file name.

User response

No response required.

CCN3464   Begin skipping tokens.

Explanation

This messages traces conditional compilation activity.

User response

No response required.

CCN3465 **#undef undefining macro name &1.**

Explanation

This message traces #undef preprocessor directive evaluation.

In the message text:

&1 is a macro name.

User response

No response required.

CCN3466 **Unary minus applied to an unsigned type.**

Explanation

The negation operator is inappropriate for unsigned types.

User response

Remove the operator or change the type of the operand.

CCN3467 **String literals concatenated.**

Explanation

Two string literals, each delimited by quotation marks, have been combined into a single literal.

User response

No response is necessary. This is an informational message.

CCN3468 **Macro name &1 on #define is also an identifier.**

Explanation

The name of the macro has already been used.

In the message text:

&1 is a macro name.

User response

Change the name of the macro.

CCN3469 **The static function "&1" is declared or defined but never referenced.**

Explanation

A function that is defined but never used serves no purpose.

In the message text:

&1 is a function name.

User response

Remove the function definition.

CCN3470 **Function "main" should return int, not void.**

Explanation

According to the ANSI/ISO standard, main should return int not void. Earlier standards (such as k&R) allowed a void return type for main.

User response

Change the return type of the function.

CCN3471 **Case label is not a member of enum type "&1"**

Explanation

Case labels must be members of the type of the switch expression.

In the message text:

&1 is an enumerated type name.

User response

Change the value of the case label.

CCN3472 **Statement may be unreachable.**

Explanation

The flow of execution may cause this statement to never be reached.

User response

Change the control flow in the program or remove the potentially unreachable statement.

CCN3473 **An unintended semi-colon may have created an empty loop body.**

Explanation

The loop body has no statements, and the conditional expression has no side effects.

User response

If this is what was intended, use "{}" instead of a semi-colon as empty loop body to avoid this message.

CCN3474   **Loop may be infinite.**

Explanation

The value of the conditional expression and/or the lack of exit points may result in an infinite loop.

User response

Adjust the conditional expression or add loop exit statements.

CCN3475   **The real constant arithmetic expression folds to positive infinity.**

Explanation

Constant folding results in an overflow.

User response

Change the expression.

CCN3476   **The real constant arithmetic expression folds to negative infinity.**

Explanation

Constant folding results in an overflow.

User response

Change the expression.

CCN3477   **The real constant arithmetic expression folds to a NaN.**

Explanation

Constant folding results in Not-a-Number (NaN).

User response

Change the expression.

CCN3478   **The then branch of conditional is an empty statement.**

Explanation

If the condition is true, then no statement is executed.

User response

Add a statement to be executed, or remove the conditional statement.

CCN3479   **Both branches of conditional statement are empty statements.**

Explanation

A conditional statement with empty branches is possibly degenerate.

User response

Add code to the conditional branches.

CCN3480   **Missing break statement allows fall-through to this case.**

Explanation

The preceding case did not end with a break, return, or goto statement, allowing the path of execution to fall-through to the code in this case.

User response

Add an appropriate terminating statement to the previous case, unless the fall-through was intentional.

CCN3481   **The end of the function may be reached without returning a value.**

Explanation

A return statement should be used to exit any function whose return type is non-void.

User response

Add a return statement, or change the function to return void.

CCN3482   **The opening brace before this point is redundant.**

Explanation

The initialization expression contains extra, possibly unnecessary, braces.

User response

Remove the extra braces.

CCN3483   **Switch statement contains no cases or only default case.**

Explanation

Code within a switch statement block that is not preceded by either "default" or "case" is never executed, and may be removed. Switch statements with neither "default" or "case" are probably incorrect.

User response

Change the switch statement to include cases.

CCN3484	External name &1 has been truncated to &2.

Explanation

The external name exceeds the maximum length and has been truncated. This may result in unexpected behavior if two different names become the same after truncation.

In the message text:

&1 and &2 are identifier names.

User response

Reduce the length of the external name.

CCN3485	Parameter declaration list is incompatible with declarator for &1.

Explanation

An attempt has been made to attach a parameter declaration list with a declarator which cannot have one.

User response

Change declarator or remove parameter declaration list.

CCN3486	A pointer to an incomplete type cannot be indexed.

Explanation

An index has been used with a pointer to an incomplete type.

User response

Declare the type that is pointed at or remove the index.

CCN3487	An argument cannot be an incomplete struct or union.

Explanation

An incomplete aggregate cannot be used as an argument to a function.

User response

Declare the type that is pointed at or use a pointer to the aggregate.

CCN3489	The incomplete struct or union tag &1 was not completed before going out of scope.

Explanation

A struct or union tag was declared inside a parameter list or a function body, but no member declaration list was provided.

In the message text:

&1 is a struct or union tag name.

User response

If the struct or union tag was declared inside a parameter list, provide a member declaration list at file scope. If the tag was declared inside a function body, provide a member declaration list within that function body.

CCN3490	The static variable "&1" is set but never referenced.

Explanation

A variable that is initialized but never used serves no purpose.

In the message text:

&1 is an identifier.

User response

Remove the variable definition if you do not intend to use it.

CCN3491	The automatic variable "&1" is set but never referenced.

Explanation

A variable that is initialized but never used likely serves no purpose.

In the message text:

&1 is an identifier.

User response

Remove the variable definition if you do not intend to use it.

CCN3492	**Redefinition of &1 hides previous definition.**

Explanation

The definition within the current scope hides a definition with the same name in an enclosing scope.

In the message text:

&1 is an identifier.

User response

Change the name to avoid redefining it.

CCN3493	**The external variable "&1" is defined but never referenced.**

Explanation

A variable that is defined but never used likely serves no purpose.

In the message text:

&1 is an identifier.

User response

Remove the variable definition, unless needed in another compilation unit.

CCN3494	**The external variable "&1" is set but never referenced.**

Explanation

A variable that is initialized but never used serves no purpose.

In the message text:

&1 is an identifier.

User response

Remove the variable definition, unless needed in another compilation unit.

CCN3495	**Pointer type conversion found.**

Explanation

An attempt is being made to convert a pointer of one type to a pointer of another type.

User response

Check the types of the values involved in the expression, and make them compatible.

CCN3496	**Parameter(s) for pragma &1 are of the wrong type.**

Explanation

The parameter for the pragma is incorrect and of the wrong type.

In the message text:

&1 is a pragma name.

User response

Find the correct type in the C/C++ Language Reference.

CCN3497	**Incomplete enum type not allowed.**

Explanation

An incomplete enum is being used where a complete enum type is required.

User response

Complete the type declaration.

CCN3498	**Member of struct or union cannot be incomplete type.**

Explanation

An incomplete aggregate is being used where a complete struct or union is required.

User response

Complete the type declaration.

CCN3499	**Function "main" should return int.**

Explanation

A return type other than int was specified for function main.

User response

Change the return type to int.

CCN3503	**The option "&1" is not supported.**

Explanation

The option specified is not supported on this platform.

In the message text:

&1 is an option name.

User response

Remove the option.

CCN3505   **Type "&1" of identifier "&2" was incomplete at the end of its scope.**

Explanation

An incomplete declaration was made of some identifier and it is still incomplete at the end of its scope.

In the message text:

&1 is a type name, &2 is an identifier.

User response

Complete the declaration.

CCN3508   **Option &1 for pragma &2 is not supported.**

Explanation

For a list of all valid options for pragma directives, see the C/C++ Language Reference.

In the message text:

&1 is an option name, &2 is a pragma name.

User response

Ensure the pragma syntax and options are correct.

CCN3509   **Symbol &1 on a pragma &2 was not found.**

Explanation

For a list of all valid options for pragma directives, see the C/C++ Language Reference.

In the message text:

&1 is a symbol name, &2 is a pragma name.

User response

Ensure the pragma syntax and options are correct.

CCN3512   **An initializer is not allowed for "&1".**

Explanation

An attempt was made to initialize an identifier whose type does not permit initialization.

In the message text:

&1 is a C name or keyword.

User response

Remove the initializer.

CCN3513   **Array element designator exceeds the array dimension. Designator will be ignored.**

Explanation

The value of the designator was larger than the dimension declared for the array object.

User response

Change the expression forming the array index.

CCN3514   **Array element designator cannot be applied to an object of type "&1".**

Explanation

An array element designator can only be applied to an object of array type.

In the message text:

&1 is a type.

User response

Remove the subscript.

CCN3515   **Member designator cannot be applied to an object of type "&1".**

Explanation

A member designator can only be applied to an object of type struct or union.

In the message text:

&1 is a type.

User response

Remove the member designator.

CCN3517   **Option &1 for pragma is not supported.**

Explanation

For a list of all valid options for pragma directives, see the C/C++ Language Reference.

In the message text:

&1 is an option name.

User response

Ensure the pragma syntax and options are correct.

CCN3518	Option(s) for pragma &1 are missing or incorrectly specified.

Explanation

pragma &1 is not correctly specified.

In the message text:

&1 is a pragma name.

User response

Ensure the pragma syntax and options are correct.

CCN3519	Index operator ([]) cannot be applied to pointer to void.

Explanation

Index operator ([]) can only be applied to arrays or pointers to objects.

User response

Change the operand.

CCN3520	Switch block begins with declarations or unlabeled statements that are unreachable.

Explanation

Code within a switch block must be labeled with either "case" or "default" to be reachable.

User response

Add a label or remove the unreachable code.

CCN3521	Pointer arithmetic can only be applied to a arrays that are lvalues.

Explanation

Because the array is compiler-generated, it is not an lvalue. Therefore, you cannot apply pointer arithmetic to it.

User response

Change the expression.

CCN3522	Unable to open precompiled header &1 for output.

Explanation

The compiler was unable to open the precompiled header file.

In the message text:

&1 is a file name.

User response

Ensure that the compiler has write access to the precompiled header files.

CCN3524	The _Packed qualifier can only qualify a struct or union.

Explanation

The _Packed qualifier is only valid for structures and unions.

User response

Remove _Packed qualifier.

CCN3531	End of precompiled header processing.

Explanation

The compiler has finished processing a precompiled header.

User response

No response required. This message merely traces the activity of the precompiled header processing.

CCN3545	The decimal size is outside the range of 1 to &1.

Explanation

The specified decimal size should be between 1 and DEC_DIG.

In the message text:

&1 is an integer.

User response

Specify the decimal size between 1 and DEC_DIG.

CCN3546   **The decimal precision is outside the range of 0 to &1.**

Explanation

The specified decimal precision should be between 0 and DEC_PRECISION.

In the message text:

&1 is an integer.

User response

Specify the decimal precision between 0 and DEC_PRECISION.

CCN3547   **The decimal size is not valid.**

Explanation

The decimal size must be a positive constant integral expression.

User response

Specify the decimal size as a positive constant integral expression.

CCN3548   **The decimal precision is not valid.**

Explanation

The decimal precision must be a constant integral expression.

User response

Specify the decimal precision as a constant integral expression.

CCN3549   **The decimal precision is bigger than the decimal size.**

Explanation

The specified decimal precision should be less than or equal to the decimal size.

User response

Specify the decimal precision less than or equal to the decimal size.

CCN3550   **The decimal constant is out of range.**

Explanation

The compiler detected a decimal overflow in scanning a decimal constant.

User response

Change the decimal constant so that it does not exceed the maximum value.

CCN3551   **The fraction part of the result was truncated.**

Explanation

Due to limitations on the number of digits representable, the calculated intermediate result may result in truncation in the decimal places after the operation is performed.

User response

Check to make sure that no significant digit is lost.

CCN3552   **The pre- and post- increment and decrement operators cannot be applied to type &1.**

Explanation

The decimal types with no integral part cannot be incremented or decremented.

In the message text:

&1 is a type.

User response

Reserve at least one digit in the integral part of the decimal types.

CCN3553   **Only decimal types can be used with the &1 operator.**

Explanation

The operand of the digitsof or precisionof operator is not valid. The digitsof and precisionof operators can only be applied to decimal types.

In the message text:

&1 is an operator.

User response

Change the operand.

CCN3554   **Whole-number-part digits in the result may have been lost.**

Explanation

Due to limitations on the number of digits representable, the calculated intermediate result may

result in loss of digits in the integer portion after the operation is performed.

User response

Check to make sure that no significant digit is lost.

CCN3555	Digits have been lost in the whole-number part.

Explanation

In performing the operation, some nonzero digits in the whole-number part of the result are lost.

User response

Check to make sure that no significant digit is lost.

CCN3556	Digits may have been lost in the whole-number part.

Explanation

In performing the operation, some digits in the whole-number part of the result may have been lost.

User response

Check to make sure that no significant digit is lost.

CCN3557	The name in option &1 is not valid. The option is reset to &2.

Explanation

The name specified as a suboption of the option is syntactically or semantically incorrect and thus can not be used.

In the message text:

&1 and &2 are option names.

User response

Make sure that the suboption represents a valid name. For example, in option LOCALE(localename), the suboption "localename" must be a valid locale name which exists and can be used. If not, the LOCALE option is reset to NOLOCALE.

CCN3558	pragma &1 is ignored because the locale compiler option is not specified.

Explanation

The LOCALE compiler option is required for pragma &1.

In the message text:

&1 is a pragma name.

User response

Remove all the pragma &1 directives or specify the locale compiler option.

CCN3559	pragma filetag is ignored because the conversion table from &1 to &2 cannot be opened.

Explanation

During compilation, source code is converted from the code set specified by pragma filetag to the code set specified by the locale compiler option, if they are different. A conversion table form &1 to &2 must be loaded prior to the conversion. No conversion is done when the conversion table is not found.

In the message text:

&1 and &2 are code sets.

User response

Create the conversion table from &1 to &2 and ensure it is accessible from the compiler. If message files are used in the application to read and write data, a conversion table from &2 to &1 must also be created to convert data from the runtime locale to the compile time locale.

CCN3560	Error messages are not converted because the conversion table from &1 to &2 cannot be opened.

Explanation

Error messages issued by C/370 are written in code page 1047. These messages must be converted to the code set specified by the locale compiler option because they may contain variant characters, such as #. Before doing the conversion, a conversion table from &1 to &2 must be loaded. The error messages are not converted because the conversion table cannot be found.

In the message text:

&1 and &2 are code sets.

User response

Make sure the conversion table from &1 to &2 is accessible from the compiler.

CCN3561	No conversion on character &1 because it does not belong to the input code set &2.

Explanation

No conversion has been done for the character because it does not belong to the input code set.

In the message text:

&1 is a character, &2 is a code set.

User response

Remove or change the character to the appropriate character in the input code set.

CCN3562   **Incomplete character or shift sequence was encountered during the conversion of the source line.**

Explanation

Conversion stops because an incomplete character or shift sequence was encountered at the end of the source line.

User response

Remove or complete the incomplete character or shift sequence at the end of the source line.

CCN3563   **Only conversion tables that map single byte characters to single byte characters are supported.**

Explanation

Compiler expected single byte to single byte character mapping during conversion. Conversion stops when there is insufficient space in the conversion buffer.

User response

Make sure the conversion table is in single byte to single byte mapping.

CCN3564   **Invalid conversion descriptor was encountered during the conversion of the source line.**

Explanation

No conversion was performed because conversion descriptor is not valid.

User response

No response required.

CCN3565   **pragma &1 must appear on the first directive before any C code.**

Explanation

The specified pragma must be the first directive before any C code.

In the message text:

&1 pragma type *CHAR 100

User response

Put this pragma as the first directive before any C code.

CCN3566   **Option DECK ignored because option OBJECT specified.**

Explanation

The second option must not be specified for the first to have an effect.

User response

Remove the first or second option.

CCN3567   **Option OFFSET ignored because option LIST not specified.**

Explanation

The second option must be specified for the first to have an effect.

User response

Specify the second option, or remove the first.

CCN3568   **The external name &1 in pragma csect conflicts with another csect name.**

Explanation

A pragma csect was specified with a name which has already been specified as a csect name.

In the message text:

&1 is an identifier.

User response

Ensure that the two csect names are unique.

CCN3569   **A duplicate pragma csect(&1) is ignored.**

Explanation

Only one pragma csect may be specified for either CODE or STATIC.

In the message text:

&1 is the list of arguments used.

User response

Remove the duplicate pragma csect.

CCN3570   **The pragma map name &1 must not conflict with a pragma csect name or the csect name generated by the compiler.**

Explanation

The external name used in the pragma map is identical to the external name specified by the pragma csect or the name generated by the compiler.

In the message text:

&1 is an identifier.

User response

Change the name of the pragma csect or turn off the CSECT option.

CCN3571   **The external name &1 must not conflict with the name in pragma csect or the csect name generated by the compiler.**

Explanation

The external name specified is identical to the name specified by a pragma csect or the name generated by the CSECT option.

In the message text:

&1 is an identifier.

User response

Change the name of the pragma csect or turn off the CSECT option.

CCN3572   **Expected text &1 was not encountered on option &2.**

Explanation

Missing text &1 for option &2.

In the message text:

&1 is text, &2 is an option name.

User response

Use the correct syntax for specifying the option

CCN3573   **To use the built-in form of the &1 function add the #include <&2> directive.**

Explanation

Include the header file &2 to use the &1 built-in function.

In the message text:

&1 is a function name, &2 is a header file name.

User response

Add the specified #include in order to optimize code.

CCN3574   **Unable to open event file &1.**

Explanation

The compiler was unable to open the event file.

In the message text:

&1 is a file name.

User response

Ensure that there is enough disk space.

CCN3575   **Csect option is ignored due to naming error.**

Explanation

The compiler was unable to generate valid csect names.

User response

Use pragma csect to name the code and static control sections.

CCN3576   **Csect name &1 has been truncated to &2.**

Explanation

The static, data and test csect names have been truncated to 8 characters.

In the message text:

&1 and &2 are csect names.

User response

Use the GOFF or LONGNAME option.

CCN3578 **The csect name &1 must not conflict with a csect name generated by the compiler.**

Explanation

The code and static csect names are identical. Either the compiler is unable to generate unique names or a pragma csect is using a duplicate name.

In the message text:

&1 is a csect name.

User response

Use pragma csect to name the code and static control sections.

CCN3585 **Obsolete option HWOPTS defaults to corresponding ARCHITECTURE option.**

Explanation

HWOPTS is no longer supported and has been replaced by ARCHITECTURE.

User response

Use the ARCHITECTURE option to take advantage of hardware.

CCN3586 **Test csect name &1 has been truncated to &2.**

Explanation

The compiler generated test csect name has been truncated to 8 characters.

In the message text:

&1 and &2 are csect names.

User response

Use the CSECT() option to allow test csect names longer than 8 chars.

CCN3600 **3600 - 3631 are Language Environment messages.**

Explanation

Refer to the Language Environment manuals for further information about these messages

User response

Refer to the Language Environment manuals for the appropriate user response.

CCN3610 **"&1" is not allowed as an array element type.**

Explanation

The type &1 can not be used as an array element type.

In the message text:

&1 is a type.

User response

Use a different array element type.

CCN3671 **The header file name in the #include directive cannot be empty.**

Explanation

The #include directive must specify a header file.

User response

Specify a non-empty header file name in the #include directive.

CCN3675 **The return type is not valid for a function of this linkage type.**

Explanation

The function definition violates the restriction on the return type for the specified linkage.

User response

Check the linkage type restrictions and change the return type.

CCN3676 **Function "&1" which returns a return code cannot be defined.**

Explanation

This function has been defined with a FORTRAN linkage type and the RETURNCODE option. A FORTRAN function should be defined in a FORTRAN source file and only referenced in this compilation unit.

In the message text:

&1 is a function or type name.

User response

Either remove the FORTRAN linkage or move the FORTRAN function definition into a FORTRAN source file.

CCN3677	Option LONGNAME is turned on because option DLL is specified.

Explanation

Option LONGNAME is turned on by the compiler because DLL option is specified.

User response

Specify the LONGNAME option when compiling with the DLL option.

CCN3678	Option RENT is turned on because option DLL is specified.

Explanation

Option RENT is turned on by the compiler because DLL option is specified.

User response

Specify the RENT option when compiling with the DLL option.

CCN3679	Option LONGNAME is turned on because option EXPORTALL is specified.

Explanation

Option LONGNAME is turned on by the compiler because EXPORTALL option is specified.

User response

Specify the LONGNAME option when compiling with the EXPORTALL option.

CCN3680	Option RENT is turned on because option EXPORTALL is specified.

Explanation

Option RENT is turned on by the compiler because EXPORTALL option is specified.

User response

Specify the RENT option when compiling with the EXPORTALL option.

CCN3681	pragma export(&1) is ignored; both LONGNAME and RENT options must be specified.

Explanation

The variable/function is not exported because both LONGNAME and RENT must be specified to export functions/variables.

In the message text:

&1 is a function or variable name.

User response

Make sure both the LONGNAME and RENT options are specified.

CCN3682	"&1" will not be exported because pragma variable(&2,NORENT) is specified.

Explanation

Variables with NORENT option cannot be exported.

In the message text:

&1 and &2 are identifiers.

User response

Remove the pragma variable directive.

CCN3683	"&1" will not be exported because it does not have external storage class.

Explanation

Only objects with external storage class can be exported.

In the message text:

&1 is an identifier.

User response

Change the storage class for &1 to extern.

CCN3684	Exporting function main is not allowed.

Explanation

Main cannot be exported.

User response

Remove the pragma export for main.

CCN3685   **"&1" will not be exported because it is not externally defined.**

Explanation

The variable cannot be exported because it is not defined here.

In the message text:

&1 is an identifier.

User response

Remove the pragma export for the variable.

CCN3686   **Unexpected keyword(s). One or more keywords were found in an invalid location.**

Explanation

One or more keywords were found in an invalid location.

User response

Remove the keyword(s) or place them immediately to the left of the identifier to which they apply.

CCN3687   **The &1 keyword cannot be applied to the return type of a function.**

Explanation

The keyword is being applied to the return type of a function.

In the message text:

&1 is a keyword.

User response

Remove the keyword.

CCN3688   **Declaration cannot specify conflicting keywords &1 and &2.**

Explanation

The keywords conflict and cannot both be used in the same declaration.

In the message text:

&1 and &2 are keywords.

User response

Remove one of the keywords.

CCN3689   **The &1 keyword was specified more than once in the declaration.**

Explanation

The keyword was used more than once in the same declaration.

In the message text:

&1 is a keyword.

User response

Remove the duplicate keywords.

CCN3690   **Built-in function &1 is unrecognized. The default linkage convention is used.**

Explanation

The function specified in the pragma linkage builtin is not a built-in function.

In the message text:

&1 is a function name.

User response

Check the function name and correct; or remove the pragma if it is not a built-in function.

CCN3691   **The &1 keyword can only be applied to functions.**

Explanation

The keyword has been applied to an identifier which does not correspond to a function type.

In the message text:

&1 is a keyword.

User response

Check that the correct identifier is specified or remove the keyword.

CCN3693   **The &1 keyword conflicts with a previously specified keyword.**

Explanation

The keyword conflicts with another keyword specified in the same declaration.

In the message text:

&1 is a keyword.

User response

Remove one of the keywords.

CCN3694　　　　**Option LONGNAME is turned on because a qualifier is specified on the CSECT option.**

Explanation

Option LONGNAME is turned on by the compiler when the CSECT option is specified with a qualifier.

User response

Specify the LONGNAME option when compiling with the CSECT option with a qualifier specified.

CCN3695　　　　**pragma export(&1) is ignored; LONGNAME option must be specified.**

Explanation

The variable/function is not exported because LONGNAME must be specified to export functions/variables.

In the message text:

&1 is a function or variable name.

User response

Make sure the LONGNAME option is specified.

CCN3708　　　　**Only functions or typedefs of functions can be specified on pragma linkage directive.**

Explanation

The name specified on pragma linkage is not a function.

User response

Check for typo errors; remove the pragma linkage.

CCN3709　　　　**Structure members cannot follow zero-sized array.**

Explanation

The zero-sized array must be the last member in the structure.

User response

Remove members that occur after the zero-sized array.

CCN3710　　　　**Option &1 is ignored because option &2 is specified.**

Explanation

The second option must not be specified for the first to have an effect.

In the message text:

&1 and &2 are option names.

User response

Remove the first or second option.

CCN3712　　　　**Duplicate function specifier "&1" is ignored.**

Explanation

The indicated function specifier appears more than once.

In the message text:

&1 is a function specifier.

User response

Remove one of the duplicate function specifiers.

CCN3713　　　　**Keyword "&1" is not allowed in this context.**

Explanation

The specified keyword cannot be used in this context.

In the message text:

&1 is a keyword.

User response

Remove the keyword.

CCN3714　　　　**#include searching for file &1.**

Explanation

This is a compiler informational message used to show #include file searching.

In the message text:

&1 is a file name.

User response

No response required.

CCN3715 Storage class &1 cannot be used for structure members.

Explanation

The storage class is not appropriate for this declaration. Restrictions include: 1) Storage class specifier is not allowed on aggregate members, casts, sizeof or offsetof declarations. 2) Declarations at file scope cannot have "register" or "auto" storage class.

In the message text:

&1 is a storage class specifier.

User response

Remove the storage class specifier.

CCN3721 The "&1" qualifier is not supported on the target platform.

Explanation

The specified qualifier is not supported on the target platform and will have no effect.

In the message text:

&1 is a qualifier.

User response

Remove the qualifier.

CCN3722 pragma linkage &1 ignored for function &2.

Explanation

A conflicting linkage type has been specified for this function.

In the message text:

&1 is a linkage type, &2 is a function name.

User response

Check what has been specified before and remove the conflicts.

CCN3723 pragma environment is ignored because function &1 already has linkage type &2.

Explanation

A pragma linkage has already been specified and used for this function, and is in conflict with the pragma environment directive. The latter is ignored.

In the message text:

&1 is a function name, &2 is a linkage type.

User response

Remove the pragma linkage or environment directive.

CCN3724 Undefined identifier "&1" was referenced in pragma &2 directive.

Explanation

A pragma is referring to an identifier that has not been defined.

In the message text:

&1 is an identifier name, &2 is the name of the pragma.

User response

Define the identifier or remove the pragma.

CCN3728 Operation between types "&1" and "&2" is not recommended.

Explanation

The operation specified is improper between the operands having the given types.

In the message text:

&1 and &2 are types.

User response

Either change the operator or the operands.

CCN3729 "&1" should not be declared inline or static.

Explanation

Although "&1" is not a keyword, it is a special function that cannot be inlined or declared as static.

In the message text:

&1 is a function name.

User response

Remove the inline or static specifier from the declaration of "&1".

CCN3730 The pragma is accepted by the compiler. The pragma will have no effect.

Explanation

The pragma is not supported by this compiler.

User response

The pragma can be removed.

CCN3731	The &1 keyword is not supported on the target platform. The keyword is ignored.

Explanation

The specified keyword is not supported on the target platform and will have no effect.

In the message text:

&1 is a keyword.

User response

Remove the keyword.

CCN3732	pragma &1 is not supported on the target platform.

Explanation

The specified pragma is not supported on the target platform and will have no effect. See the C/C++ Language Reference for the list of valid pragma directives.

In the message text:

&1 is a pragma name.

User response

Change or remove the pragma directive.

CCN3733	Processing #include file &1.

Explanation

This message traces #include file processing.

In the message text:

&1 is a file name.

User response

No response required.

CCN3735	Suboption &1 of option &2 is ignored because option &3 is specified.

Explanation

Suboption &1 of &2 cannot be specified with option &3. &1 is ignored.

In the message text:

&1 is a suboption, &2 and &3 are option names.

User response

Remove the suboption &1 or the option &3.

CCN3736	&1 conflicts with previous &2 declaration.

Explanation

The compiler cannot resolve the conflicting declarations.

In the message text:

&1 and &2 are function specifiers.

User response

Remove one of the declarations.

CCN3737	The preprocessor macro "&1" was expanded inside a pragma directive.

Explanation

A macro was expanded in the context of a pragma directive. Please ensure that this is the result that you want.

In the message text:

&1 is a preprocessor macro.

User response

Ensure that the macro was intended for expansion.

CCN3742	64-bit portability: possible loss of digits through conversion of &1 type into &2 type.

Explanation

A long type is assigned to an int type, which may cause truncation in 64-bit mode.

In the message text:

&1 and &2 are types.

User response

Check the possible value ranges of the long type or change the assignment from an int type to a long type.

CCN3743	64-bit portability: possible change of result through conversion of &1 type into &2 type.

Explanation

An int type is assigned to a long type, which may cause unexpected results in 64-bit mode.

In the message text:

&1 and &2 are types.

User response

Check if a possible sign extension of int type into long type causes unexpected results.

CCN3744	64-bit portability: possible truncation of pointer through conversion of pointer type into &1 type.

Explanation

A pointer type is assigned to an integer type, which may lead to an invalid address in 64-bit mode.

In the message text:

&1 is a type.

User response

Use a long type to hold a pointer type.

CCN3745	64-bit portability: possible incorrect pointer through conversion of &1 type into pointer.

Explanation

An integer type is assigned to a pointer type, which may lead to an invalid address in 64-bit mode.

In the message text:

&1 is a type.

User response

Use a long type to hold the address.

CCN3746	64-bit portability: possible change of constant value through conversion into long type.

Explanation

A constant is assigned into long type leading to possible change of value in 64-bit mode.

User response

Check the possible value ranges of the constant when stored in a long type.

CCN3747	64-bit portability: constant given type "&1" when compiling in 32-bit mode may be given type "&2" when compiling in 64-bit mode.

Explanation

A constant which is given type unsigned long int in 32-bit mode may fit into a long int in 64-bit mode. A constant which is given type long long int in 32-bit mode may fit into a long int in 64-bit mode. A constant which is given type unsigned long long int in 32-bit mode may fit into an unsigned long int in 64-bit mode.

In the message text:

&1 and &2 are types.

User response

Check the use of the constant for possible changes in usual arithmetic conversion rules as it propagates through expressions.

CCN3748	64-bit portability: constant which will overflow in 32-bit mode may select unsigned long int or long int in 64-bit mode

Explanation

A constant larger than UINT_MAX but smaller than ULONGLONG_MAX will overflow in 32-bit mode, but be acceptable in an unsigned long or signed long in 64-bit mode.

User response

Make sure you intend this constant to be acceptable in 64-bit mode.

CCN3752	Number of enumerator constants exceeds &1.

Explanation

The number of enumerator constants must not exceed the value of &1.

In the message text:

&1 is an integer.

User response

Remove additional enum constants.

CCN3754	The parameter type is not valid for a function of this linkage type.

Chapter 2. z/OS XL C/C++ compiler return codes and messages **107**

Explanation

The linkage type of the function puts certain restrictions on the parameter type, which the function definition violated.

User response

Check the linkage type restrictions and change the parameter type.

CCN3755	The &1 option is not supported in this release.

Explanation

The specified option is not supported in this release.

In the message text:

&1 is an option name.

User response

Remove the option.

CCN3763	Option &1 is ignored because pragma &2 is specified.

Explanation

The pragma must not be specified for the option to have an effect.

In the message text:

&1 is an option name, &2 is a pragma name.

User response

Remove the pragma or the option.

CCN3764	Option &1 is ignored for variable &2 because pragma &3 is specified.

Explanation

The pragma must not be specified for the option for the variable indicated to have an effect.

In the message text:

&1 is an option name, &2 is an identifier, &3 is a pragma name.

User response

Remove the pragma or the option for the variable indicated.

CCN3765	&1 digits are required for the universal character name "&2".

Explanation

Universal character names must follow the format \UNNNNNNNN or \uNNNN, where N is a hexadecimal digit.

In the message text:

&1 is an integer and &2 is a universal character name.

User response

Either pad or truncate the digits used for the universal character name.

CCN3766	The universal character name "&1" is not in the allowable range for an identifier.

Explanation

Hexadecimal values representing characters in the basic character set (base source code set) and the code points reserved by ISO/IEC 10646 for control characters are not allowed. The following characters are also disallowed: (1) Any character that has a short identifier that is less than 00A0. The exceptions are 0024 ($), 0040 (@), or 0060 (`). (2) Any character that has a short identifier that is in the code point range D800 through DFFF inclusive.

In the message text:

&1 is a universal character name.

User response

Change the universal character name to an allowable one.

CCN3767	Packed decimal constant &1 is not valid.

Explanation

See the C/C++ Language Reference for a description of a packed decimal constant.

In the message text:

&1 is a numeric value.

User response

Ensure that the packed decimal constant does not contain any characters that are not valid.

CCN3776	The required conditions for using the built-in function "&1" are not met.

Explanation

The built-in function "&1" requires one or more compiler options that are not currently active.

In the message text:

&1 is a function name.

User response

Specify the correct options to use the built-in function.

CCN3777	The parameter in position &1 must be a constant literal for the built-in function "&2".

Explanation

The built-in function "&2" requires parameter &1 to be a constant literal.

In the message text:

&1 is an integer, &2 is a function name.

User response

Specify a constant literal for the parameter.

CCN3778	Type "&1" is not valid. Type specifier "&2" is assumed.

Explanation

The type "&1" is not valid; it is treated as "&2".

In the message text:

&1 is a type, &2 is a type specifier.

User response

Replace the unknown type specifier with a correct one.

CCN3779	Definition of modifiable static variable "&1" is not allowed within inline definition of "&2".

Explanation

An inline definition of function "&2" with external linkage shall not contain a definition of modifiable object "&1" with static storage duration. The static keyword is ignored.

In the message text:

&1 is an identifier, &2 is a function name.

User response

Remove the static storage class specifier.

CCN3780	Reference to "&1" with internal linkage is not allowed within inline definition of "&2".

Explanation

An inline definition of function "&2" with external linkage shall not contain a reference to an identifier "&1" with internal linkage.

In the message text:

&1 is an identifier, &2 is a function name.

User response

Remove the reference to the identifier with internal linkage.

CCN3781	Inline function "&1" is undefined.

Explanation

An inline function was declared and referenced in this file. The definition of the function was not found before the end of the file. When a function is declared to be inline, the function definition must appear in the same file.

In the message text:

&1 is a function name.

User response

Define the function in the file or remove the inline function specifier.

CCN3782	One or more error messages have been disabled.

Explanation

One or more error messages have been suppressed via user's request.

User response

Fix the errors to proceed with the compilation.

CCN3784	Decimal integer constant "&1" is out of range.

Explanation

The specified decimal constant is too large to be represented by a signed long long int.

In the message text:

&1 is an integer constant.

User response

The constant integer must have a value less than LONGLONG_MAX defined in <limits.h>.

CCN3785	Illegal suffix "&1" for integer constant "&2".

Explanation

Valid integer suffixes for a long long integer constant are ll or LL. Valid integer suffixes for an unsigned long long integer constant are ull, uLL, Ull, or ULL.

In the message text:

&1 is a suffix, &2 is an integer constant.

User response

Change or remove the suffix.

CCN3787	Hexadecimal floating-point constant "&1" cannot be represented exactly in its evaluated format.

Explanation

Due to limits on the number of significant digits, the hexadecimal floating-point constant is rounded.

In the message text:

&1 is a hexadecimal floating-point constant.

User response

Change the hexadecimal floating-point constant so that it fits in the evaluation format.

CCN3789	The operand of __alignof__ cannot be a bit field.

Explanation

The __alignof__ operator cannot be applied to an lvalue that designates a bit field object.

User response

Change the operand.

CCN3805	String literal exceeded the compiler limit of &1.

Explanation

String literal size cannot be larger than the compiler limit

In the message text:

&1 is an integer.

User response

Reduce the size of the string literal.

CCN3810	pragma runopts syntax (&1): &2

Explanation

Syntax error in the pragma. The suboption syntax is the same as the corresponding Language Environment runtime option. Please refer to the Language Environment manual for details on the CEEnnnn message number.

In the message text:

&1 and &2 are error messages.

User response

Correct the syntax error.

CCN3811	Option &1 forces option &2 to take effect.

Explanation

The first option in the message forces the second one to take effect. Specify the second option explicitly to suppress this message.

In the message text:

&1 and &2 are option names.

User response

Specify the second option explicitly.

CCN3812	Option FLOAT(IEEE) may cause slow execution time when used with ARCH less than 3.

Explanation

Binary floating-point operations (BFP) needs hardware architecture (ARCH option) of 3 or higher. For ARCH less than 3, BFP will work on OS level V2R6 or higher, which provides software emulation, but will significantly slow down the execution time.

User response

If the target hardware architecture is 3 or higher, specify it explicitly in ARCH.

CCN3813	Option FLOAT(AFP) may cause slow execution time when used with ARCH less than 3.

Explanation

The AFP suboption needs hardware architecture (ARCH option) of 3 or higher. For ARCH less than 3, BFP will work on OS level V2R6 or higher, which provides software emulation, but will significantly slow down the execution time.

User response

If the target hardware architecture is 3 or higher, specify it explicitly in ARCH.

CCN3815	Conflicting qualifiers &1 and &2 specified.

Explanation

The identified qualifiers cannot both be specified at the same time.

In the message text:

&1 and &2 are qualifiers.

User response

Remove one of the qualifiers.

CCN3870	The program name &1 has been truncated to &2.

Explanation

The program name exceeds the maximum length of 10 characters and has been truncated. This may result in unexpected behavior if two different names become the same name after truncation.

In the message text:

&1 and &2 are program names.

User response

Reduce the length of the program name. Alternatively, use pragma map to shorten the program name.

CCN3885	An anonymous union or struct declared at file scope must be static.

Explanation

Anonymous unions and structs are not allowed at global scope if they are not static.

User response

Declare all anonymous tags to be static at file scope.

CCN3894	The &1 is not valid in 64-bit mode and it is ignored.

Explanation

The &1 is not valid in 64-bit mode. It is only supported in 32-bit mode.

In the message text:

&1 is an option or pragma name.

User response

Either remove &1 or compile it in 32-bit mode.

CCN3897	Unstructured goto statement encountered.

Explanation

The target label of a goto statement should not be located in an inner block such as a loop.

User response

Ensure the target label of the goto statement is not located in an inner block.

CCN3913	The enum constants must be specified when the enum tag is declared.

Explanation

When an enumeration tag is declared, the list of the enumeration constants must be included in the declaration.

User response

Add the list of enumeration constants in the enum tag declaration.

CCN3914	Code page (CCSID) &1 specified on pragma convert directive is not valid.

Explanation

The CCSID &1 specified on the pragma convert directive is either not supported by the system or an error occurred while the compiler was trying to access code page information.

In the message text:

&1 is a CCSID number.

User response

Use a valid code page (CCSID).

CCN3919   **Variable &1 was not explicitly initialized.**

Explanation

If not explicitly initialized, variables with storage class auto or register contain indeterminate values.

In the message text:

&1 is an identifier.

User response

Initialize the variable.

CCN3920   **Bitwise operator is applied to a signed type.**

Explanation

Bitwise operators may change the value of a signed type by shifting the bit used to indicate the sign of the value.

User response

Change the operand to an unsigned type or remove the bitwise operation.

CCN3931   **Dependency file &1 cannot be opened.**

Explanation

Makedepend could not open the specified dependency file.

In the message text:

&1 is a file name.

User response

Ensure the source file name is correct. Ensure that the correct file is being read and has not been corrupted. If the file is located on a LAN drive, ensure the LAN is working properly. Also, the file may be locked by another process or access may be denied because of insufficient permission

---

CCN3932   **Too few options specified for makedepend.**

Explanation

Makedepend has been supplied with too few command line options to continue. Please consult the

``z/OS UNIX System Services Command Reference'' for usage information.

User response

Specify correct number of options for makedepend.

CCN3933   **Specify at least one source operand to be processed.**

Explanation

No source files were specified for makedepend processing.

User response

Specify at least one source operand.

CCN3934   **Compiler option &1 is invalid for compiler version &2.**

Explanation

An invalid option was specified for the compiler version specified for makedepend.

In the message text:

&1 is a compiler option, &2 is compiler version.

User response

Change the compiler version to the version that accepts this option, or remove this option.

CCN3935   **Specify a valid -W phase code (0 or c=compile, m=makedepend) instead of &1.**

Explanation

An invalid compiler phase was specified for makedepend.

In the message text:

&1 is a phase code.

User response

Specify a phase that is accepted by makedepend.

CCN3936   **Specify a series of options, separated by commas, for the -W m option.**

Explanation

No options were specified for the -W m option.

User response

Remove the -W m option.

CCN3937 **&1 has a dependency on include file &2 which is located in an MVS data set.**

Explanation

The specified #include file was found in an MVS data set. No dependency information will be recorded for this #include file.

In the message text:

&1 is an object file. &2 is a #include file.

User response

No response required.

CCN3938 **Unknown compiler version &1 for makedepend option V. Using default compiler version.**

Explanation

An invalid compiler version was specified for the makedepend option V.

In the message text:

&1 is a compiler version.

User response

Correct the compiler version.

CCN3941 **Applying &1 may cause unexpected run-time behavior.**

Explanation

Compile may be successful but it may cause unexpected run-time behavior.

In the message text:

&1 is an option or pragma.

User response

Change or remove the offending option or pragma.

CCN3942 **Attribute "&1" causes a conflict and is ignored.**

Explanation

The identified attribute is in conflict with a previously specified attribute or pragma and is ignored.

In the message text:

&1 is an attribute name.

User response

Change or remove the conflicting attribute specifier.

CCN3943 **Attribute "&1" is not supported on the target platform and is ignored.**

Explanation

The identified attribute specifier is not supported on the target platform and is ignored.

In the message text:

&1 is an attribute name.

User response

Remove the attribute specifier.

CCN3944 **Attribute "&1" is not supported and is ignored.**

Explanation

The identified attribute is not supported and is ignored.

In the message text:

&1 is an attribute name.

User response

Remove the attribute specifier.

CCN3945 **The number of arguments specified for attribute "&1" is incorrect; this attribute is ignored.**

Explanation

The number of arguments specified for the identified attribute is incorrect.

In the message text:

&1 is an attribute name.

User response

Check the syntax rules for the specified attribute, and correct the arguments.

CCN3946 **Incorrect argument type specified for attribute "&1"; this attribute is ignored.**

Explanation

The argument specified for the identified attribute has the wrong data type.

In the message text:

&1 is an attribute name.

User response

Check the syntax rules for the specified attribute, and correct the argument.

CCN3947	The explicit register specifier is unexpected and is ignored.

Explanation

An explicit register cannot be specified on this type of declaration.

User response

Remove the explicit register specifier.

CCN3949	A header name is expected in the #include_next directive.

Explanation

No header file name is provided after the #include_next directive.

User response

Specify the header file name. Enclose the system header names in angle brackets and the user header names in double quotation marks.

CCN3950	The #include_next file &1 is not found.

Explanation

The file specified in the #include_next directive cannot be found. See the documentation for the file search order.

In the message text:

&1 is a file name.

User response

Ensure that the file exists. Change the name of the included file to one that exists or use the include path option to specify the path to the file.

CCN3951	The header file name in the #include_next directive is empty.

Explanation

The header file name in the #include_next directive should not be empty.

User response

Specify a non-empty header file name in the #include_next directive.

CCN3952	An #include_next header must end before the end of the source line.

Explanation

An #include_next directive was specified across two or more lines.

User response

Change the #include_next directive so that it and its arguments are contained on a single line.

CCN3953	No path to find #include_next file.

Explanation

There is no search path to find the #include_next file.

User response

Ensure the search path is correct.

CCN3954	An #include_next directive is found in a primary source file.

Explanation

The #include_next directive should only be used in header files.

User response

Change the #include_next to an #include.

CCN3955	Type "int" is assumed for declaration of "&1".

Explanation

A declaration was made without a type specifier.

In the message text:

&1 is an identifier.

User response

Add the type specifier into the declaration.

CCN3963   **The attribute "&1" is not a valid variable attribute and is ignored.**

Explanation

The identified attribute specifier is ignored because it does not apply to variables.

In the message text:

&1 is an attribute name.

User response

Remove the attribute specifier.

CCN3970   **Incorrect _Pragma operator.**

Explanation

Error in _Pragma operator.

User response

Correct the syntax in the _Pragma Operator.

CCN3971   **Invalid standard pragma.**

Explanation

Error(s) in a standard pragma.

User response

Correct the error in the standard pragma.

CCN3973   **pragma FP_CONTRACT OFF overrides option FLOAT(MAF).**

Explanation

When the pragma is turned off anywhere in the program, the option FLOAT(NOMAF) will be set for the entire compilation unit and override option FLOAT(MAF), if it has been specified on the command line.

User response

Remove the command line option FLOAT(MAF) or remove the pragma.

CCN3974   **Attribute "&1" has been specified more than once; the last specification is used.**

Explanation

The identified attribute was specified more than once; the last specification is used.

In the message text:

&1 is an attribute name.

User response

Remove the duplicate attribute specifier.

CCN3976   **Alias specification cannot be provided for a function definition.**

Explanation

An alias specification is not allowed within a function definition.

User response

Remove the alias specification or change the function definition into a function declaration.

CCN3987   **The invalid character "&1" was found in a wide character or wide string literal. The character will be ignored.**

Explanation

The wide character is not valid. The character is displayed as one or more hexadecimal byte values.

In the message text:

&1 is a character.

User response

Correct the literal.

CCN3990   **The maximum size of the stack has been exceeded.**

Explanation

The size of the stack has reached its maximum size, no more entries may be added.

User response

Remove some entries from the stack.

CCN3991   **Only a variable can be declared in the declaration part of a "for" statement.**

Explanation

A tag, a function declaration, or a typedef definition is not allowed in the declaration part of a "for" statement.

User response

Correct the declaration part of the for statement.

CCN3992	Storage class "&1" cannot be used in this context.

Explanation

Only variables having storage class "register" or "auto" can be declared in the declaration part of a "for" statement.

In the message text:

&1 is a storage class specifier.

User response

Delete the storage class specifier or use a different storage class specifier.

CCN3994	A flexible array member is not allowed.

Explanation

A flexible array member is permitted as the last member of a structure containing more than one named member. Unions cannot contain flexible array members.

User response

Correct the usage of the flexible array member.

CCN3995	An aggregate containing a flexible array member cannot be used as a member of a structure or as an array element.

Explanation

A flexible array member is permitted as the last member of a structure containing more than one named member. Such a structure cannot be a member of another structure or an array element, although it can be a member of a union and such a union cannot be a member of a structure or an array element.

User response

Remove the aggregate containing the flexible array member.

CCN3996	Definition of tag "&1" is not allowed.

Explanation

A tag cannot be defined in the declaration part of a "for" statement.

In the message text:

&1 is a tag.

User response

Move the declaration tag so that it is before the "for" statement.

CCN3997	Structure members cannot follow a flexible array member/zero extent array.

Explanation

The flexible array member/zero extent array must be the last member in the structure.

User response

Move the flexible array member/zero extent array to the end of the structure.

CCN3998	A different section was specified for "&1"; the new specification is used.

Explanation

The new section specification overwrites the previous one.

In the message text:

&1 is an identifier.

User response

Remove the previous specification of attribute "section".

CCN4100	A &1 attribute that is not applied to a global or static variable is ignored.

Explanation

The section attribute is not supported for automatic variables, parameters, or variables with external linkage.

In the message text:

&1 is an attribute name.

User response

Remove the section attribute specifier.

CCN4102 **Hexadecimal floating-point constants are not supported in the current language level.**

Explanation

A hexadecimal floating-point constant is not allowed in the current language level.

User response

Change the language level to one that supports the hexadecimal floating-point constant notation, or use the decimal floating-point notation instead.

CCN4103 **__VA_ARGS__ can only be used in the replacement list of a function-like macro with a variable argument list.**

Explanation

The identifier __VA_ARGS__ can only occur in the replacement list part of a C99 function-like macro that uses the ellipsis notation in the parameter list.

User response

Remove the __VA_ARGS__ identifier.

CCN4104 **The static keyword or type qualifiers are ignored unless they are in the outermost array index of a function parameter.**

Explanation

The array index contains the static keyword or type qualifiers. When the static keyword or type qualifiers are used to specify the dimension of an array, they can only be used for the declaration of function parameters and only in the outermost array dimension.

User response

Remove the static keyword or type qualifiers.

CCN4106 **Initializer does not evaluate to a constant that fits in the target type.**

Explanation

The expression used as an initializer evaluates to a number that is not within the range that can be stored by the target.

User response

Change the expression so it evaluates to a value in the valid range.

CCN4107 **Initialization of function pointer "&1" with a function that has "&2" linkage is not allowed.**

Explanation

An attempt was made to initialize a function pointer with the address of a function that has incompatible linkage.

In the message text:

&1 is an identifier, &2 is a linkage.

User response

Ensure the function pointer is initialized with the address of a function that has compatible linkage.

CCN4108 **The use of keyword &1 is non-portable.**

Explanation

The specified keyword may cause problems when porting the code to another system.

In the message text:

&1 is a keyword.

User response

Change the language level to one that supports the specified keyword or remove the use of the specified keyword.

CCN4118 **Character constant &1 has more than 1 character.**

Explanation

A character constant can only have up to four bytes.

In the message text:

&1 is a character constant.

User response

Change the character constant to contain four bytes or less.

CCN4119 **The initializer list should not be empty.**

Explanation

An initializer list should contain at least one initializer.

User response

Remove the empty initializer list or add an initializer to the list.

CCN4124 The use of directive &1 is non-portable.

Explanation

The specified directive may cause problems when porting the code to another system.

In the message text:

&1 is a directive.

User response

Remove the use of the specified directive.

CCN4125 Option &1 forces &2 to take effect due to &3.

Explanation

The first option in the message forces the second one to take effect due to the third option.

In the message text:

&1, &2 and &3 are option names.

User response

Specify the second option explicitly to suppress this message.

CCN4136 Statement expressions are not supported in the current language level.

Explanation

A statement expression is not allowed in the current language level.

User response

Change the language level to one that supports statement expressions.

CCN4137 Only one &1 pragma may be specified for the same loop. This pragma is ignored.

Explanation

Only the pragma immediately preceding a loop will have effect.

In the message text:

&1 is a pragma name.

User response

Specify only one pragma for a loop. Remove any multiple pragmas.

CCN4140 The &1 pragma cannot be applied to a &2 loop. This pragma is ignored.

Explanation

Only specific unrolling optimizations are appropriate for certain loops.

In the message text:

&1 is a pragma name, &2 is a type of loop (i.e. "for", "do while" or "while").

User response

Apply a different unrolling pragma to the loop, or remove the pragma.

CCN4141 The total number of operands exceeds &1.

Explanation

The maximum number of asm operands has been exceeded.

In the message text:

&1 is an integer.

User response

Reduce the number of operands in the asm instruction.

CCN4142 The named operand "&1" is not defined.

Explanation

The operand name specifier does not refer to an operand.

In the message text:

&1 is an identifier.

User response

Change the name specifier of the intended operand.

CCN4143　　　　The operand number is out of range.

Explanation

The operand number specified does not refer to an operand.

User response

Change the operand number to the intended operand.

CCN4145　　　　"&1" is not the first character of the output constraint.

Explanation

"=" or "+" is not the first character of the constraint for an output operand.

In the message text:

&1 is a character.

User response

Make "=" or "+" the first character of the constraint for output operands.

CCN4146　　　　The asm constraint "&1" cannot be used for this operand type.

Explanation

The asm operand constraint is not valid for the provided operand.

In the message text:

&1 is an asm constraint.

User response

Remove the constraint or modify the operand to a compatible type.

CCN4147　　　　The asm constraint "&1" is not supported.

Explanation

The asm operand constraint is not recognized.

In the message text:

&1 is an asm constraint.

User response

Remove the constraint.

CCN4148　　　　The symbolic name "&1" used in a %[name] operand specifier is a duplicate. You must use a unique name.

Explanation

An operand name specifier has been used more than once.

In the message text:

&1 is a symbolic name.

User response

Select a different operand name specifier for one of the operands.

CCN4149　　　　The register name "&1" is unknown.

Explanation

The register name in the clobber set of the asm statement is not recognized. The clobbered register information is ignored.

In the message text:

&1 is a register name.

User response

Correct the name of the clobbered register.

CCN4197　　　　The use of &1 in designated initializer syntax is non-portable.

Explanation

The use of the specified token in a designated initializer is obsolete and non-portable. To maximize code portability, use the standard conforming syntax for a designated initializer.

In the message text:

&1 is a token, for example ':'.

User response

Use the standard conforming syntax for a designated initializer.

CCN4198　　　　Missing &1 in designated initializer syntax.

Explanation

The designated initializer syntax used is obsolete and non-portable. To maximize code portability, use the standard conforming syntax for a designated initializer.

In the message text:

&1 is a token, for example ':'.

User response

Use the standard conforming syntax for a designated initializer.

CCN4230	The built-in function "&1" is not valid for the target system.

Explanation

The built-in function is not valid for the target operating system.

In the message text:

&1 is a function name.

User response

User cannot use this built-in in the current operating system.

CCN4231	The built-in function "&1" is not valid for the target architecture.

Explanation

The built-in function is not valid for the target hardware architecture.

In the message text:

&1 is a function name.

User response

User cannot use this built-in in the target architecture.

CCN4232	The built-in function "&1" requires option "&2".

Explanation

The built-in function is not valid with the current compilation options.

In the message text:

&1 is a function name, &2 is an option name.

User response

In order to use the built-in, the user has to specify the required option.

CCN4233	The parameter in position &1 must be a power of 2 and must be a constant literal for the built-in function "&2".

Explanation

The built-in function requires the argument be a power of 2 and a constant literal.

In the message text:

&1 is a number, &2 is a function name.

User response

In order to use the built-in, the argument must be a power of 2 and a constant literal.

CCN4234	The argument &2 of the built-in function "&1" must be in the range &3.

Explanation

The built-in function requires the argument be in the range.

In the message text:

&1 is a function name, &2 is a number, &3 is a range.

User response

Modify the argument's value to meet the range requirement.

CCN4250	Too many registers are required in the asm statement.

Explanation

The asm statement cannot be compiled because it requires too many registers.

User response

Reduce the complexity of the asm statement.

CCN4252	Output operand is not specified with = or + constraint. = is assumed.

Explanation

An output operand must be specified with an "=" or "+".

User response

Add an "=" or "+" to the constraint set for all output operands.

CCN4253   **GCC asm statement is used. This feature is not standard conforming and is not portable.**

Explanation

asm feature can not be used in a strictly conforming language level.

User response

Remove the asm statements.

CCN4254   **Wide string literals (L, U or u) of different types cannot be concatenated.**

Explanation

A string that is prefixed by L, U or u can only be concatenated with one that is similarly prefixed, or with a normal string literal.

User response

Check the string literal prefix and correct the syntax.

CCN4255   **Source file encoding cannot be converted to Unicode using iconv. The UTF option is ignored.**

Explanation

The compiler converts string literals that are prefixed by U or u to Unicode using iconv. The required UTF-8 converter is not found on the system.

User response

Check that the source file encoding can be converted to UTF-8 by iconv.

CCN4256   **Specify the UTF option to process string literals prefixed by u or U.**

Explanation

The compiler encountered a syntax error possibly caused by string literals prefixed by u or U. The UTF option is needed to process these string literals.

User response

Check if the program is using string literals prefixed by u or U.

CCN4258   **The matching constraint cannot be used in the output operand.**

Explanation

The matching constraint can only appear in the input operands.

User response

Remove the matching constraint from the output operand.

CCN4259   **The matching constraint cannot reference the input operand.**

Explanation

The matching constraint can only refer to an output operand.

User response

Modify the matching constraint.

CCN4260   **The asm operand does not match the specified constraint.**

Explanation

The constraint and the expression of the operand must be consistent.

User response

Modify the constraint or the expression of the operand to make them consistent.

CCN4263   **The matching constraint '&1' has been used more than once.**

Explanation

One output operand can only be referenced by the matching constraint once.

In the message text:

&1 is an asm matching constraint.

User response

Modify the matching constraint.

CCN4266   **This designation of a range of array elements is non-portable.**

Explanation

The designator used to initialize a range of array elements is a non-portable extension. To maximize

code portability, use the standard conforming syntax for designating array elements.

User response

Use the standard conforming syntax to designate array elements.

CCN4270	Option &1 is ignored when option &2 is specified.

Explanation

The option &1 is not valid when used in conjunction with &2.

In the message text:

&1 and &2 are option names.

User response

Remove option &2.

CCN4271	Subption &1 of option &2 is ignored because option &3 is not specified.

Explanation

The suboption &1 is only valid when used in conjunction with &3.

In the message text:

&1 is a suboption, &2 and &3 are option names.

User response

Compile with &3.

CCN4278	Duplicate or overlapping range expression specified for case label. Labels must be unique.

Explanation

Two case label ranges in the same switch statement cannot overlap.

User response

Change one of the label ranges.

CCN4279	The &1 pragma cannot be applied to a #pragma block_loop.

Explanation

Only other block_loop pragmas or loopid pragmas can be applied to a #pragma block_loop.

In the message text:

&1 is a pragma name.

User response

Remove the erroneous loop pragma directive.

CCN4298	The subscript &1 is out of range. The valid range is 0 to &2.

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

&1 and &2 are integers.

User response

Change the index so it falls within the bounds of the array or increase the size of the array. This message is usually generated when the user tries to index the array with the size of the array and forgets to substract one.

CCN4299	The subscript &1 is out of range. The only valid subscript is 0.

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

&1 is an integer.

User response

Change the index so it falls within the bounds of the array or increase the size of the array. This message is usually generated when the user tries to index the array with the size of the array and forgets to substract one.

CCN4300	The subscript &1 is less than zero. The subscript of an array should be greater than or equal to zero.

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

&1 is an integer.

User response

Change the index so it falls within the bounds of the array.

CCN4301   **The subscript &1 may be out of range. The known range is 0 to &2.**

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

&1 and &2 are integers.

User response

Change the index so it falls within the bounds of the array or increase the size of the array.

CCN4302   **The subscript &1 may be out of range. The only known subscript is 0.**

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

&1 is an integer.

User response

Change the index so it falls within the bounds of the array or increase the size of the array.

CCN4307   **Skipping a declaration with variably modified type at line &1 is invalid.**

Explanation

A goto statement shall not jump from outside the scope of an identifier having a variably modified type to inside the scope of that identifier.

In the message text:

&1 is a line number.

User response

Change the program so that the goto statement is inside the scope of the identifier having a variably modified type.

CCN4308   **Operand of unary ++ or -- operator must be a real or pointer type.**

Explanation

The operand of the unary ++ or -- operator should have a real type or pointer type.

User response

Change the type of the operand, or use a different operand.

CCN4312   **pragma noinline conflicts with the inline function specifier for function "&1". The pragma is ignored.**

Explanation

pragma noinline is specified with a function name, which has an inline function specifier.

In the message text:

&1 is a function name.

User response

Remove either pragma noinline or the inline function specifier.

CCN4319   **The string literal specified may not exceed &1 characters. The pragma is ignored.**

Explanation

The length of the string literal exceeds a limit, so the pragma is ignored.

In the message text:

&1 is an integer.

User response

Reduce the length of the string literal.

CCN4320   **A flexible array member is not supported in the current language level.**

Explanation

A flexible array member as the last member of a structure containing more than one named member is not allowed in the current language level.

User response

Change the language level to one that supports flexible array members.

CCN4324   **Array size must have integer type.**

Explanation

When the size of an array is an expression, it must have integer type.

User response

Change the array size expression to an integer type.

CCN4334   **&1 value must contain only decimal digits or only '*'.**

Explanation

A non-numerical and non-asterisk character was encountered in the &1 value.

In the message text:

&1 is a pragma name, for example 'sequence' or 'margins'.

User response

Check the syntax of the value given.

CCN4339   **The specified register name '&1' is not valid.**

Explanation

The named specified for the asm register is not a valid register on the target architecture.

In the message text:

&1 is the name of an invalid register.

User response

Use a valid register name.

CCN4340   **Register variable &1 cannot be initialized.**

Explanation

A register variable cannot be initialized since a register does not occupy storage in the object.

In the message text:

&1 is an identifier.

User response

Remove the initializer.

CCN4343   **Register '&1' is already used for variable &2.**

Explanation

The register has already been reserved for another variable. A register cannot be reserved for multiple variables.

In the message text:

&1 is a register name. &2 is an identifier.

User response

Use a unique register name for each register variable.

CCN4344   **Register variable &1 already reserves register '&2'. This asm specification is ignored.**

Explanation

A register variable cannot reserve more than one register.

In the message text:

&1 is an identifier. &2 is a register name.

User response

Remove one of the register names specified for the variable.

CCN4345   **The data type '&1' of variable &2 is not suitable for a register.**

Explanation

A register specification is used for a variable with a data type that cannot be held in a register.

In the message text:

&1 is a data type. &2 is an identifier.

User response

Remove the register specification for the variable.

CCN4347   **The register variable specification is non-portable.**

Explanation

Using the register variable specification may cause problems when porting the code to another system.

User response

Remove the use of the register variable specification.

CCN4349   **Decimal floating-point constant "&1" is out of range.**

Explanation

The compiler detected a decimal floating-point overflow or underflow while scanning a constant or performing constant arithmetic folding.

In the message text:

&1 is a decimal floating-point constant.

User response

Change the decimal floating-point constant so that it fits in the evaluation format or use a format with higher precision and range. See the XL C/C++ Language Reference for details on the valid format.

CCN4350	Decimal floating-point constant "&1" cannot be represented exactly in its evaluated format.

Explanation

Due to limits on the number of significant digits, the decimal floating-point constant is rounded.

In the message text:

&1 is a decimal floating-point constant.

User response

Change the decimal floating-point constant so that it fits in the evaluation format or use a format with higher precision. See the XL C/C++ Language Reference for details on the valid format.

CCN4351	Decimal floating-point constant "&1" is not valid.

Explanation

See the XL C/C++ Language Reference for a description of a decimal floating-point constant.

In the message text:

&1 is the text in the source program which the compiler tries to parse as a decimal floating-point constant.

User response

Ensure that the decimal floating-point constant does not contain any characters that are not valid.

CCN4359	Host variable &1 is a pointer with unknown length.

Explanation

The size of the memory area pointed to by the host variable cannot be determined.

In the message text:

&1 is a variable name.

User response

Correct the host variable definition by using the bounded pointer definition described in the Db2® Application Programming and SQL Guide.

CCN4366	Identifier "&1" with static storage duration was referenced in pragma &2 directive.

Explanation

A pragma is referring to an identifier that has static storage duration.

In the message text:

&1 is an identifier name, &2 is the name of the pragma.

User response

Remove the static storage class specifier in the identifier declaration or remove the pragma.

CCN4367	Only one operand can be defined in the asm statement.

Explanation

The asm statement is used to define the assembly data. Defining more that one operand in an asm statement is not allowed.

User response

Write the asm statement to define one operand.

CCN4368	The constraint '&1' cannot be used on the operand that has the "XL" constraint.

Explanation

The operand that has the constraint "XL" cannot have any other constraint.

In the message text:

&1 is the constraint.

User response

Remove all the constraints except "XL" and its parameter for the operand that is defined in the asm statement.

CCN4369	Only the output operand can be defined in the asm statement.

Explanation

The asm statement is used to define the assembly data. The defined operand cannot be listed in the input operand list.

User response

Move the operand that is defined in the asm statement to the output operand list.

CCN4370	The specified data size &1 is not valid. The default value &2 is assumed.

Explanation

Only a positive integer number can be used to specify the data size.

In the message text:

&1 is the data size. &2 is the default data size, currently 256.

User response

Specify the data size with an integer that is greater than 0.

CCN4371	The inline asm statement is only supported when the option GENASM is specified. The asm statement is ignored.

Explanation

The inline asm statement is only supported when the option GENASM is specified.

User response

Specify the option GENASM in order to use the inline asm statement.

CCN4372	No valid constraint is specified for the operand. The constraint '&1' is assumed.

Explanation

At least one valid constraint needs to be specified for the operand.

In the message text:

&1 is the constraint.

User response

Specify a valid constraint.

CCN4379	The specified register name '&1' is not a valid general purpose register name.

Explanation

Only a general purpose register name can be used.

In the message text:

&1 is the register name.

User response

Specify a valid general purpose register name.

CCN4380	There is no matching #pragma extension for "pop" argument in the same file. The pragma will be ignored.

Explanation

An attempt has been made to apply #pragma extension (pop) without a matching pragma extension in the same file.

User response

Remove the pragma.

CCN4381	There is a #pragma extension without a matching #pragma extension (pop) in the same file. The "pop" has been inserted by the compiler at the end of the file.

Explanation

A #pragma extension should always have a matching #pragma extension (pop).

User response

Insert a #pragma extension (pop) to match the #pragma extension.

CCN4382	pragma &1 ignored due to pragma &2.

Explanation

The pragma &1 cannot be embedded in code under pragma &2.

In the message text:

&1 and &2 are pragma names.

User response

Move pragma &1 out of the pragma &2 section or remove pragma &1.

CCN4383	The register "&1" specified for the variable "&2" is ignored. The compiler requires the GENASM option to parse this syntax.

Explanation

The register variable is only supported when the option GENASM is specified.

In the message text:

&1 is the register name. &2 is the variable name.

User response

Specify the option GENASM in order to use the register variable.

CCN4384	pragma &1 must be specified at file scope. The pragma is ignored.

Explanation

The pragma directive is ignored because it has been specified at an invalid scope.

In the message text:

&1 is the name of the pragma.

User response

Move the pragma directive to file scope.

CCN4391	A different common/nocommon was specified for "&1"; the new specification is used.

Explanation

The new common/nocommon specification overwrites the previous one.

In the message text:

&1 is an identifier.

User response

Remove the previous specification of attribute "common" or "nocommon".

CCN4392	An asm statement used in global scope can only define data.

Explanation

Only an asm statement that is used to define data can appear in global scope.

User response

Remove the asm statement in the global scope.

CCN4396	The attribute "&1" is only supported with "&2".

Explanation

"&2" must be specified to support the attribute "&1".

In the message text:

&1 is an attribute name. &2 is an option.

User response

Remove the attribute specifier or specify the "&2" option.

CCN4397	The AMODE for "&1" differs from the amode attribute specified on line "&2" of "&3".

Explanation

The AMODE for this function is not compatible with the amode attribute specified for the same function.

In the message text:

&1 is an identifier, &2 is a line number, and &3 is a file name.

User response

Remove the amode attribute specifier or change the compiler option to either LP64 or ILP32 to yield the same AMODE.

CCN4398	Using a NULL pointer constant in a relational operator with a __far pointer may not yield the correct result.

Explanation

A NULL pointer has no ordering with other pointers and therefore should not be used in a relational operator.

User response

If you want to test a pointer for NULL, use the equality operators, == or !=.

| CCN4399 | The __far pointer dereferencing operations can only be used under ARMODE. The operation is ignored. |

Explanation

The __far pointer operation requires access to addional data space. This access can only be provided by ARMODE. You can use either the ARMODE option or a function attribute to set ARMODE.

User response

Remove the __far pointer operation or specify ARMODE, either on the command line or by using function attributes.

| CCN4402 | The type "&1" is not supported on the target architecture. |

Explanation

The target architecture does not natively support machine instructions for the specified type.

In the message text:

"&1" is a type.

User response

Remove the reference to the type.

| CCN4403 | The aggregate cannot have a variable-sized member. |

Explanation

The aggregate must have a constant size. A variable-sized member is not permitted as a member of an aggregate.

User response

Remove the variable-sized member from the aggregate.

| CCN4404 | The "&1" option must be specified with the "&2" compiler option(s). |

Explanation

Given option(s) "&2", the compiler is enabling option "&1".

In the message text:

"&1" and "&2" are the names of compiler options.

User response

Specify option "&1".

| CCN4405 | The parameter in position &1 of the built-in function "&2" must hold an even value. |

Explanation

The parameter specified for the built-in function can not be an odd constant literal.

In the message text:

&1 is a number and &2 is a function name.

User response

Modify the argument's value to an even value.

| CCN4406 | The parameter in position &1 of the built-in function "&2" must hold an odd value. |

Explanation

The parameter specified for the built-in function can not be an even constant literal.

In the message text:

&1 is a number and &2 is a function name.

User response

Modify the argument's value to an odd value.

| CCN4407 | The pragma &1 must appear in function scope. The pragma is ignored. |

Explanation

The pragma can not appear outside function scope.

In the message text:

&1 is a pragma name.

User response

Place the pragma directive within a function scope or remove it.

| CCN4408 | Variable &1 in &2 is not a function argument. |

Explanation

The specified variable identifier must be a parameter of the enclosing function.

In the message text:

&1 is an identifier and &2 is a pragma name.

User response

Use a variable that is a function parameter.

CCN4409	Type of variable &1 in pragma &2 is not supported in this release.

Explanation

Type of the specified variable is not supported for this pragma.

In the message text:

&1 is a variable name and &2 is a pragma name.

User response

Use a variable that has a type which is supported by this pragma.

CCN4410	Argument &1 of &2 must be a compile-time constant.

Explanation

The specified argument must be a compile-time constant.

In the message text:

&1 is a argument number and &2 is a pragma name.

User response

Use a compile-time constant.

CCN4413	pragma &1 must appear before all explicit declarations and statements inside a function body.

Explanation

This pragma must appear after the opening brace of function definition but before any explicit declaration or statement.

In the message text:

&1 is a pragma name.

User response

Move the pragma inside the function body so that it appears before any explicit declaration or statement.

CCN4414	The modifier '&1' cannot be used on the last operand.

Explanation

The modifier '&1' can be used on any operand other than the last one.

In the message text:

&1 is the modifier in the asm statement.

User response

Remove the modifier '&1' from the last operand.

CCN4417	Initialization between types "&1" and "&2" is not allowed due to linkage mismatch.

Explanation

An attempt was made to initialize a variable with incompatible linkage.

In the message text:

&1 and &2 are both type names.

User response

Ensure types are compatible and have compatible linkage.

CCN4419	Pragma &1 must appear in global scope.

Explanation

This pragma can not be used in function or block scope.

In the message text:

&1 is a pragma name.

User response

Move the pragma statement to global scope.

CCN4425	The option "&1" only has an effect when preprocessed output is generated. The option is ignored.

Explanation

The specified option is ignored because it only has an effect when preprocessed output is generated.

In the message text:

"&1" is the ignored option.

User response

Specify an option configuration that generates preprocessed output.

CCN4426 **The "&1" qualifier cannot be applied to the variable "&2" because the variable does not have external linkage.**

Explanation

The specified qualifier requires that the variable has external linkage and not be defined within this compilation unit.

In the message text:

"&1" is a qualifier name. "&2" is a variable name.

User response

Remove the qualifier, or ensure the variable has external linkage and is not defined within this compilation unit.

CCN4427 **"&1" cannot be applied to the function or function pointer.**

Explanation

"&1" is invalid for function or function pointers.

In the message text:

"&1" can be a qualifier, type specifier, or attribute.

User response

Do not specify "&1" for function or function pointer.

CCN4428 **Operation between types "&1" and "&2" is not allowed due to linkage mismatch.**

Explanation

The operation specified is not valid between the operands having the given linkages.

In the message text:

&1 and &2 are both type names.

User response

Ensure types have compatible linkage.

CCN4430 **Compiling with the SQL compiler option resulted in the following message: &1**

Explanation

A general SQL related compiler message.

In the message text:

&1 is the SQL for Db2 Coprocessor message text.

User response

Refer to the SQL Reference.

CCN4431 **Compiling a CICS command resulted in the following message: &1**

Explanation

A message from the CICS translator for the CICS command.

In the message text:

&1 is the CICS Translator message text.

User response

Refer to the CICS Transaction Server for zOS Application Programming Reference.

CCN4436 **The linkage "&1" for function "&2" is invalid in &3 mode.**

Explanation

The linkage "&1" is not supported in &3 mode. The behaviour is undefined.

User response

Specify a valid &3 mode linkage for the function.

CCN4437 **The linkage "&1" is invalid when function "&2" is defined.**

Explanation

The linkage specified is not a supported linkage for a defined C function. The behaviour is undefined.

User response

Specify a valid linkage for a C function.

CCN4438 **Error when loading the SQL coprocessor. SQL coprocessor Return Code: &1.**

Explanation

The SQL coprocessor could not be loaded by the compiler.

In the message text:

&1 is the return code from the SQL coprocessor.

User response

Ensure that the SQL coprocessor resides in your STEPLIB. Refer to the Db2 for zOS Application Programming and SQL Guide

CCN4439	Unable to initilaize SQL coprocessor services. SQL coprocessor Return Code: &1.

Explanation

The SQL coprocessor could not be initialized.

In the message text:

&1 is the return code from the SQL coprocessor.

User response

Refer to the Db2 for zOS Application Programming and SQL Guide.

CCN4440	Host variable not found after SQL statement 'USING DESCRIPTOR'.

Explanation

A host variable is required ater the SQL statement 'USING DESCRIPTOR'.

User response

Specify a valid host variable prefaced with a colon character. Refer to the Db2 for zOS Application Programming and SQL Guide.

CCN4441	The SQL coprocessor cannot compile the SQL statement. SQL coprocessor Return Code: &1.

Explanation

The SQL coprocessor failed to compile the SQL statement.

In the message text:

&1 is the return code from the SQL coprocessor.

User response

Refer to the Db2 for zOS Application Programming and SQL Guide.

CCN4442	The SQL TYPE "&1" is not recognized by the compiler.

Explanation

The SQL TYPE specified is not supported or recognized by the compiler.

In the message text:

&1 is the type code for the SQL TYPE specified.

User response

Ensure that the version of the Db2 Coprocessor is supported by the compiler.

CCN4443	The compiler failed to terminate the SQL coprocessor. SQL coprocessor Return Code: &1

Explanation

The SQL coprocessor cannot be terminated.

In the message text:

&1 is the return code from the SQL coprocessor.

User response

Ensure that the SQL coprocessor is still available on your system.

CCN4444	Cannot retrieve a message from the SQL coprocessor. SQL coprocessor Return Code: &1

Explanation

The compiler cannot retrieve a message from the SQL coprocessor.

In the message text:

&1 is the return code from the SQL coprocessor.

User response

Refer to the Db2 for zOS Application Programming and SQL Guide.

CCN4445	The variable "&1" cannot be added as a host variable. SQL coprocessor Return Code: &2.

Explanation

The variable "&1" will not be recognized as a host variable by the SQL coprocessor.

In the message text:

&1 is a variable name. &2 is a return code from the SQL coprocessor.

User response

Refer to the Db2 for zOS Application Programming and SQL Guide for valid host variables.

CCN4446	The variable "&1" cannot be found as a host variable.

Explanation

The variable "&1" is not recognized as a host variable by the SQL coprocessor.

In the message text:

&1 is a variable name.

User response

Refer to the Db2 for zOS Application Programming and SQL Guide for valid host variables.

CCN4447	Unable to process the SQL statement.

Explanation

The SQL Coprocessor did not compile the SQL statement.

User response

Refer to the Db2 for zOS Application Programming and SQL Guide for valid SQL statements.

CCN4448	Static storage duration is assumed for non-file scope compound literals.

Explanation

In general, an object in non-file scope should have automatic storage duration.

User response

Use initialize list instead of compound literal.

CCN4453	The asm constraint "&1" cannot be used for the output operand.

Explanation

The asm operand constraint is not valid for the output operand.

In the message text:

&1 is the letter in the constraint set that is incompatible with the output operand.

User response

Remove the constraint or change it to a compatible constraint for the output operand.

CCN4454	Operand must be an lvalue.

Explanation

An lvalue is an expression that has address.

User response

Change the operand.

CCN4456	The component notation "%1$s" has an invalid number of components.

Explanation

The number of components must result in a scalar or vector type.

User response

Change the number of components.

CCN4457	The component notation "%1$s" is invalid.

Explanation

The number of components must consist of letters "xyzw" or numeric indices "0123456789abcdef".

User response

Change the component notation.

CCN4458	The component notation "%1$s" exceeds the "%2$s" elements of the vector "%3$s".

Explanation

The component specified cannot exceed the maximum number of vector elements.

User response

Do not access elements beyond the total element number of the vector.

CCN4459	Cannot assign to an object in the "%1$s" address space.

Explanation

The assignment is not allowed.

In the message text:

argument 1 is one of __constant, __global, __local, or __private

User response

Remove the assignment, or change the address space.

CCN4460	Pointer parameters for __kernel functions cannot point to objects in the __private address space.

Explanation

This function cannot have parameters that point to objects in the __private address space.

User response

remove the __private qualifier, or the __kernel function specifier.

CCN4461	Parameters to functions with the function specifier "%1$s" must be "%2$s" qualified.

Explanation

Parameters to this kind of function must have the specific qualifier.

User response

add the missing qualifier to the function parameter declaration.

CCN4474	Attribute malloc was specified on function "%1$s" which does not have a pointer return type; the attribute is ignored.

Explanation

Attribute malloc is valid only on functions that have a pointer return type.

In the message text:

"%1$s" is the function name.

User response

Remove the attribute or modify the function return type.

CCN4475	The attribute "&1" is not specified on function "&2" consistently.

Explanation

Once the attribute "&1" is specified on a function, it is required for the same function in the same compilation unit.

In the message text:

"&1" is an attribute name, "&2" is a function name.

User response

Add or remove the attribute "&1" to "&2", so the attribute "&1" is consistent on "&2".

CCN4478	Nested functions are not supported.

Explanation

Defining a function inside another function is not allowed.

User response

Move the nested function outside the enclosing function and modify it to be inline or external instead.

CCN4483	"&1" should not be declared _Noreturn.

Explanation

Although "&1" is not a keyword, it is a special function that cannot be declared with the specifier _Noreturn.

In the message text:

&1 is a function name.

User response

Remove the _Noreturn specifier from the declaration of "&1".

CCN4485	pragma &1 is ignored because it was already specified for this loop.

Explanation

The pragma reported in the message can not be specified more than onece for the same loop.

In the message text:

&1 is a pragma name.

User response

Specify this pragma only once for the same loop.

CCN4500	A struct or union type specifier applied to an unnamed member of a struct or union should not be tagged.

Explanation

A struct or union should not have an unnamed member with a tagged struct or union type specifier.

User response

Remove the tag.

CCN4501	There might be performance loss if "&1" has the "&2" visibility and is specified by "&3".

Explanation

There might be performance loss if any of the functions that are marked as imported resolve to statically bound objects.

In the message text:

&1 is the symbol name, &2 is one of the protected/ hidden/internal visibility attribute, and &3 is the option name.

User response

The generated code might be larger and run more slowly than the default code sequence generated for the functions.

CCN4502	The attribute "&1" is ignored.

Explanation

The attribute "&1" is for external linkage. The attribute is ignored.

User response

Remove the attribute specifier.

CCN4503	The attribute "&1" is re-declared with a different value.

Explanation

The attribute "&1" is re-declared with a different value.

User response

Remove the attribute specifier.

CCN4504	The type name "&1" in the generic association cannot be an incomplete type or variably modified type.

Explanation

The type of the generic association must be a complete type other than a variably modified type.

User response

Change the type name of the generic association.

CCN4505	Duplicate generic association with type "&1" is ignored.

Explanation

Duplicate generic association type specified.

User response

Change the type of the generic association.

CCN4506	Generic selection cannot contain more than one default generic association.

Explanation

Multiple default generic associations specified.

User response

Remove the extra generic associations.

CCN4507	Unable to find generic association of type "&1".

Explanation

The controlling expression shall have a type compatible with one of the generic associations.

User response

Change the controlling expression.

CCN4508	&1 is not compatible with &2. &3 is being set.

Explanation

The option is not compatible with another option so it is reset.

In the message text:

&1, &2 and &3 are option names.

User response

Remove one of the options.

CCN4509	The type of the controlling expression is also compatible with type "&1".

Explanation

The controlling expression can have type compatible with at most one of the types in the generic association list.

User response

Change the type of the generic association.

CCN4512	An atomic transaction or a body of a transaction_safe function cannot contain calls to unsafe statements or transaction_unsafe functions.

Explanation

An atomic transaction or a body of a function declared with the transaction_safe attribute must not contain calls to transaction_unsafe functions and other unsafe statements.

User response

Remove the unsafe statement or function.

CCN4513	A function type must not be both transaction-safe and transaction-unsafe.

Explanation

A function declaration, function pointer declaration, or typedef declaration for function pointer type must not specify both the transaction_safe and the transaction_unsafe attributes.

User response

Remove one of the conflict attributes.

CCN4514	The statement &1 is not allowed to transfer control into a transaction statement.

Explanation

The statement &1 must not be used to change the control flow to a transaction statement.

User response

Remove the statement &1.

CCN4515	The function &1 declared with the attribute &2 is not consistent with the attribute in its prior declaration.

Explanation

The function &1 declared in a compilation unit must have the same transaction attribute &2 in all of other compilation units.

User response

Modify the function &1 transaction attribute &2.

CCN4516	The conversion from transaction-unsafe function pointer &1 to transaction-safe function pointer &2 is not allowed.

Explanation

There is no conversion from transaction-unsafe function pointers to transaction-safe function pointers.

User response

Change one of the function pointers in the assignment to have the same type transaction attribute.

CCN4517	The &1 option can prevent the &2 option from detecting the use of some variables before they are set.

Explanation

The option &1 initializes automatic variables, as a result, it will hide uninitialized variables to be detected from the option &2.

User response

Remove one of the options.

CCN4518	The &1 keyword is not supported.

Explanation

The specified keyword is not supported.

In the message text:

&1 is a keyword.

User response

Remove the keyword or specify appropriate compiler option.

CCN4519	The debug option for assertion has no effect.

Explanation

The option has no effect because the compiler was built without assertions.

User response

Remove -qdebug=assert from the command line or switch to the compiler with assertions.

CCN4522	The value "&1" specified on pragma "&2" is not a valid architecture level. The pragma is ignored.

Explanation

A valid architecture level is required on this pragma.

In the message text:

"&1" is the architecture level. "&2" is the name of the pragma.

User response

Correct the architecture level or remove the pragma.

CCN4523	The value "&1" specified on pragma "&2" is lower than the effective architecture value "&3". The pragma is ignored.

Explanation

The nested architecture level must be specified in the increasing order.

In the message text:

"&1" is the invalid architecture level. "&2" is the name of the pragma. "&3" is the effective architecture level.

User response

Correct the architecture level by increasing its value or removing the pragma.

CCN4524	The use of vector type is invalid for architecture level "&1".

Explanation

The effective architecture value specified on either #pragma arch_section or a compiler option does not support vector.

In the message text:

"&1" is the architecture level specified on #pragma arch_section or a target architecture option.

User response

Use architecture which supports vector processing, or remove the use of vector.

CCN4525	The use of decimal floating-point is invalid for architecture level "&1".

Explanation

The effective architecture value specified on either #pragma arch_section or a compiler option does not support decimal floating-point.

In the message text:

"&1" is the architecture level specified on #pragma arch_section or a target architecture option.

User response

Use architecture value 7 and above, or remove the use of decimal floating-point.

CCN4526	The argument "&1" of the built-in function "&2" is invalid.

Explanation

The built-in function accepts only one of the predefined strings. See the documentation for this built-in function for valid arguments.

In the message text:

"&1" is the argument passed to the built-in function, "&2" is the built-in function name.

User response

Change the argument to one of the predefined strings.

CCN4527	Variable arguments macro &1 was invoked with an empty variable argument list.

Explanation

In a strict standards compliance mode, variable arguments macro requires at least one parameter in the variable argument list, but none were supplied.

In the message text:

&1 is a macro name.

User response

Specify at least one macro parameter in the variable argument list, or use an extended language level.

CCN4528	The function "%1$s" is deprecated.

Explanation

A deprecated function has been used.

In the message text:

"%1$s" is a deprecated function.

User response

Do not use any deprecated function.

CCN5001	A typedef must not have an initializer.

Explanation

A typedef represents a type, and a type must not have an initializer.

User response

Remove the initializer.

CCN5002	A typedef must not be specified on a function.

Explanation

A typedef represents a type and must not be specified on a function definition.

User response

Remove the typedef keyword.

CCN5003	A destructor must be a class member.

Explanation

A destructor is a special member function that cannot be declared outside a class declaration.

User response

Remove the destructor declaration or move it inside the class declaration.

CCN5004	A conversion operator must be a class member.

Explanation

A conversion operator is a special member function that converts an object of the class type to an object of the conversion type.

User response

Move the conversion operator declaration inside the class from which you want to convert.

CCN5005	"%1$s" must have "C" linkage.

Explanation

The main function "%1$s" cannot be specified with any linkage type other than extern "C".

In the message text:

"%1$s" is the string representing the main function.

User response

Remove the linkage specification or change it to extern "C".

CCN5006	The "%1$s" specifier must not be specified for an explicit template specialization.

Explanation

The "%1$s" specifier is not correct on an explicit template specialization.

In the message text:

"%1$s" is the invalid specifier.

User response

Remove the invalid specifier.

CCN5007	The "%1$s" specifier must not be specified for an explicit template instantiation.

Explanation

The "%1$s" specifier is not correct on an explicit template instantiation.

In the message text:

"%1$s" is the invalid specifier.

User response

Remove the invalid specifier.

CCN5008	An initializer is not allowed here.

Explanation

A function declaration cannot have an initializer.

User response

Remove the initializer.

CCN5009	A union must not have base classes.

Explanation

Only a struct or a class can have a base class.

User response

Change the union to a class or struct.

CCN5010	A name must not be used more than once within a template parameter list.

Explanation

Duplicated template parameter names are not allowed.

User response

Change the name of one of the template parameters.

CCN5011	"%1$s" is not a namespace.

Explanation

Only namespaces can be used in using directives, but the entity named is not a namespace.

In the message text:

"%1$s" is the name used in the source.

User response

Remove the using directive or change the name to be that of a namespace.

CCN5012	A using declaration for a member is allowed only in a class or struct.

Explanation

The using declaration is in a union, but using declarations are only allowed in classes and structs.

User response

Remove the using declaration.

CCN5013	"%1$s" is not a destructor.

Explanation

The name following the "~" must denote a destructor when it is used in a member list, but the name specified is not a destructor.

In the message text:

"%1$s" is the name in error.

User response

Change the name to be a destructor.

CCN5014	The literal type is unknown.

Explanation

The type of literal specified is not recognized.

User response

Change the literal to a recognized type.

CCN5015	A declaration that is "const" must have an initializer.

Explanation

The declaration has the const specifier so it must also have an initializer.

User response

Supply an initializer or remove the "const" specifier.

CCN5016	The expression must be an integral non-volatile constant expression.

Explanation

Only a constant expression can be used in this context, but a non-constant expression is specified.

User response

Change the non-constant expression to a constant expression.

CCN5017   **A class or struct declaration must have a class name, a declarator, or both.**

Explanation

Anonymous classes and structs are extensions to the language and may result in code that is not portable to other compilers.

User response

Name the class or add a declarator list.

CCN5018   **An enumeration must not be a template.**

Explanation

A template can only be a class, struct, or function.

User response

Remove the template keyword and template arguments, or nest the enumerator within a template.

CCN5019   **A typedef declaration must not be a template.**

Explanation

A template can only be a class, struct, or function.

User response

Remove the template keyword and template arguments, or nest the typedef within a template.

CCN5020   **A bit field must not have a "%1$s" specifier.**

Explanation

A bit field should have integral or enumeration type, and it should not be static.

In the message text:

"%1$s" is the specifier that is not valid for a bit field.

User response

Remove the incorrect specifier from the bit field or use an array rather than a bit field.

CCN5021   **The named class is not defined.**

Explanation

The class named in the elaboration is qualified but does not exist.

User response

Change the name to refer to a declared class.

CCN5022   **The named class is not a class name.**

Explanation

The name specified in the elaboration is not a class or struct.

User response

Change the name to be a class or struct, or remove the elaboration.

CCN5023   **The named struct is not defined.**

Explanation

The struct named in the elaboration is qualified but does not exist.

User response

Change the name to refer to a declared struct.

CCN5024   **Statements are not allowed within expressions.**

Explanation

The extension of having statements within expressions is not allowed.

User response

Remove the construct or set the appropriate options to allow this extension.

CCN5025   **The named union is not defined.**

Explanation

The union named in the elaboration is qualified but does not exist.

User response

Change the name to refer to a declared union.

CCN5026   **The named union is not a union name.**

Explanation

The name specified in the elaboration is not a union.

User response

Change the name to be a union.

CCN5027   **A function template must not be a qualifier.**

Explanation

Qualifiers can only be namespaces or classes.

User response

Correct the qualifier name or remove it.

CCN5028   **A qualified name is not allowed in the definition of "%1$s".**

Explanation

A name specified as a parameter, in a enumeration definition, or as an enumerator must not be a qualified name.

In the message text:

"%1$s" is the name in error.

User response

Remove the qualifiers from the name.

CCN5029   **The named enumeration is not defined.**

Explanation

Either the enumeration named in the elaboration is not defined or a forward declaration of an incorrect enumeration is being attempted.

User response

Change the name to be a defined enumeration or define the enumeration.

CCN5030   **The named enumeration is not an enumeration name.**

Explanation

The name specified in the elaboration is not an enumeration.

User response

Change the name to be an enumeration.

CCN5031   **A function template must not be the class referred to by a pointer-to-member.**

Explanation

Only classes can form pointer-to-members.

User response

Correct the class or remove the pointer-to-member.

CCN5032   **The destructor name is not valid.**

Explanation

A destructor name cannot be a qualifier.

User response

Change the name to be a destructor.

CCN5033   **A typedef declaration must declare a name.**

Explanation

A typedef declaration declares a type but no name is specified for the declaration.

User response

Add a name to the typedef declaration.

CCN5034   **The attributes are not attached to any type, function or variable. The attributes are ignored.**

Explanation

Type attributes must immediately follow the class, struct, or union keyword.

User response

Remove the attributes or move them to immediately after the class, struct, or union keyword.

CCN5035   **A simple namespace name is expected.**

Explanation

The name specified in a namespace declaration or a namespace alias cannot be qualified.

User response

Remove the qualifiers from the name.

CCN5036   **A namespace name is expected.**

Explanation

The name specified in the namespace alias declaration must refer to a namespace.

User response

Change the name to be a namespace name.

CCN5037   **A qualified name is expected in a using declaration.**

Explanation

An unqualified name has been specified in a using declaration. A using declaration must nominate a member of a namespace or class.

User response

Change the name to be a qualified name.

CCN5038   **The name "%1$s" is not a type.**

Explanation

The name is elaborated with "typename" but the name specified in the template instantiation is not a type.

In the message text:

"%1$s" is the name in error.

User response

Change the name to refer to a type in the instantiation.

CCN5039   **A label must be a simple identifier.**

Explanation

The label specified was a qualified name, but only unqualified names can be used for labels.

User response

Remove the qualifiers from the label.

CCN5040   **The text "%1$s" is unexpected. "%2$s" may be undeclared or ambiguous.**

Explanation

There is a syntax error in the declaration. It may be that a name that is expected to be a type is unknown or ambiguous.

In the message text:

"%1$s" is the symbol causing the syntax error. "%2$s" is the name that may be causing the error if it is expected to be a type.

User response

Remove the offending symbol or ensure that the name used as a type name is actually a type.

CCN5041   **A pointer-to-member must not be specified because "%1$s" is not a class.**

Explanation

The final qualifier in a pointer-to-member must be a class.

In the message text:

"%1$s" is the erroneous class type.

User response

Change the final qualifier to be a class.

CCN5042   **The value given for init_priority attribute must be a constant integral expression in the range between 101 and 65535. The attribute is ignored.**

Explanation

The attribute is ignored because the argument is not a constant integral expression in the range between 101 and 65535.

User response

Change the argument to evaluate to the required range.

CCN5043   **The explicit register specifier is unexpected. It is ignored.**

Explanation

An explicit register cannot be specified in this type of declaration.

User response

Remove the explicit register specifier.

CCN5044   **Only function declarations can have default arguments.**

Explanation

A default initializer has been specified in the parameter list of a function but the function is not being declared.

User response

Remove the default initializers.

CCN5045	The attribute "%1$s" has too many parameters. The attribute is ignored.

Explanation

The attribute is ignored because it has more comma-separated parameters specified than needed.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove offending parameters.

CCN5046	The attributes "%1$s" must not be specified for a parameter.

Explanation

It is not valid to specify the attribute "%1$s" for a function parameter or template parameter.

In the message text:

"%1$s" is the invalid specifier.

User response

Remove the specifier.

CCN5047	A template class declaration or definition must have a class name.

Explanation

Anonymous class templates are not allowed.

User response

Add a name.

CCN5048	The attribute "%1$s" is not a valid function parameter attribute. The attribute is ignored.

Explanation

The attribute is ignored because it cannot be specified for function parameters.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove the offending attribute.

CCN5049	A template function must not be explicitly specialized as a class.

Explanation

A template function can only be specialized as a function.

User response

Correct the specialization or the template.

CCN5050	A default template-argument should not be specified in a friend template declaration.

Explanation

A friend declaration cannot introduce a new default argument for a template parameter.

User response

Remove the default template-argument.

CCN5051	A template parameter must be a simple identifier.

Explanation

A template parameter is a type parameter or a parameter declaration.

User response

Correct the template parameter name.

CCN5052	The text "%1$s" is unexpected. The keyword "template" may need to prefix "%2$s".

Explanation

There is a syntax error in the declaration. It may be that the name is intended to be used as a template but does not have a template keyword.

In the message text:

"%1$s" is the symbol causing the syntax error. "%2$s" is the name used as a template but is not known to be a template.

User response

Add the template keyword or ensure that the name used as a template is actually a template.

CCN5053   **The declaration of a class member within the class definition must not be qualified.**

Explanation

A class member that is declared in the member list of a class must not be a qualified name.

User response

Remove the qualifier.

CCN5054   **A class or struct declaration must have a tag, a declarator, or both.**

Explanation

Anonymous classes and structs are extensions to the language, and the option allowing them is turned off.

User response

Name the class, add a declarator list, or use the appropriate language level option to allow anonymous structs.

CCN5055   **"%1$s" is specified more than once.**

Explanation

The specifier is used in the declaration more than once but the extra specifiers are ignored.

In the message text:

"%1$s" is the extra specifier.

User response

Remove the extra specifiers.

CCN5056   **Incorrect argument type specified for attribute "%1$s". The attribute is ignored.**

Explanation

The argument specified for the identified attribute has the wrong data type.

In the message text:

"%1$s" is the attribute name.

User response

Check the syntax rules for the specified attribute, and correct the argument.

CCN5057   **The declaration specifier is missing.**

Explanation

Implicit int types are no longer valid in C++.

User response

Add a complete type to the declaration or use the appropriate language level option to allow implicit int types.

CCN5058   **The declaration of a class member within the class definition must not be qualified.**

Explanation

A class member that is declared in the member list of a class must not be a qualified name.

User response

Remove the qualifier.

CCN5059   **The parameter of attribute "%1$s" is missing. The attribute is ignored.**

Explanation

The attribute "%1$s" has specific parameter type.

In the message text:

"%1$s" is the invalid attribute.

User response

Add the right parameter type to the attribute.

CCN5060   **An internal parser error has occurred: "%1$s".**

Explanation

The parser has detected an unrecoverable error.

In the message text:

"%1$s" is a description of the error.

User response

Report the problem to your IBM C++ service representative.

CCN5061   **This message is no longer used.**

Explanation

This message is an internal error caught in the C++ front end.

User response

Report the problem to your IBM C++ service representative.

CCN5062   **The incomplete class "%1$s" must not be used as a qualifier.**

Explanation

A class that is incomplete because it is only declared or because of some error in the declaration cannot be used as a qualifier.

In the message text:

"%1$s" is the incomplete class.

User response

Define the class.

CCN5063   **The text "%1$s" is unexpected.**

Explanation

A syntax error has occurred and the first unexpected token is "%1$s".

In the message text:

"%1$s" is the first invalid token.

User response

Change or remove the offending text.

CCN5064   **Syntax error: "%1$s" was expected but "%2$s" was found.**

Explanation

A syntax error has occurred and the first unexpected token is "%1$s". The only valid token at this point is "%2$s".

In the message text:

"%2$s" is the invalid text. "%1$s" is expected correct text.

User response

Change the incorrect token to the expected one.

CCN5065   **The qualifier "%1$s" is neither a class nor a namespace.**

Explanation

Only names representing classes and namespaces can be used as qualifiers.

In the message text:

"%1$s" is the invalid qualifier.

User response

Change the qualifier to a class name or namespace name.

CCN5066   **A function must not be defined in this scope.**

Explanation

Function definitions are only allowed in namespace scope or in a member list of a class.

User response

Move the definition into an appropriate scope.

CCN5067   **A return type must not be specified for "%1$s".**

Explanation

Return types cannot be specified for conversion functions.

In the message text:

"%1$s" is the function that cannot have a return type.

User response

Remove the return type.

CCN5068   **No member except a constructor can have the same name as its class, struct, or union.**

Explanation

An attempt was made to declare a member of a class that has the same name as the class itself.

User response

Change the name of the member.

CCN5069   **The bit field length must be greater than, or equal to, zero.**

Explanation

A bit field length must not be a negative number.

User response

Change the bit field length to zero or a positive number.

CCN5070   **The friend class declaration must use the "%1$s" keyword in the friend declaration of "%2$s".**

Explanation

The C++ language has changed. Now declarations of friend classes must contain an elaborated type specifier.

In the message text:

"%1$s" is the expected elaboration. "%2$s" is the offending text.

User response

Add the elaboration of class, struct, or union to the declaration.

CCN5071   **A class or union must not be defined in this context.**

Explanation

An attempt was made to define a class in a context where this is not valid.

User response

Move the definition to an appropriate context.

CCN5072   **The attribute "%1$s" is not supported on the target platform. The attribute is ignored.**

Explanation

The identified attribute specifier is not supported on the target platform and it is ignored.

In the message text:

"%1$s" is an attribute name.

User response

Remove the attribute specifier.

CCN5073   **A template specialization must not be declared here.**

Explanation

An explicit specialization can only be declared in namespace scope, either in the namespace in which the primary template is declared or, for a member template, in the namespace of which the enclosing class is declared.

User response

Remove the specialization or move it to a valid location.

CCN5074   **The "%1$s" specifier must not be specified for a friend.**

Explanation

The "%1$s" specifier is not correct on a friend declaration.

In the message text:

"%1$s" is the invalid specifier.

User response

Remove the invalid specifier.

CCN5075   **A static member function must not be virtual.**

Explanation

The virtual specifier must not be used on a member function that is declared static.

User response

Remove the virtual or static specifier.

CCN5076   **The pure-specifier (= 0) is not valid for a static member function.**

Explanation

The pure-specifier must not be used on a member function that is declared static.

User response

Remove the pure-specifier or static specifier.

CCN5077   **The array bound is too large.**

Explanation

The specified array bound is too large for the system to handle.

User response

Use a smaller array bound.

CCN5078	**A template must not be defined here.**

Explanation

A template can only be defined at namespace or class scope.

User response

Remove the template definition or move it to a valid location.

CCN5079	**The bit field length is too large.**

Explanation

The specified bit field length is larger than the system allows.

User response

Use a smaller bit field length.

CCN5080	**Template specializations must be prefixed with "template<>".**

Explanation

Old-style template specializations are accepted but are no longer compliant.

User response

Add the "template <>" syntax.

CCN5081	**The attribute "%1$s" is not a valid type attribute. The attribute is ignored.**

Explanation

The attribute is ignored because it is not a valid type attribute.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove the offending attribute.

CCN5082	**The attribute "%1$s" is not a valid variable attribute. The attribute is ignored.**

Explanation

The attribute is ignored because it does not apply to variables.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove offending attribute.

CCN5083	**An explicit template specialization must not be an untagged class.**

Explanation

An identifier is required for this declaration.

User response

Supply the identifier of the template that is being explicitly specialized.

CCN5084	**An explicit template instantiation must not be an untagged class.**

Explanation

An identifier is required for this declaration.

User response

Supply the identifier of the template that is being explicitly instantiated.

CCN5085	**The attribute "%1$s" is not a valid function attribute. The attribute is ignored.**

Explanation

The attribute is ignored because it does not apply to functions.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove offending attribute.

CCN5086	**The declaration of the template parameters is missing for template "%1$s".**

Explanation

A template must have at least one template parameter.

In the message text:

"%1$s" is the incorrect template declaration.

User response

Correct the template parameters or remove the invalid template declaration.

CCN5087	The arguments of the template qualifier do not match those of "%1$s".

Explanation

The types and order of template arguments must match the original template.

In the message text:

"%1$s" is the matching template declaration.

User response

Correct the arguments in the template qualifier.

CCN5088	An enumeration must not be defined in this context.

Explanation

An attempt is being made to define an enumeration in a context where it is not valid to define an enumeration.

User response

Move the definition to an appropriate context.

CCN5089	Too many template prefixes are specified for the declaration of "%1$s".

Explanation

The number of template scopes must match the template nesting level of the declaration.

In the message text:

"%1$s" is the incorrect declaration.

User response

Remove some of the template scopes.

CCN5090	Not enough template prefixes are specified for the declaration of "%1$s".

Explanation

The number of template scopes must match the template nesting level of the declaration.

In the message text:

"%1$s" is the incorrect declaration.

User response

Add the correct number of template scopes.

CCN5091	A function explicit instantiation must specify only "template instantiation-name".

Explanation

You cannot provide a definition or use the pure virtual specification on a function explicit instantiation.

User response

Correct the function explicit instantiation.

CCN5092	The explicit instantiation of "%1$s" was ignored because the function definition was not found.

Explanation

The function must be defined in the same translation unit as the explicit instantiation.

In the message text:

"%1$s" is the function being instantiated.

User response

Define the function in the same translation unit as the explicit instantiation

CCN5093	A partial specialization of a function is not allowed.

Explanation

Only class templates can be partially specialized.

User response

Remove the function partial specialization.

CCN5094	The template parameter must not be qualified.

Explanation

A template parameter defines the parameter to be a type in the scope of the template and therefore cannot be qualified.

User response

Remove all qualifiers.

CCN5095	The friend function declaration "%1$s" will cause an error when the enclosing template class is instantiated with arguments that declare a friend function that does not match an existing definition. The function declares only one function because it is not a template but the function type depends on one or more template parameters.

Explanation

This friend function makes use of one or more of the enclosing template's parameters. Therefore different instantiations of the template will create different friend functions. If a created friend function does not exist, the program will not link.

In the message text:

"%1$s" is the non-template friend declaration that depends on template parameters.

User response

Change the friend declaration to a template function (by adding explicit template arguments) or ensure that all instantiations will match an existing function.

CCN5096	No primary class template "%1$s" is found for a partial specialization.

Explanation

A primary class template must exist for a partial specialization.

In the message text:

"%1$s" is the incorrect class template partial specialization.

User response

Declare the primary template or remove the partial specialization.

CCN5098	The partial specialization "%1$s" must be declared in the same scope as the primary template or in a namespace scope that encloses the primary template.

Explanation

The primary template must be visible at the point the partial specialization is made.

In the message text:

"%1$s" is the incorrect class template partial specialization.

User response

Move the partial specialization into a correct scope.

CCN5099	The explicit specialization "%1$s" must be made in the nearest enclosing namespace of the template.

Explanation

The explicit specialization must be in the nearest enclosing namespace of the primary template or a namespace in the enclosing namespace set.

In the message text:

"%1$s" is the incorrect explicit specialization.

User response

Move the explicit specialization into a correct scope.

CCN5100	The class qualifier "%1$s" contains a circular reference back to "%2$s".

Explanation

The two classes contain references to each other that require each class to be defined before the other.

In the message text:

"%1$s" and "%2$s" are the classes with circular references.

User response

Change one of the classes so that it does not require the other class to be defined.

CCN5101	A typedef declaration must not contain the specifier "%1$s".

Explanation

A typedef defines another name to use in place of the declared type. The indicated specifier is not valid in this context.

In the message text:

"%1$s" is the invalid specifier.

User response

Remove the specifier.

CCN5102	A declaration with a "%1$s" specifier must contain a declarator ID.

Explanation

The type for the declaration contains a specifier that requires an object to be declared.

In the message text:

"%1$s" is the specifier in question.

User response

Remove the specifier or declare an object.

CCN5103	An anonymous union, struct or class declared at namespace scope must be declared static.

Explanation

Data members of an anonymous union, struct, or class declared at namespace scope have internal linkage so they must be declared static.

User response

Add the static specifier to the union, struct, or class.

CCN5104	The "%1$s" specifier must be applied only to objects declared in a block or to function parameters.

Explanation

The "%1$s" specifier has been used on a declaration that is not in an appropriate scope.

In the message text:

"%1$s" is the specifier in question.

User response

Remove the specifier.

CCN5105	Functions declared within a block must not be "%1$s".

Explanation

A function declared in a lexical block scope cannot have the "%1$s" specifier.

In the message text:

"%1$s" is the specifier in question.

User response

Remove the specifier.

CCN5106	The "static" specifier must be applied only to objects, functions, and anonymous unions, structs and classes.

Explanation

The "static" specifier has been applied to an inappropriate object.

User response

Remove the specifier.

CCN5107	The "extern" specifier must be applied only to objects and functions.

Explanation

The "extern" specifier cannot be applied to an out-of-line member variable or a type.

User response

Remove the "extern" specifier.

CCN5108	Class members must not be declared "extern".

Explanation

The "extern" specifier cannot be applied to an out-of-line member variable.

User response

Remove the "extern" specifier.

CCN5109	The "mutable" specifier must be applied only to non-reference class data members.

Explanation

The "mutable" specifier is being applied to a declaration that is not a member of a class or a member that is a reference.

User response

Remove the "mutable" specifier.

CCN5110	The "inline" specifier must be applied only to function declarations.

Explanation

The "inline" specifier is being applied to something other than a function.

User response

Remove the "inline" specifier.

CCN5111	The "explicit" specifier must be applied only to declarations of constructors within a class declaration.

Explanation

The "explicit" specifier is being applied to something other than a constructor that is being declared in-line in the class.

User response

Remove the "explicit" specifier.

CCN5112	The "virtual" specifier must be applied only to declarations of non-static class member functions within a class declaration.

Explanation

An attempt is being made to apply the "virtual" specifier inappropriately.

User response

Remove the "virtual" specifier from member functions using classes that are not static, or do not use it outside of a class.

CCN5113	The "static" specifier must be applied only to class member declarations within a class declaration.

Explanation

An attempt is being made to apply the "static" specifier inappropriately.

User response

Remove the "static" specifier.

CCN5114	A parameter name must not be the same as another parameter of this function.

Explanation

All parameter names for a given function must be unique.

User response

Give the parameter a unique name.

CCN5115	A member variable must have the "%1$s" attribute to be initialized in the definition of a class.

Explanation

Only constants that are also static may be initialized in the definition of a class.

In the message text:

"%1$s" is the missing specifier.

User response

Remove the initializer or ensure that the member is specified as both static and const.

CCN5116	A template declaration must declare a function, a class, a static member of a template class, or a template member of a class.

Explanation

An attempt is being made to create an invalid template.

User response

Change the declaration so it is not a template, or correct the template declaration.

CCN5117	Linkage specification must be at namespace scope.

Explanation

Linkage specifications are only valid for declarations at namespace scope.

User response

Remove the linkage specification.

CCN5118	A class name is expected in the base specifier.

Explanation

The name given in the base specifier is not a class.

User response

Remove the base specifier or change it to refer to a class.

CCN5119	A friend template must not be declared in a local class.

Explanation

A friend of a class defined in a lexical block must not be a template.

User response

Move the class to namespace scope or remove the friend declaration.

CCN5120	The out-of-line member definition "%1$s" of an explicit specialization should not use a template prefix.

Explanation

Out-of-line members of explicit specializations are defined in the same manner as members of non-template classes.

In the message text:

"%1$s" is the identifier of the out-of-line member.

User response

Remove the template prefix.

CCN5121	A template cannot have "C" linkage.

Explanation

Any linkage other than C++ is defined by implementation. The behavior with any linkage other than C++ is implementation-defined.

User response

Remove the "C" linkage.

CCN5122	The duplicate attribute "%1$s" is ignored.

Explanation

The attribute "%1$s" has been specified more than once.

In the message text:

"%1$s" is the duplicate attribute.

User response

Remove the extra attributes.

CCN5123	The operator symbol is not recognized.

Explanation

The operator symbol specified is not valid.

User response

Change the operator symbol to a valid symbol.

CCN5124	The text "typename" is unexpected because it cannot be used to modify a base specifier.

Explanation

A name specified in a base specifier list must be a type so typename is not required for template dependent names in a base specifier list.

User response

Remove the "typename" elaboration from the name.

CCN5125	The duplicate specifier "%1$s" is ignored.

Explanation

The specifier "%1$s" has been specified more than once.

In the message text:

"%1$s" is the duplicate specifier.

User response

Remove the extra specifiers.

CCN5126   **Taking the address of a label is not supported.**

Explanation

The gcc extension of taking the address of a label is not supported.

User response

Remove the "&&" from in front of the identifier.

CCN5127   **The text "typename" is unexpected because it cannot be used to modify a name in a constructor initializer list.**

Explanation

A name specified in a constructor initializer list must be a member or a base class so typename is not required for template dependent names in a constructor initializer list.

User response

Remove the "typename" elaboration from the name.

CCN5128   **"%1$s" is an ambiguous qualifier.**

Explanation

The qualifier "%1$s" is ambiguous since there is more than one name to which it resolves.

In the message text:

"%1$s" is the ambiguous qualifier.

User response

Add extra qualification to remove the ambiguity.

CCN5129   **The qualifier "%1$s" is not defined in the current scope.**

Explanation

The name being used as a qualifier has not been declared in a visible scope.

In the message text:

"%1$s" is the unknown qualifier.

User response

Change the qualifier to a name that has been declared.

CCN5130   **"%1$s" is not declared.**

Explanation

The name "%1$s" is not declared in any visible scope.

In the message text:

"%1$s" is the unknown name.

User response

Change the name to one that has been declared.

CCN5131   **Only one calling convention can be specified here.**

Explanation

More than one calling convention is being specified.

User response

Remove the extra calling conventions.

CCN5132   **The expression must be a constant expression.**

Explanation

A constant expression is expected but the expression specified is not a constant expression.

User response

Make the expression a constant expression.

CCN5133   **The attributes "%1$s" are not allowed.**

Explanation

The specifier or qualifier "%1$s" is incorrect in this type of declaration.

In the message text:

"%1$s" is the invalid attributes.

User response

Remove the invalid attributes.

CCN5134   **A function return type must not be a type definition. There may be a missing ";" after a "}".**

Explanation

An attempt has been made to define a class in the return type of a function. This is usually caused by a missing ";" after the class definition.

User response

Change the return type or ensure that a previous class definition has a ";" at the end of it.

| CCN5135 | The array bound cannot be zero. |

Explanation

An array cannot be declared with zero elements.

User response

Change the array bound.

| CCN5136 | A return type must not be specified for a constructor. |

Explanation

Constructors cannot have return types. A member or member function that has the same name as the class is considered a constructor, even if it is ill-formed.

User response

Remove the return type or rename the member.

| CCN5137 | The attribute "%1$s" is not allowed for a constructor. |

Explanation

A declaration of a constructor cannot have the "%1$s" attribute.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove the attribute.

| CCN5138 | The undefined template "%1$s" must not be explicitly instantiated. |

Explanation

An explicit instantiation requires a definition.

In the message text:

"%1$s" is the identifier of the undefined template.

User response

Define the template or remove the explicit instantiation.

| CCN5139 | In the context of the forward declaration, the name "%1$s" must not be qualified. |

Explanation

A qualified name cannot be used in a forward declaration for a class.

In the message text:

"%1$s" is the qualified name.

User response

Remove the qualifiers from the name.

| CCN5140 | The text "%1$s" is unexpected. "%2$s" may be undeclared, ambiguous, or may require "typename" qualification. |

Explanation

There is a syntax error in the declaration. A name may be expected to be a type that is unknown or ambiguous, or the type specified may be template-dependent and require typename qualification.

In the message text:

"%1$s" is the symbol causing the syntax error. "%2$s" is the name that may be causing the error if it is expected to be a type.

User response

Remove the offending symbol, ensure that the name used as a type name is actually a type, or add typename qualification to the type.

| CCN5141 | The declaration "%1$s" must not become a function because of a template argument. |

Explanation

Only a declaration that uses the syntactic form of a function can be a function.

In the message text:

"%1$s" is the declaration that is acquiring function type.

User response

Change the template argument, or change the declaration.

| CCN5142 | cv-qualifiers must not be added to a typedef of function type. |

Explanation

The const and volatile qualifiers cannot be specified on a type where a typedef that refers to a function is used.

User response

Remove the const or volatile specifiers.

CCN5143   **The qualifier "%1$s" is not a class.**

Explanation

A typedef that does not refer to a class is being used as a qualifier.

In the message text:

"%1$s" is the invalid qualifier.

User response

Change the qualifier to refer to a class.

CCN5144   **A non-local declaration is not allowed in a function body.**

Explanation

Only local declarations are allowed in a function body.

User response

Change the declaration to be a local declaration, or move it to the correct scope.

CCN5145   **The explicit instantiation "%1$s" of the class template does not match the primary template.**

Explanation

If the primary template is a union, the explicit instantiation must be a union as well. If the primary template is a class, the explicit instantiation must be a class.

In the message text:

"%1$s" is the explicit instantiation.

User response

Make sure that the class keys match.

CCN5147   **Friend declarations are allowed only in classes and structs.**

Explanation

Friends allow access to protected and private members. Because only classes and structs have members, only classes and structs can have friend declarations.

User response

Remove the friend declaration.

CCN5148   **A friend declaration must not be an explicit specialization.**

Explanation

An explicit specialization declaration must not be a friend declaration.

User response

Remove the friend or change it so it is not an explicit specialization.

CCN5149   **A template defined in an unnamed namespace must not be exported.**

Explanation

Exported namespace scope template definitions must be in a named namespace.

User response

Do not export the template, give the namespace a name, or move the template to another namespace scope.

CCN5150   **A using declaration must not specify a template-id.**

Explanation

You cannot specify a template ID in a using declaration.

User response

Remove or change the using declaration.

CCN5151   **A friend function that is qualified must not be defined.**

Explanation

Only friend functions without qualification can be defined in the friend declaration.

User response

Define the friend function in a different declaration.

CCN5152   **A template dependent name that is a type must be qualified with "typename".**

Explanation

The keyword "typename" is used to identify a name in a template as a type.

User response

Add the keyword typename.

CCN5153   **The attribute "%1$s" is ignored.**

Explanation

The identified attribute is not supported and it is ignored.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove the attribute specifier.

CCN5154   **A class, struct, or union must not be defined in a friend declaration.**

Explanation

Only functions can be defined in friend declarations.

User response

Define the friend in another declaration.

CCN5155   **A template parameter must not be used in an elaborated type specifier.**

Explanation

If the identifier in an elaborated type specifier resolves to a typedef or a template type parameter, it is ill-formed.

User response

Remove the construct.

CCN5156   **"%1$s" keyword is not supported on this platform. The keyword is ignored.**

Explanation

The keyword has no meaning for the current platform and is ignored.

In the message text:

"%1$s" is the ignored keyword.

User response

Remove the keyword for this platform.

CCN5157   **The text ">" is unexpected. It may be that this token was intended as a template argument list terminator but the name is not known to be a template.**

Explanation

An unexpected ">" was seen. This situation can arise when a template name is misspelled and is thus interpreted as a variable name rather than a template.

User response

Check that previous template names are correct.

CCN5158   **The attribute "%1$s" is not supported on the target platform. The attribute is ignored.**

Explanation

The identified attribute specifier is not supported on the target platform, and it is ignored.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove the attribute specifier.

CCN5159   **A storage class cannot be specified on a declaration directly contained in a linkage specification.**

Explanation

This declaration is contained within a linkage specification and therefore cannot have a storage class.

User response

Remove the storage class.

CCN5160   **"__thread" is not allowed on a class.**

Explanation

The "__thread" specifier cannot be used on a declaration for a class.

User response

Remove the "__thread" specifier.

CCN5161   **"%1$s" is already specified.**

Explanation

The name has already been specified.

In the message text:

"%1$s" is the name that has been already specified.

User response

Remove the duplicate name.

CCN5162   **"__thread" is not allowed on an enumeration.**

Explanation

The "__thread" specifier cannot be used in a declaration for an enumeration.

User response

Remove the "__thread" specifier.

CCN5163   **The array bound must not be negative.**

Explanation

An array cannot be declared with a negative number of elements.

User response

Change the array bound.

CCN5164   **The operator "%1$s" is ambiguous.**

Explanation

The specified operator is ambiguous because it can resolve to more than one declaration.

In the message text:

"%1$s" is the ambiguous operator.

User response

Add more qualifiers to resolve the ambiguity.

CCN5165   **Only a positive integral constant which is a power of 2 is allowed in the __align specifier.**

Explanation

The __align specifier must have a power of two since these are the only boundaries that align with memory.

User response

Change the integral constant to be a power of two.

CCN5166   **The __align specifier can only be applied to the definition of an aggregate tag or the declaration of a global or static variable.**

Explanation

The __align specifier has been applied to an inappropriate type of declaration.

User response

Remove the __align specifier.

CCN5167   **Only a positive integral constant which is a power of 2 is allowed in the aligned attribute specifier.**

Explanation

The aligned attribute must have a power of two since these are the only boundaries that align with memory.

User response

Change the integral constant to be a power of two.

CCN5168   **The specified alignment of "%1$s" exceeds the maximum supported value of "%2$s". The attribute is ignored.**

Explanation

An alignment value exceeded the maximum supported value. The alignment will be ignored.

In the message text:

"%1$s" is the specified alignment. "%2$s" is the maximum supported value for alignment.

User response

Use an alignment less than or equal to the maximum.

CCN5169 **__align specifier and __attribute__((aligned)) are both specified. Only the last one will be accepted.**

Explanation

Only a one of __align specifier or __attribute__((aligned)) will have an effect on alignment.

In the message text:

__align is a keyword __attribute__((aligned)) is a keyword

User response

Specify only one of __align specifier or __attribute__((aligned))

CCN5170 **Attribute "%1$s" is not supported for type specifications, and is ignored.**

Explanation

The specified attribute is not supported as a type attribute, and it is ignored.

In the message text:

"%1$s" is a type attribute name.

User response

Remove the type attribute.

CCN5171 **The value given for "%1$s" attribute is not a valid number. The attribute is ignored.**

Explanation

The attribute is ignored because the argument is not a valid number.

In the message text:

"%1$s" is an attribute name.

User response

Change the argument to evaluate to the required range.

CCN5172 **Arguments to be formatted must follow the format string argument.**

Explanation

The attribute is ignored because an incorrect argument value is specified.

User response

Change the argument to evaluate to the required range.

CCN5173 **"{" is expected.**

Explanation

An opening brace is expected for the function or member list.

User response

Add appropriate bracing.

CCN5174 **Arguments to be formatted cannot be specified for strftime formats.**

Explanation

The attribute is ignored because an incorrect argument value is specified.

User response

Change the argument to evaluate to the required range.

CCN5178 **An enumeration must not contain both a negative value and an unsigned value greater than LONGLONG_MAX.**

Explanation

An enumeration cannot contain both negative values and unsigned values greater than LONGLONG_MAX because they both cannot be represented by the same type.

User response

Remove the enumeration constant with the invalid value.

CCN5179 **The enumeration value is too large.**

Explanation

The enumeration value cannot be represented because it is too large for the underlying type.

User response

Remove the invalid enumeration value.

CCN5183	The explicit instantiation "%1$s" should be in a namespace enclosing the template and use a qualified name if not in the innermost such namespace.

Explanation

Where the template name is qualified, the explicit instantiation must be in an enclosing namespace of the primary template. If the template name is not qualified, the explicit instantiation must be in the nearest enclosing namespace of the primary template or a namespace in the enclosing namespace set.

In the message text:

"%1$s" is the explicit instantiation.

User response

Move the explicit instantiation to a correct scope or properly qualify it.

CCN5184	The "{" has no matching "}".

Explanation

There are not enough "}"s in the source so some construct is not complete.

User response

Add the appropriate number of "}"s.

CCN5185	The "%1$s" linkage specifier must only be applied to a function or a pointer to a function.

Explanation

The "%1$s" linkage specifier is being applied to something other than a function or pointer to function.

In the message text:

"%1$s" is the linkage specifier from the user's source code.

User response

Remove the linkage specifier.

CCN5186	A ";" or "," is expected following the initializer.

Explanation

An initializer was incomplete.

User response

Add ";" after the initializer.

CCN5187	The "(" has no matching ")".

Explanation

There is an imbalance of left and right parentheses.

User response

Ensure that each left parenthesis has a matching right parenthesis.

CCN5188	A ")" or "," is expected following the initializer.

Explanation

The initializer is not properly formed.

User response

Add the appropriate ending token to complete in the initializer.

CCN5189	Only static member variables of templates can be instantiated.

Explanation

A non-static data member of a template cannot be explicitly instantiated.

User response

Remove the explicit instantiation, or explicitly instantiate the class.

CCN5190	A "{" must follow a constructor initializer.

Explanation

A body for the constructor must follow the constructor initializer list.

User response

Add a body for the constructor.

CCN5191	A handler must be a compound statement.

Explanation

A catch handler must be a lexical block enclosed by "{" and "}".

User response

Add a well-formed catch handler.

CCN5192	A "{" must follow a base specifier list.

Explanation

Only class definitions can have a base specifier list. All class definitions must include a member list.

User response

Add a member list to the class definition.

CCN5193	A typedef name cannot be used in this context.

Explanation

Only actual class names, and not typedef names, can be used in elaborations.

User response

Replace the typedef name with the class it represents.

CCN5194	The "%1$s" declaration must declare a function.

Explanation

An operator or conversion function name in a declaration can only be used in a function declaration.

In the message text:

"%1$s" is the declaration from the user's source code.

User response

Change the name in the declaration.

CCN5195	The initializer has a syntax error.

Explanation

The initializer is not well-formed.

User response

Correct the syntax error in the initializer.

CCN5196	A friend declaration must not declare a partial specialization.

Explanation

The partial specialization of a template class cannot be declared in a friend declaration.

User response

Remove the friend declaration or change it from a partial specialization.

CCN5197	The "asm" keyword declaration is not supported.

Explanation

Inserting inline assembler instructions using the "asm" declaration is not supported. It is ignored.

User response

Remove the "asm" declaration.

CCN5198	The omitted keyword "private" is assumed for base class "%1$s".

Explanation

The access to the base class is not specified and is assumed to be private.

In the message text:

"%1$s" is the name of the base class which is assumed to be private.

User response

Add either "public," "protected," or "private" to the base class specifier.

CCN5199	An explicit instantiation must specify only a template class instantiation name.

Explanation

An explicit instantiation cannot contain a class definition. It must have a template argument list.

User response

Correct or remove the explicit instantiation.

CCN5200	The "%1$s" operator is not allowed between "%2$s" and "%3$s".

Explanation

The "%1$s" operator cannot be used between the two specified expressions because the operator is not defined for the types of the expression.

In the message text:

"%1$s" is the operator. "%2$s" and "%3$s" are the operands.

User response

Change the operator or one or both of the operands.

CCN5201	The "%1$s" operator is not allowed for type "%2$s".

Explanation

The "%1$s" operator cannot be used with the specified expression because the operator is not defined for the type of the expression.

In the message text:

"%1$s" is the operator. "%2$s" is the operand.

User response

Change the operator or the operand.

CCN5202	An expression of type "%1$s" is not allowed on the left side of "%2$s%3$s".

Explanation

The type of the expression on the left side of the operator is not correct.

In the message text:

"%2$s%3$s" are the operands. "%1$s" is the operator.

User response

Change the left operand.

CCN5203	The member expression ".%1$s" or "->%1$s" must be used with the function call operator ().

Explanation

The member expression refers to a member function so it must be used with the function call operator.

In the message text:

where "%1$s" is the name of the member function.

User response

Add the function call operator with the parameters required for the member function call.

CCN5204	An expression of type "%1$s" must not be followed by the function call operator ().

Explanation

Only functions can be followed by a function call operator ().

In the message text:

where "%1$s" is the type of the name referenced with the function call operator ().

User response

Remove the function call operator ().

CCN5205	An expression of type "%1$s" is not allowed where an rvalue is expected.

Explanation

The expression cannot be used in this situation since it has void type.

In the message text:

"%1$s" is the type of the expression.

User response

Change the expression.

CCN5206	An rvalue of type "%1$s" cannot be converted to an rvalue of type bool.

Explanation

There is no valid conversion sequence for converting the expression to an expression of type bool.

In the message text:

"%1$s" is the type of expression.

User response

Change the expression or provide a conversion sequence.

CCN5207	No common type found for operands with type "%1$s" and "%2$s".

Explanation

There is no standard conversion sequence between the two types.

In the message text:

"%1$s" and "%2$s" are the types of the operands.

User response

Define a conversion sequence between the two types.

CCN5208	The operand for "%1$s" is of type "%2$s" but a pointer-to-member type is required.

Explanation

The operator is expecting a pointer-to-member as an operand but the operand is of type "%2$s".

In the message text:

"%1$s" is the operator. "%2$s" is the unexpected type.

User response

Change the operand to be a pointer-to-member.

CCN5209	The result of this pointer-to-member operator must be the operand of the function call operator ().

Explanation

This expression is expected to be a function call.

User response

Change the expression to be a function call.

CCN5210	"%1$s" is not a base class of "%2$s".

Explanation

The class specified is not a base class, so the devirtualization or destructor name is not valid.

In the message text:

"%1$s" is the problematic class. "%2$s" is the expected derived class.

User response

Change the name to refer to a base class.

CCN5211	The array operator must have one operand that is a pointer to a

complete type and an operand that is of integral type.

Explanation

Either the variable is not an array or pointer or the index is not an integral type.

User response

Change the variable to be an array or pointer or the index to be an integer.

CCN5212	The operand of the "%1$s" operator must be an lvalue.

Explanation

The operator expects an object as its operand.

In the message text:

"%1$s" is the operator.

User response

Change the operand to be an object.

CCN5213	The local label "%1$s" is not defined.

Explanation

The label is declared but it is not defined.

In the message text:

"%1$s" is the label that is not defined.

User response

Create the label statement.

CCN5214	The conditional expression of a switch statement must be of integral or enumeration type.

Explanation

Integral types are all sizes of int and char as well as enumerations. A switch statement condition must have an integral type or something that can be converted to an integral type.

User response

Modify the switch condition or use an if statement instead of a switch.

CCN5215	The wrong number of arguments has been specified for "%1$s".

Explanation

When a function is called, the arguments are matched against the actual parameters in the function declaration. There must be the same number of arguments in the call as there are parameters in the declaration unless there are default arguments specified.

In the message text:

Where "%1$s" is the name of the function being called.

User response

Verify the function declaration and provide the correct number of arguments in your call.

CCN5216	An expression of type "%1$s" cannot be converted to type "%2$s".

Explanation

To convert between types, the compiler uses a set of specific rules defined in the C++ language. In this case the compiler was unable to convert between the specified types.

In the message text:

"%1$s" is the type being converted from. "%2$s" is the type being converted to.

User response

Modify the expression so that the conversion can be made, or define a conversion function to do the conversion.

CCN5217	"%1$s" is not a member of "%2$s".

Explanation

When using the . or -> operators to access a class member, the name after the operator must be a member of the class.

In the message text:

"%1$s" is the name of the member you are attempting to access. "%2$s" is the name of the class.

User response

Verify with the class declaration to see that you are accessing a member.

CCN5218	The call does not match any parameter list for "%1$s".

Explanation

The compiler will attempt to match the arguments in your function call against all functions defined with the name you are calling. It cannot match the number and types or arguments in your call with one of the declarations for the function.

In the message text:

"%1$s" is the name of the function.

User response

Check the declaration of the function you want to call and modify your arguments so that they match.

CCN5219	The call to "%1$s" has no best match.

Explanation

When a function is called, the compiler will check all the function declarations it has for the name you are calling. In this case, the compiler was unable to determine which one to call because there is not a single version that is a best match. The criteria for a best match is based on the types of the parameters and the conversions required to match them with the arguments in your call.

In the message text:

"%1$s" is the name of the function being called.

User response

Check the declarations for functions with that name and modify your arguments so that the correct one can be matched.

CCN5220	The address of a bit field cannot be taken.

Explanation

C++ language standards indicate that the & operator cannot be applied to bit fields.

User response

Change the bit field to an array or remove the line which attempts to take the address of the bit field.

CCN5221	The case expression must be an integral constant expression.

Explanation

Integral types are all sizes of int and char as well as enumerations. A case expression must be an integral

constant expression which is an expression which results in an integral type.

User response

Modify the expression so that it is an integral constant expression, or change the switch statement to an if statement.

CCN5222	The function must not have a return value.

Explanation

The function was declared with a return type of void, so it cannot have a return value specified.

User response

Remove the return value, or modify the function declaration to return the required type.

CCN5223	A return value of type "%1$s" is expected.

Explanation

The function was declared with a specific return type, so it should return a value of that type.

In the message text:

"%1$s" is the type of the expected return value.

User response

Modify the return type to match the declaration, or modify the declaration.

CCN5224	The type name "%1$s" is used where a variable or function name is expected.

Explanation

The expression was expected to be an object or function name but a type name was found.

In the message text:

"%1$s" is the type name.

User response

Replace the type with an object or function name.

CCN5225	The initializer list has too many initializers.

Explanation

An initializer list should not have more initializers than the number of elements to initialize.

User response

Remove some initializers or increase the number of elements to initialize.

CCN5226	The initializer must not be enclosed in braces.

Explanation

Only initializers for classes and arrays can have braces "{" and "}".

User response

Remove the braces.

CCN5227	"%1$s" cannot be initialized with an initializer list.

Explanation

The specified type cannot be initialized with an initializer list in braces "{" and "}".

In the message text:

"%1$s" is the type that cannot be initialized with an initializer list.

User response

Verify that the type is one that may be used with an initializer list. References cannot be initialized with an initializer list.

CCN5228	A "&" must precede the qualified member "%1$s" to form an expression with type pointer-to-member.

Explanation

A non-static member of a class was referred to with a qualified name, but no object is specified.

In the message text:

"%1$s" is the member.

User response

Refer to an object.

CCN5229	The best viable function "%1$s" uses an ambiguous conversion sequence.

Explanation

The overloaded function that has the closest match requires a conversion where one of the steps has more than one valid choice.

In the message text:

"%1$s" is the overloaded function.

User response

Provide a closer matching overload for the function being called.

CCN5230	The overloaded function name is not used in a valid context.

Explanation

It is not valid to use an overloaded function here.

User response

Use a non-overloaded function.

CCN5231	The array bound must be specified and must be a positive integral constant expression.

Explanation

Only the first array bound in a series of array bounds can be omitted when declaring a multi-dimensional array.

User response

Add the missing array bounds.

CCN5232	The implicit constructor for "%1$s" initializes a const member.

Explanation

The class contains a const member which must be initialized so a constructor must be provided.

In the message text:

"%1$s" is the class.

User response

Provide a constructor.

CCN5233	The implicit constructor for "%1$s" initializes a reference member.

Explanation

The class contains a reference member which must be initialized so a constructor must be provided.

In the message text:

"%1$s" is the class.

User response

Provide a constructor.

CCN5234	The implicit constructor for "%1$s" initializes a member of class type with an ill-formed constructor.

Explanation

The class contains a member of class type which does not have a default constructor so a constructor must be provided.

In the message text:

"%1$s" is the class.

User response

Provide a constructor.

CCN5235	The implicit constructor for "%1$s" initializes a base class with an ill-formed constructor.

Explanation

The class has a base class which does not have a default constructor so a constructor must be provided.

In the message text:

"%1$s" is the class.

User response

Provide a constructor.

CCN5236	The constructor initializer is unexpected. All bases and members have been initialized.

Explanation

The constructor initializer list has more elements being initialized than exist in the class. Either objects are initialized more than once or non-members are in the initializer list.

User response

Remove the extra initializers from the constructor initializer list.

CCN5237   **"%1$s" designates both a direct non-virtual base class and an inherited virtual base class.**

Explanation

The class is ambiguous because it refers to both a virtual base class and a non-virtual base class.

In the message text:

"%1$s" is the ambiguous base class name.

User response

Add qualifiers to make the name unambiguous.

CCN5238   **The data member "%1$s" cannot be initialized because there is no corresponding default constructor.**

Explanation

The data member was not in the constructor initializer list, but the type does not have a default constructor so the type cannot be constructed.

In the message text:

"%1$s" is the class member.

User response

Add the member to the constructor initializer list.

CCN5239   **The base class "%1$s" cannot be initialized because it does not have a default constructor.**

Explanation

The base class was not in the constructor initializer list. The type does not have a default constructor so the base class cannot be constructed.

In the message text:

"%1$s" is the base class.

User response

Add the base class to the constructor initializer list.

CCN5240   **A duplicate case value is not allowed.**

Explanation

The switch statement cannot choose a single case if there are duplicate case values.

User response

Remove or modify the duplicate case value.

CCN5241   **A "%1$s" statement is not allowed in this scope.**

Explanation

It is not valid to have this type of statement in this scope.

In the message text:

"%1$s" is the type of statement.

User response

Remove the statement.

CCN5242   **"goto %1$s" bypasses the initialization of "%2$s".**

Explanation

The goto statement skips over the initialization of an automatic variable.

In the message text:

"%1$s" is the label. "%2$s" is the missed variable.

User response

Move the label before the variable declaration.

CCN5243   **Label "%1$s" is already defined.**

Explanation

A label can only refer to one location in a function.

In the message text:

"%1$s" is the duplicate label.

User response

Rename the label.

CCN5244   **Label "%1$s" is not declared in this function.**

Explanation

Labels are only visible within the function in which they exist; either the label is not defined or it is in a different function than the goto.

In the message text:

"%1$s" is the missing label.

User response

Add the label to the function.

CCN5245	The switch statement already has a "default" statement.

Explanation

A switch statement may contain only one default statement.

User response

Remove the extra default statement.

CCN5246	The "%1$s" statement bypasses the initialization of "%2$s".

Explanation

A case in the switch statement contains automatic variables that are not contained within a compound statement.

In the message text:

"%1$s" is the case or default statement. "%2$s" is the bypassed variable.

User response

Add a pair of braces {} to enclose the code containing the automatic variable.

CCN5248	"%1$s" is not a class name.

Explanation

The name was expected to be a class name but it is not.

In the message text:

"%1$s" is the name.

User response

Change the name to be a class name.

CCN5249	Default arguments are not available due to other errors.

Explanation

This error is a cascade error. The default initializers cannot be used because of other errors.

User response

Fix the errors in the default initializers.

CCN5250	The keyword "this" is only allowed in a non-static class member function body or in a constructor member initializer.

Explanation

The "this" keyword has been used in the wrong context.

User response

Remove the "this" keyword.

CCN5251	The "%1$s" operator cannot be applied to the undefined class "%2$s".

Explanation

The use of the "%1$s" operator requires that the class that is being used as the operand be defined and not just declared.

In the message text:

"%1$s" is the operator. "%2$s" is the undefined class.

User response

Define the class.

CCN5252	"%1$s" contains a circular reference back to "%2$s".

Explanation

The two classes contain references to each other that require each class to be defined before the other.

In the message text:

"%1$s" and "%2$s" are the classes with circular references.

User response

Change one of the classes so that it does not require the other class to be defined.

CCN5253	This use of undefined class "%1$s" is not valid.

Explanation

The usage requires that the class be defined and not just declared.

In the message text:

"%1$s" is the class.

User response

Define the class.

CCN5254	The non-static member "%1$s" must be associated with an object or a pointer to an object.

Explanation

A member of a class has been referred to without an object but it is not a static member.

In the message text:

"%1$s" is the member.

User response

Specify an object.

CCN5255	The implicit member function "%1$s" cannot be defined.

Explanation

This is a cascading error. The implicit member function cannot be defined due to other errors in the class.

In the message text:

"%1$s" is the member function that cannot be defined.

User response

Fix the errors in the class.

CCN5256	A parameter of type "%2$s" cannot be initialized with an expression of type "%1$s".

Explanation

The type of the argument for the function does not match the type of the parameter.

In the message text:

"%2$s" is the parameter type. "%1$s" is the initialization expression type.

User response

Change the type of the parameter to match the expected type.

CCN5257	An object or reference of type "%2$s" cannot be initialized with an expression of type "%1$s".

Explanation

The type of the expression is not correct for initializing the object or reference.

In the message text:

"%2$s" is the object or reference type. "%1$s" is the initialization expression type.

User response

Change the type of the initializer.

CCN5258	A return value of type "%2$s" cannot be initialized with an expression of type "%1$s".

Explanation

The type of the expression in the return statement does not match the return type of the function.

In the message text:

"%2$s" is the return value type. "%1$s" is the initialization expression type.

User response

Change the type of the expression to the return type of the function.

CCN5259	The name lookups of "%1$s" do not yield the same type in the context of the expression and in the context of the class of the object expression.

Explanation

When a qualified name is specified in a member access, it is looked up in the context specified on the left side of the "." or "->" and in the context of the entire expression. It must resolve in only one of these lookups or it must resolve to the same declaration in both lookups.

In the message text:

"%1$s" is the name being looked up.

User response

Change the name.

CCN5260	A goto must not enter a try block or handler.

Explanation

A goto has been specified to a label that is in a try block or catch handler that does not also contain the goto statement.

User response

Change the label.

CCN5261	The header <typeinfo> must be included before using the typeid operator.

Explanation

The use of the typeid operator requires that the standard header <typeinfo> be included using a #include directive before it is used.

User response

Include the <typeinfo> header.

CCN5262	The first argument to the "offsetof" macro must be a class type.

Explanation

The "offsetof" macro can only be used with class types.

User response

Change the first argument to be a class type.

CCN5263	The non-const member function "%1$s" is called for "%2$s".

Explanation

Only const member functions can be called with a const object.

In the message text:

"%1$s" is the function. "%2$s" is the object.

User response

Change the member function to be const or change the object to be non-const.

CCN5264	The non-volatile member function "%1$s" is called for "%2$s".

Explanation

Only volatile member functions can be called with a volatile object.

In the message text:

"%1$s" is the function. "%2$s" is the object.

User response

Change the member function to be volatile or change the object to be non-volatile.

CCN5265	A pointer to non-const member function type "%1$s" is called for "%2$s".

Explanation

Only const member functions can be called with a const pointer-to-member.

In the message text:

"%1$s" is the function. "%2$s" is the type.

User response

Change the member function to be const or change the pointer-to-member to be const.

CCN5266	A pointer to non-volatile member function type "%1$s" is called for "%2$s".

Explanation

Only volatile member functions can be called with a volatile pointer-to-member.

In the message text:

"%1$s" is the function. "%2$s" is the type.

User response

Change the member function to be volatile or change the pointer-to-member to be volatile.

CCN5267	The second operand to the "offsetof" macro is not valid.

Explanation

The second operand of the "offsetof" macro is expected to be a member.

User response

Change the second operand to be a member.

CCN5268	"%1$s" has more than one default constructor.

Explanation

A class can only have one default constructor. A constructor with default initializers for all but the first parameter is considered a default constructor if all of the defaults are used.

In the message text:

"%1$s" is the class.

User response

Remove one of the default initializers or specify more arguments when calling the constructor.

CCN5269	"%1$s" has no default constructor.

Explanation

The class has no default constructor and one cannot be generated since the class contains objects that do not have default constructors.

In the message text:

"%1$s" is the class.

User response

Specify a default constructor.

CCN5270	An object of type "%2$s" cannot be constructed from an lvalue of type "%1$s".

Explanation

There is no constructor for the object that can be used for constructing the object.

In the message text:

"%2$s" and "%1$s" are the types of the target and the expression.

User response

Add an appropriate constructor or change the type.

CCN5271	"%1$s" is an ambiguous base class of "%2$s".

Explanation

The base class is ambiguous because the class has more than one base class with the same name.

In the message text:

"%1$s" is the base. "%2$s" is the class.

User response

Add qualifiers to uniquely specify the base class.

CCN5272	An array allocated by "new" cannot have an initializer.

Explanation

An initializer cannot be specified for an array that is allocated using new.

User response

Remove the initializer.

CCN5273	The array bound must have a positive value.

Explanation

An array cannot be declared with a negative number of elements.

User response

Change the array bound.

CCN5274	The name lookup for "%1$s" did not find a declaration.

Explanation

The name is not declared within this or an enclosing scope.

In the message text:

"%1$s" is the unresolved name.

User response

Declare the variable or change the name to a name in this or an enclosing scope.

CCN5275	The array boundary must have integral type or enumeration type.

Explanation

Only integral types can be used to specify an array bound.

User response

Change the array bound to be an integral type.

CCN5276	The local variable "%1$s" cannot be used in this context.

Explanation

A local variable cannot be used to specify default initializers for a function.

In the message text:

"%1$s" is the local variable.

User response

Remove the default initializers.

CCN5277	The local variable "%1$s" from function "%2$s" cannot be used in function "%3$s".

Explanation

A local variable from an enclosing function cannot be used in this context.

In the message text:

"%1$s" is the variable. "%2$s" is the enclosing function. "%3$s" is the current function.

User response

Remove the variable usage.

CCN5278	The reference variable "%1$s" must be initialized.

Explanation

All reference variables must be initialized but no initializer is specified.

In the message text:

"%1$s" is the reference variable.

User response

Specify an initializer.

CCN5279	The class member "%1$s" of type "%2$s" must be initialized in the initializer list of the constructor.

Explanation

The member must be initialized in the constructor initializer list.

In the message text:

"%1$s" is the member. "%2$s" is the class type.

User response

Add an initializer to the constructor initializer list.

CCN5280	The initializer is too long.

Explanation

The initializer for the array has too many initializers.

User response

Remove the extra initializers.

CCN5281	An expression of type "%1$s" cannot be modified.

Explanation

The expression on the left side of the assignment or reference parameter cannot be modified.

In the message text:

"%1$s" is the type that cannot be modified.

User response

Substitute an object that can be modified.

CCN5282	The const variable "%1$s" is uninitialized.

Explanation

All const variables must be initialized.

In the message text:

"%1$s" is the const variable.

User response

Initialize the variable.

CCN5283	"%1$s" is not a valid type for a function-style cast.

Explanation

Only simple type specifiers (built-in types and named types) can be used in a function-style cast.

In the message text:

"%1$s" is the type that is attempting to be cast to.

User response

Change the type of the cast.

CCN5284	The bit field "%1$s" cannot be bound to a non-const reference.

Explanation

A bit field can only be bound to a non-volatile const reference.

In the message text:

"%1$s" is the bit field.

User response

Change the reference type.

CCN5285   **The expression calls the undefined pure virtual function "%1$s".**

Explanation

Undefined pure virtual functions cannot be directly called.

In the message text:

"%1$s" is the function.

User response

Change the function being called.

CCN5286   **The unqualified member "%1$s" should be qualified with "%2$s::" and preceded by an "&" when forming an expression with type pointer-to-member.**

Explanation

A non-static member must be associated with an object.

In the message text:

"%1$s" is the member. "%2$s" are the qualifiers.

User response

Add the qualifiers and address operator.

CCN5287   **"offsetof" must not be applied to "%1$s". It is not a POD (plain old data) type.**

Explanation

"offsetof" cannot be applied to a class that is not a POD. POD types do not have non-static pointers-to-member, non-POD members, destructors, or copy assignment operators (that is, they are similar to C-style structs).

In the message text:

"%1$s" is the type.

User response

Change the type to be a POD type.

CCN5288   **The function template parameter of type "%2$s" cannot be initialized with an argument of type "%1$s".**

Explanation

The type of the argument is not appropriate for the type expected.

In the message text:

"%2$s" is the function template parameter type. "%1$s" erroneous argument specified.

User response

Change the type of the argument.

CCN5289   **The function template parameter "%1$s" has been found to have two types: type "%2$s" and type "%3$s".**

Explanation

Template argument deduction has arrived at two equally likely types for the same template type parameter.

In the message text:

"%1$s" is the template parameter. "%2$s" and "%3$s" are the two conflicting deduced types.

User response

Explicitly specify the template arguments.

CCN5290   **The function template parameter "%1$s" has been found to have two values: "%2$s" and "%3$s".**

Explanation

Template argument deduction has arrived at two equally likely values for the same non-type template parameter.

In the message text:

"%1$s" is the template parameter. "%2$s" and "%3$s" are the two conflicting deduced values.

User response

Explicitly specify the template arguments.

CCN5291   **The template argument for "%1$s" cannot be found.**

Explanation

Template argument deduction has failed. Either nothing matched or there was an ambiguity.

In the message text:

"%1$s" is the template parameter.

User response

Explicitly specify the template argument, or change the template.

CCN5292   **Jumping to a %1$s statement must not enter a try block or handler.**

Explanation

A case or default statement has been specified in a try block or catch handler that does not also contain the enclosing switch statement.

In the message text:

"%1$s" is the label.

User response

Change the location of case or default statement.

CCN5293   **The argument to va_start must be a parameter name.**

Explanation

A non-parameter has been specified to va_start.

User response

Change the argument to a parameter name.

CCN5294   **An object or reference of type "%2$s" cannot be initialized with an rvalue of type "%1$s".**

Explanation

This object or reference must be initialized with an object.

In the message text:

"%2$s" is the type of the object. "%1$s" is the type of the rvalue.

User response

Change the type of the object or reference.

CCN5295   **A parameter of type "%2$s" cannot be initialized with an rvalue of type "%1$s".**

Explanation

This parameter must be initialized with an object.

In the message text:

"%2$s" is the type of the parameter. "%1$s" is the type of the rvalue.

User response

Change the type of the parameter.

CCN5296   **A return value of type "%2$s" cannot be initialized with an rvalue of type "%1$s".**

Explanation

The return value must be initialized with an object.

In the message text:

"%2$s" is the return type. "%1$s" is the type of the rvalue.

User response

Change the return type.

CCN5297   **The call to "%1$s" resolves to a function for which multiple default arguments for a given parameter have been specified.**

Explanation

Function overload resolution has failed. The best match function has been declared in different namespaces with conflicting default arguments.

In the message text:

"%1$s" is the function call. "%2$s" is the declaration of the best match function.

User response

Two declarations of "%2$s" in different namespaces have specified default arguments for a given parameter. Only one such declaration may be visible at a point of call using default arguments.

CCN5298 Template argument deduction cannot be performed using the function "%1$s".

Explanation

Argument deduction can only be performed with a function if the set of overloaded functions does not contain a template function.

In the message text:

"%1$s" is the name of the function.

User response

Explicitly specify the template argument or change the template.

CCN5299 The "%1$s" operator cannot be applied to a pointer to incomplete type: "%2$s".

Explanation

The "%1$s" operator requires that the type of its operand be defined and not just declared.

In the message text:

"%1$s" is the operator. "%2$s" is the incomplete type.

User response

Define the type of the operand.

CCN5300 The "private" member "%1$s" cannot be accessed.

Explanation

The member is declared in a private section of the class and cannot be accessed.

In the message text:

"%1$s" is the member.

User response

Change the access of the member.

CCN5301 The "protected" member "%1$s" cannot be accessed.

Explanation

The member is declared in a protected section of the class and cannot be accessed.

In the message text:

"%1$s" is the member.

User response

Change the access of the member or remove the reference.

CCN5302 "%1$s" is a "private" base class of "%2$s".

Explanation

The base class is private and cannot be accessed.

In the message text:

"%1$s" is the base class. "%2$s" is the derived class.

User response

Change the access of the base class.

CCN5303 "%1$s" is a "protected" base class of "%2$s".

Explanation

The base class is protected and cannot be accessed.

In the message text:

"%1$s" is the base class. "%2$s" is the derived class.

User response

Change the access of the base class.

CCN5304 The "private" copy constructor "%1$s" cannot be accessed to create a temporary object.

Explanation

The creation of a temporary object requires access to the copy constructor, but the copy constructor is private.

In the message text:

"%1$s" is the copy constructor.

User response

Change the access of the copy constructor.

CCN5305 The "protected" copy constructor "%1$s" cannot be accessed to create a temporary object.

Explanation

The creation of a temporary object requires access to the copy constructor, but the copy constructor is protected.

In the message text:

"%1$s" is the copy constructor.

User response

Change the access of the copy constructor.

CCN5306   **The "private" copy constructor "%1$s" cannot be accessed.**

Explanation

Access to the copy constructor is required but the copy constructor is private.

In the message text:

"%1$s" is the copy constructor.

User response

Change the access of the copy constructor.

CCN5307   **The "protected" copy constructor "%1$s" cannot be accessed.**

Explanation

Access to the copy constructor is required but the copy constructor is protected.

In the message text:

"%1$s" is the copy constructor.

User response

Change the access of the copy constructor.

CCN5308   **The semantics specify that a temporary object must be constructed.**

Explanation

Informational message indicating that the semantics of the language require a temporary object to be constructed.

User response

See the primary message.

CCN5309   **The temporary is not constructed, but the copy constructor must be accessible.**

Explanation

Informational message that the temporary is not constructed as an optimization but the language semantics require that the copy constructor be accessible.

User response

See the primary message.

CCN5310   **The assignment-style initialization of an object of type "%1$s" with an expression of type "%2$s" requires access to the copy constructor.**

Explanation

An assignment-style initialization requires access to the copy constructor, but the parentheses-style initialization does not.

In the message text:

"%1$s" is the type of the object. "%2$s" is the type of the expression.

User response

Make the assignment operator a friend of the class or use parenthesis-style initialization.

CCN5311   **Access to the copy constructor is not required if parentheses-style initialization is used.**

Explanation

An assignment-style initialization requires access to the copy constructor, but the parentheses-style initialization does not.

User response

Make the assignment operator a friend of the class or use parenthesis-style initialization.

CCN5312   **"%1$s" is a "private" base class of "%2$s". Injected-class-name "%1$s" is inaccessible.**

Explanation

The base class is private and cannot be accessed.

In the message text:

"%1$s" is the base class. "%2$s" is the derived class.

User response

Change the access of the base class.

CCN5400   **"%1$s" has a conflicting declaration.**

Explanation

The specified name has already been given a different declaration.

In the message text:

"%1$s" is the name which has a conflicting declaration

User response

Change the name for this declaration, or use the existing declaration.

CCN5401	The member "%1$s" is already declared.

Explanation

The member name has already been used in this class. The compiler cannot tell the difference between two members with the same name unless they are both member functions with different parameters.

In the message text:

"%1$s" is the name of the member.

User response

Change the name of the member, or use the existing declaration. If the member name is a member function, modify the parameters to overload the function.

CCN5402	The non-static member "%1$s" must not be defined outside of the class definition.

Explanation

Only static members can have a definition outside of the class definition. Non-static members only exist when a object is created from the class.

In the message text:

"%1$s" is the member.

User response

Move the definition of the member inside the class constructor or make the member static.

CCN5403	"%1$s" is already defined.

Explanation

The specified name has already been defined in another location.

In the message text:

"%1$s" is the name which has already been defined.

User response

Remove one of the definitions for this name, or use another name.

CCN5404	The out-of-line member function declaration for "%1$s" must have a body.

Explanation

A member function must be declared inside its class and may be defined either inside its class or outside its class. It may not be redeclared outside its class.

In the message text:

"%1$s" is the name of the member function.

User response

Add the definition for the body of this function.

CCN5405	The default arguments for "%1$s" must not be redefined.

Explanation

If there is more than one declaration for the specified function, the default arguments should be given the same values in both.

In the message text:

"%1$s" is the name of the function.

User response

Remove the duplicate declaration, or change the default arguments so that they match.

CCN5406	The namespace alias "%1$s" is already defined.

Explanation

A namespace alias in a declarative region can only be redefined to denote the same namespace.

In the message text:

"%1$s" is the namespace alias.

User response

Remove or change the namespace alias.

CCN5407	The base class "%1$s" contains a circular reference back to "%2$s".

Explanation

A reference in the base class requires that the derived class be complete. There is no way to complete both classes.

In the message text:

"%1$s" and "%2$s" are the names of the conflicting classes.

User response

Change one of the classes to remove the circularity.

CCN5408	The base class "%1$s" is declared but not defined.

Explanation

A base class must be a complete class.

In the message text:

"%1$s" is the name of the base class.

User response

Define the base class before it is used in a base specifier list.

CCN5409	"%1$s" must not be used more than once in the list of base classes.

Explanation

Listing the same class twice or more in a base specifier list is not allowed.

In the message text:

"%1$s" is the name of the duplicate base class.

User response

Remove the duplicate base class.

CCN5410	The direct base "%1$s" of class "%2$s" is ignored because "%1$s" is also an indirect base of "%2$s".

Explanation

The base class has been specified directly as well as indirectly.

In the message text:

"%1$s" is the name of the base class. "%2$s" is the name of the derived class.

User response

No response required, but the redundant base class can be removed.

CCN5411	The default arguments for "%1$s" are in error.

Explanation

A default template argument cannot refer to the template parameter.

In the message text:

"%1$s" is the template parameter declaration.

User response

Correct the default arguments.

CCN5412	The union "%1$s" cannot be used as a base class.

Explanation

A union must not have, or be used as a base class.

In the message text:

"%1$s" is the name of the union.

User response

Remove the union base specifier or change it to a class.

CCN5413	"%1$s" is already declared with a different access.

Explanation

A member declaration must have only one access.

In the message text:

"%1$s" is the name of the member.

User response

Remove the offending declaration or declare it with the same access.

CCN5414	"%1$s" is declared differently in the body of function "%2$s".

Explanation

The specified local name has already been given a different declaration.

In the message text:

"%1$s" is the duplicate local declaration. "%2$s" is the function containing it.

User response

Change the name for this declaration, or remove the conflicting duplicate declaration.

CCN5415	"%1$s" is already declared with default template arguments.

Explanation

A template parameter may not be given default arguments in two different declarations.

In the message text:

"%1$s" is the name of the template parameter.

User response

Remove the default argument on one of the declarations.

CCN5416	"%1$s" cannot be declared because its name has already been used.

Explanation

A member can only be declared once in a class.

In the message text:

"%1$s" is the member name.

User response

Change or remove one of the uses.

CCN5417	The qualified id-declarator "%1$s" cannot refer to a name introduced by a using declaration.

Explanation

The qualified ID collides with a name in a using declaration.

In the message text:

"%1$s" is the qualified ID.

User response

Change the declaration or remove the using declaration.

CCN5418	The definition of "%1$s" cannot contain an initializer because the initializer was specified in the class definition.

Explanation

The out-of-line definition of a static data member can only have an initializer when there is no initializer on the declaration in the class.

In the message text:

"%1$s" is the data member.

User response

Remove one of the initializers.

CCN5419	An exception-specification must be specified as "%1$s" to match the implicit declaration.

Explanation

All declarations of a function including definitions and explicit specializations must have either no exception specification or the same set of types listed in their exception specifications.

In the message text:

"%1$s" is the exception specification.

User response

Correct the exception specification.

CCN5420	"%1$s" is declared differently than the implicit declaration "%2$s".

Explanation

A duplicate declaration of an implicit declaration is in error.

In the message text:

"%1$s" is the declaration. "%2$s" is the implicit declaration.

User response

Correct or remove the declaration.

CCN5421	"%1$s" is declared differently than the internally generated declaration "%2$s".

Explanation

A duplicate declaration of an internal declaration is in error.

In the message text:

"%1$s" is the declaration. "%2$s" is the internally generated declaration.

User response

Correct or remove the declaration.

CCN5422	"%1$s" cannot be declared before "%2$s", and "%2$s" cannot be declared before "%1$s".

Explanation

Each of the two declarations is coded so that it requires the other declaration first.

In the message text:

"%1$s" and"%2$s" are the two declarations.

User response

Change the dependence between the two declarations.

CCN5423	The new declaration "%1$s" cannot be added.

Explanation

The IDE is browsing and can't add a new declaration to the code store.

In the message text:

"%1$s" is the declaration.

User response

Reincorporate with the changed source.

CCN5424	"%1$s" is declared on line %3$s of "%2$s".

Explanation

An informational message giving the location of a declaration.

In the message text:

"%1$s" is the declaration. %3$s is the line number. "%2$s" is the source.

User response

See the primary message.

CCN5425	"%1$s" is defined on line %3$s of "%2$s".

Explanation

An informational message giving the location of a definition.

In the message text:

"%1$s" is the declaration. %3$s is the line number. "%2$s" is the source.

User response

See the primary message.

CCN5426	The name "%1$s" is used on line %3$s of "%2$s".

Explanation

An informational message giving the location of the use of a name.

In the message text:

"%1$s" is the name. %3$s is the line number. "%2$s" is the source.

User response

See the primary message.

CCN5427	The using declaration introduces "%1$s" in conflict with a declaration in this scope.

Explanation

A using declaration is a declaration, so the restrictions on declaring the same name twice in the same region apply.

In the message text:

"%1$s" is the declaration in conflict.

User response

Remove the using declaration or remove the conflicting declaration.

CCN5428	The using declaration "%1$s" must not introduce a name into its own scope.

Explanation

A using declaration is a declaration, so the restrictions on declaring the same name twice in the same region apply.

In the message text:

"%1$s" is the using declaration.

User response

Remove or change the using declaration.

CCN5429   **"%1$s" must not be repeated at block scope.**

Explanation

A using declaration is a declaration, so the restrictions on declaring the same name twice in the same region apply (a variable at lexical block scope in this case).

In the message text:

"%1$s" is the using declaration.

User response

Remove the repeated using declaration.

CCN5430   **The out-of-line member declaration for "%1$s" must be in a namespace scope that encloses the class definition.**

Explanation

The class definition cannot be seen in the scope that the out-of-line member declaration exists.

In the message text:

"%1$s" is the out-of-line member declaration.

User response

Move the out-of-line member declaration into the same scope as its class definition or a scope that encloses its class definition.

CCN5431   **The declarator cannot be qualified with the enclosing namespace "%1$s".**

Explanation

A nested-name-specifier cannot name any of the namespaces that enclose the member's definition.

In the message text:

"%1$s" is the namespace declaration.

User response

Remove the qualifiers.

CCN5432   **The qualified declarator "%1$s" must refer to an existing declaration.**

Explanation

When the declarator-id is qualified, the declaration has to refer to a previously declared member of a class or namespace and the member cannot have been introduced by a using declaration already.

In the message text:

"%1$s" is the qualified declarator.

User response

Remove the qualified ID, or add it to the class or namespace.

CCN5433   **The explicitly specialized template class member "%1$s" cannot be defined unless the template class is specialized.**

Explanation

An out-of-line class member definition can only be made for an existing class. A class template explicit specialization is a separate class with different members from the primary template.

In the message text:

"%1$s" is the explicitly specialized template class member.

User response

Write the class template explicit specialization or remove this declaration.

CCN5434   **The friend function must also be declared in the enclosing block scope.**

Explanation

If a friend declaration appears in a local class and the name specified is an unqualified name, a prior declaration is looked up without considering scopes that are outside the innermost enclosing non-class scope. For a friend function declaration, if there is no prior declaration, the program is ill-formed.

User response

Remove the local friend function or add the declaration to the enclosing block scope.

CCN5435   **The template "%1$s" must not be explicitly specialized more than once with the same set of template arguments.**

Explanation

This is a violation of the one definition rule.

In the message text:

"%1$s" is the template.

User response

Remove the duplicate explicit specialization.

CCN5436   **The template "%1$s" must not be explicitly instantiated more than once with the same set of template arguments.**

Explanation

Only one explicit instantiation of a template with the same set of arguments is allowed in a program.

In the message text:

"%1$s" is the template.

User response

Remove the duplicate explicit instantiation.

CCN5437   **The template "%1$s" must not be explicitly specialized after explicit instantiation with the same set of template arguments.**

Explanation

A program cannot have explicit specialization after explicit instantiation of a template with the same set of arguments.

In the message text:

"%1$s" is the template.

User response

Remove either the explicit specialization or the explicit instantiation or change the order.

CCN5438   **The template parameter "%1$s" must not be redeclared.**

Explanation

A template parameter can be declared at most once in a template parameter list.

In the message text:

"%1$s" is the template parameter.

User response

Remove or change the template parameter.

CCN5439   **The template parameters "%1$s" do not match the parameters for the previous declaration for "%2$s".**

Explanation

A redeclaration of a template must agree in the number and type of the template parameters.

In the message text:

"%1$s" and "%2$s" are the template parameters.

User response

Correct the template parameters.

CCN5440   **"%1$s" may have different pass-by-value semantics.**

Explanation

If you are linking with code from some older compilers, they may use a different set of rules to decide whether a class is passed by value.

In the message text:

"%1$s" is the declaration. %3$s is the line number. "%2$s" is the source.

User response

Use pragma pass_by_value or, if available, the compiler option which changes pass-by-value semantics.

CCN5441   **The init_priority attribute can only be used in objects of class type in namespace scope. The attribute is ignored.**

Explanation

The attribute is ignored because it is not attached to an object in namespace scope.

User response

Remove offending attribute.

CCN5442   **Priority values in successive init_priority attribute specifiers and pragma priority directives must increase.**

Explanation

Last init_priority or pragma priority has a lower or equal priority value than the current one.

User response

Check previous init_priority attribute or pragma priority value and make sure that it has a higher priority than current specification.

CCN5443	The specified function has already been given a different linkage.

Explanation

Two or more declarations for a function must have matching language linkages if the linkages are specified.

User response

Ensure that the specified language linkages match.

CCN5444	The function can only have C++ language linkage specified because the function has already been given C++ linkage.

Explanation

A previous declaration did not have a language linkage specification so the only valid language linkage specification is C++.

User response

Ensure that the specified language linkages match.

CCN5445	The mapped name "%1$s" for the declaration "%2$s" conflicts with the existing mapped name "%3$s".

Explanation

An asm label or pragma map conflicts with an existing asm label or pragma map.

In the message text:

"%1$s" is the new name mapping. "%2$s" is the declaration being mapped. "%3$s" is the previous name mapping.

User response

Remove the asm label specifier or pragma map.

CCN5446	The template "%1$s" has already been explicitly specialized with

the same arguments. The explicit instantiation has no effect.

Explanation

A explicit instantiation after an explicit specialization with the same arguments is skipped by the compiler as it causes no additional instantiation to take place.

In the message text:

"%1$s" is the template.

User response

This explicit instantiation has no effect and may be removed without changing the program.

CCN5447	The entity "%1$s" is subject to a prior explicit instantiation definition. The prior instantiation is not suppressed.

Explanation

The attempt to suppress instantiation only affects implicit instantiation; an explicit instantiation already occurred.

In the message text:

"%1$s" is the entity for which there was an attempt to suppress instantiation.

User response

To retain the current behaviour, remove the attempt to suppress instantiation.

CCN5448	The template specialization "%1$s" has internal linkage. The explicit instantiation declaration is ignored.

Explanation

The C++ standard prohibits naming template specializations with internal linkage in an explicit instantiation declaration.

In the message text:

"%1$s" is the entity named in the explicit instantiation declaration

User response

To retain the current behaviour, remove the explicit instantiation declaration.

CCN5449	"inline" used on definition of namespace "std".

Explanation

The namespace "std" is used by the C++ standard library and should not be inline.

User response

Remove the inline keyword.

CCN5450	"inline" used on definition of namespace "%1$s", which is not inline in its first definition.

Explanation

A namespace cannot be made inline after its first definition.

In the message text:

"%1$s" is the namespace.

User response

Remove the inline keyword or make the namespace inline on its first definition.

CCN5500	The configuration file "%1$s" cannot be opened: %2$s.

Explanation

The configuration file could not be opened.

In the message text:

"%1$s" is the name of the configuration file that could not be opened. "%2$s" is the string returned by the operating system when the file open failed.

User response

Check the permissions on the configuration file and that it exists.

CCN5501	The directive in the configuration file is not recognized.

Explanation

The directive in the configuration file is not recognized.

User response

Change the directive.

CCN5502	The build was interrupted.

Explanation

The compilation was interrupted and stopped.

User response

Start the compile again.

CCN5503	The name is already used in the configuration file.

Explanation

The identifier has already been used in the configuration file.

User response

Change the name to be another name that is not already used.

CCN5504	The template argument must be a constant integral expression.

Explanation

The argument for the template was not an integral constant expression.

User response

Change the expression to be an integral constant expression.

CCN5505	The build failed and there are no messages.

Explanation

The compiler has experienced an internal failure.

User response

Report the problem to your IBM C++ service representative.

CCN5506	The configuration file "%1$s" is empty.

Explanation

The configuration file is empty.

In the message text:

"%1$s" is the name of the configuration file.

User response

Check that the right configuration file has been specified.

CCN5507	The attempt to load "%1$s" from the default library path failed.

Explanation

The dynamic load of the compiler extension failed.

In the message text:

"%1$s" is the name of the extension that failed to load.

User response

Check the tool option on the command line or in the configuration file.

CCN5508   **The file "%1$s" cannot be loaded: the program file is not an ordinary file, or its mode does not allow execution, or search permission is denied on a component of the path prefix.**

Explanation

The loading of the file failed because of access permissions or it was incorrectly specified.

In the message text:

"%1$s" is the name of the file.

User response

Check the tool option on the command line or in the configuration file.

CCN5509   **The file "%1$s" cannot be loaded: the program file has a valid magic number in its header, but the header is damaged or is incorrect for the machine on which the file is to be run.**

Explanation

The program could not be loaded because the header for the file is corrupt.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that the file has not been corrupted.

CCN5510   **The file "%1$s" cannot be loaded: too many symbolic links were encountered in translating the path name.**

Explanation

The file could not be loaded because there were too many symbolic links in the path name.

In the message text:

"%1$s" is the name of the file.

User response

Remove some of the symbolic links in the path name.

CCN5511   **The file "%1$s" cannot be loaded: incorrect XCOFF header or some problems in linking.**

Explanation

The file could not be loaded because the header is corrupt or improperly linked.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that the file has not been corrupted.

CCN5512   **The file "%1$s" cannot be loaded: the program requires more memory than is allowed by the system.**

Explanation

The file could not be loaded because it requires too much memory.

In the message text:

"%1$s" is the name of the file.

User response

Increase the allocated memory to the program.

CCN5513   **The file "%1$s" cannot be loaded: the file is currently open for writing by a process.**

Explanation

The file could not be loaded because it is currently open for writing.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that the file is not being used by another process and recompile.

CCN5514	The file "%1$s" cannot be loaded: a component of a path name exceeded 255 characters, or an entire path name exceeded 1023 characters.

Explanation

The file could not be loaded because the path or some component of the path is too long.

In the message text:

"%1$s" is the name of the file.

User response

Shorten the length of the path or of the component of the path that is too long.

CCN5515	The file "%1$s" cannot be loaded: a component of the file name does not exist.

Explanation

The file could not be loaded because some component of the name does not exist.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that all directories in the path name exist or change the path for the file.

CCN5516	The file "%1$s" cannot be loaded: a component of the path prefix is not a directory.

Explanation

The file could not be loaded because one of the components of the name is not a directory.

In the message text:

"%1$s" is the name of the file.

User response

Change the path so that all components in the path prefix are directories.

CCN5517	The file "%1$s" cannot be loaded: the process root or current

directory is located in a virtual file system that has been unmounted.

Explanation

The file could not be loaded because the file system is not mounted.

In the message text:

"%1$s" is the name of the file.

User response

Mount the required file system.

CCN5518	The file "%1$s" cannot be loaded: the file name is null.

Explanation

The file could not be loaded because the file name is null.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that the file name is not null.

CCN5519	The file "%1$s" cannot be loaded: the file cannot be found.

Explanation

The file could not be loaded because the could not be found.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that the file exists.

CCN5522	The file "%1$s" cannot be loaded: DosLoadModule return code is %2$s.

Explanation

The file could not be loaded because of operating system errors.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that the file is correctly specified for the operating system.

CCN5523   **Linkage %1$s is not known. extern "C" is assumed.**

Explanation

The specified linkage is unknown and extern "C" will be used.

In the message text:

%1$s is the unrecognized linkage.

User response

Change the linkage specification.

CCN5524   **The file "%1$s" cannot be loaded.**

Explanation

The file could not be loaded because of operating system errors.

In the message text:

"%1$s" is the name of the file.

User response

Ensure that the file is correctly specified for the operating system.

CCN5525   **The enum cannot be packed to the requested size of %1$s bytes.**

Explanation

The range of values specified for the enumeration is too large to be packed into the specified number of bytes.

In the message text:

%1$s is the number of bytes specified.

User response

Change the number of bytes allowed for the enumeration or change the enumerators to have a smaller range.

CCN5526   **One or more error messages have been disabled.**

Explanation

An error was encountered but the error message has been suppressed.

User response

Do not suppress the error message or fix the error.

CCN5527   **The build failure may be because of an Internal Compiler Error or because a tool failed to generate a message.**

Explanation

Informational message about why the build failed with no message.

User response

Report the problem to your IBM C++ service representative.

For more information visit Internal Compiler Error (www.ibm.com/support/docview.wss?uid=swg21110810).

CCN5530   **Unable to load server rc = %1$s.**

Explanation

Call to the Db2 coprocessor API to load the server failed.

In the message text:

%1$s is the return code from the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5531   **Unable to open DBRM file.**

Explanation

Call to open DBRM file failed.

User response

Ensure the file name is correctly specified.

CCN5532   **Unable to initialize SQL coprocessor services: rc = %1$s.**

Explanation

Call to initialize SQL Statement Coprocessor failed.

In the message text:

%1$s is the return code from the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5533	Unable to compile SQL statement: rc = %1$s.

Explanation

Call to the Db2 coprocessor API to compile SQL statement failed.

In the message text:

%1$s is the return code from the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5534	Unrecognized SQL TYPE IS statement.

Explanation

The SQL TYPE IS statement is unrecognized.

User response

Refer to the Db2 documentation for the cause of the problem and the corrective action.

CCN5535	Unable to terminate services: rc = %1$s.

Explanation

Call to terminate SQL statement coprocessor failed.

In the message text:

%1$s is the return code from the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5536	Unable to print SQL message: rc = %1$s.

Explanation

Call to extract formatted message for SQLCODE failed.

In the message text:

%1$s is the return code from the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5538	Unable to register host variable:%1$s, rc = %2$s.

Explanation

Call to register a host variable failed.

In the message text:

%1$s is the name of the host variable. %2$s is the return code from the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5539	Compiling with the SQL compiler option resulted in the following message: %1$s.

Explanation

Call to compile an SQL statement failed.

In the message text:

%1$s is a diagnostic message emitted by the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and the corrective action.

CCN5540	Unable to find host variable %1$s.

Explanation

The host variable does not exist.

In the message text:

%1$s is the return code from the Db2 coprocessor API call.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5541　　　　　The asm constraint "%1$s" is not supported.

Explanation

The asm operand constraint is not recognized.

In the message text:

"%1$s" is the letter in the constraint set that is not supported.

User response

Remove the constraint.

CCN5542　　　　　The asm constraint "%1$s" cannot be used for this operand type.

Explanation

The asm operand constraint is not valid for the provided operand.

In the message text:

"%1$s" is the letter in the constraint set that is incompatible with the operand type.

User response

Remove the constraint or modify the operand to a compatible type.

CCN5543　　　　　Too many registers are required in the asm statement.

Explanation

The asm statement cannot be compiled because it requires too many registers.

User response

Reduce the complexity of the asm statement.

CCN5545　　　　　The total number of operands exceeds %1$s.

Explanation

The maximum number of asm operands has been exceeded.

In the message text:

%1$s is the maximum number of operands an asm statement may have.

User response

Reduce the number of operands in the asm instruction.

CCN5546　　　　　The operand expression is not a modifiable l-value.

Explanation

The operand must be a modifiable l-value.

User response

Provide a modifiable l-value output operand.

CCN5548　　　　　The symbolic name %1$s used in an operand specifier is duplicated.

Explanation

An operand name specifier has been used more than once.

In the message text:

%1$s is the duplicated operand name specifier.

User response

Select a different operand name specifier for one of the operands.

CCN5549　　　　　The named operand "%1$s" is not defined.

Explanation

The operand name specifier does not refer to an operand.

In the message text:

"%1$s" is the undefined operand name specifier.

User response

Change the name specifier to the intended operand.

CCN5551　　　　　The output operand is not specified with an "=" or "+" constraint. "=" is assumed.

Explanation

An output operand must be specified with an "=" or "+".

User response

Add an "=" or "+" to the constraint set for all output operands.

CCN5552　　　　　The register name "%1$s" is unknown.

Explanation

The register name in the clobber set of the asm statement is not recognized. The clobbered register information is ignored.

In the message text:

"%1$s" is the unknown register name.

User response

Correct the name of the clobbered register.

CCN5553	The operand number is out of range.

Explanation

The operand number specified does not refer to an operand.

User response

Correct the operand number to the intended operand.

CCN5554	The output constraint "%1$s" is not at the beginning.

Explanation

"=" or "+" is not the first character of the constraint for an output operand.

In the message text:

"%1$s" is the output operand constraint without a leading "=" or "+".

User response

Make "=" or "+" the first character of the constraint for output operands.

CCN5555	A matching constraint cannot be used in an output operand.

Explanation

Matching constraints are not permitted for output operands.

User response

Provide the correct constraint for the output operand.

CCN5556	A matching constraint cannot reference an input operand.

Explanation

Matching constraints may not refer to input operands.

User response

Provide the correct constraint for the input operand.

CCN5557	The asm operand does not match the specified constraint.

Explanation

The provided operand expression is not valid for the specified constraint.

User response

Provide the correct constraint or correct the operand expression.

CCN5558	The matching constraint "%1$s" has been used more than once.

Explanation

Two matching constraints in an asm statement may not refer to the same operand.

In the message text:

"%1$s" is the duplicated asm matching constraint.

User response

Provide the correct constraint for one of the operands.

CCN5562	No code is generated for the asm statement. The asm statement is ignored.

Explanation

ASM code generation is disabled.

User response

Specify the ASM option to enable code generation of asm statements.

CCN5563	Member function "%1$s" was not explicitly instantiated with its containing class because the function definition was not found.

Explanation

The function must be defined in the same translation unit as the explicit instantiation.

In the message text:

"%1$s" is the function being instantiated.

User response

Define the function in the same translation unit as the explicit instantiation.

CCN5564   **Explicit instantiation requires a function definition in the same translation unit, definitions in a tempinc definition file cannot be seen.**

Explanation

A template must be defined in the same translation unit as the explicit instantiation.

User response

The user should not attempt to explicitly instantiate a function defined in the template definition file for tempinc processing.

CCN5565   **The asm constraint "%1$s" cannot be used for the output operand.**

Explanation

The asm operand constraint is not valid for the output operand.

In the message text:

"%1$s" is the letter in the constraint set that is incompatible with the output operand.

User response

Remove the constraint or change it to a compatible constraint for the output operand.

CCN5566   **Register variable "%1$s" cannot be initialized.**

Explanation

A register variable cannot be initialized since a register does not occupy storage in the object.

In the message text:

"%1$s" is an identifier.

User response

Remove the initializer.

CCN5567   **Register "%1$s" is already used for variable "%2$s".**

Explanation

The register has already been reserved for another variable. A register cannot be reserved for multiple variables.

In the message text:

"%1$s" is a register name and "%2$s" is an identifier.

User response

Use a unique register name for each register variable.

CCN5569   **The specified register name %1$s is not a valid general purpose register name.**

Explanation

Only the general purpose register name can be used.

In the message text:

"%1$s" is the register name.

User response

Specify a valid general purpose register name.

CCN5570   **The register %1$s specified for the variable "%2$s" is ignored.**

Explanation

The register variable is supported only when inline assembler support is enabled.

In the message text:

"%1$s" is the register name. "%2$s" is the variable name.

User response

Specify the inline assembler option to enable inline assembler support.

CCN5571   **The operand has the invalid type "%1$s".**

Explanation

The type of the operand expression is not valid.

In the message text:

"%1$s" is the invalid type of the operand.

User response

Change the type of the operand if necessary or remove the operand altogether.

CCN5572   **The specified register name "%1$s" is not valid.**

Explanation

The name specified for the asm register is not a valid register on the target architecture.

In the message text:

"%1$s" is the name of an invalid register.

User response

Use a valid register name.

CCN5574   **The constraint "%1$s" cannot be used on the operand that has the "XL" constraint.**

Explanation

The operand that has the operand "XL" cannot have any other constraint.

In the message text:

"%1$s" is the constraint.

User response

Remove all the constraints except "XL" and its parameter for the operand that is defined in the asm statement.

CCN5575   **Only the output operand can be defined in the asm statement.**

Explanation

The asm statement is used to define the assembly data. The defined operand cannot be listed in the input operand list.

User response

Move the operand that is defined in the asm statement to the output operand list.

CCN5576   **The specified data size "%1$s" is not valid. The default value "%2$s" is assumed.**

Explanation

Only the positive integer number can be used to specify the data size.

In the message text:

"%1$s" is the data size. "%2$s" is the default data size, currently 256.

User response

Specify the data size with an integer that is greater than 0.

CCN5577   **No valid constraint is specified for the operand. The constraint "%1$s" is assumed.**

Explanation

At least one valid constraint needs to be specified for the operand.

In the message text:

"%1$s" is the constraint.

User response

Specify the valid constraint.

CCN5578   **The specified register name "%1$s" is not a valid general purpose register name.**

Explanation

Only the general purpose register name can be used.

In the message text:

"%1$s" is the register name.

User response

Specify a valid general purpose register name.

CCN5582   **The data type "%1$s" of variable %2$s is not suitable for a register.**

Explanation

A register specification is used for a variable with a data type that cannot be held in a register.

In the message text:

"%1$s" is a data type. "%2$s" is an identifier.

User response

Remove the register specification for the variable.

CCN5583   **The register variable specification is non-portable.**

Explanation

Using the register variable specification may cause problems when porting the code to another system.

User response

Remove the use of the register variable specification.

CCN5584   **The type of host variable expression "%1$s" is not a supported type.**

Explanation

An invalid host variable expression was specified in an SQL statement.

In the message text:

"%1$s" is the number of the host variable expression in the statement where 1 is the first expression, 2 is the second, and so on.

User response

Refer to the Db2 documentation for the cause of the problem and use a corrective action for the return code.

CCN5585   **Variable "%1$s" cannot be used as a host variable as its type is not supported by Db2.**

Explanation

The user specified a host variable that is not of a type supported by Db2.

In the message text:

"%1$s" is the name of the variable.

User response

Refer to the Db2 documentation for host variable types that are supported.

CCN5589   **An SQL statement that requires conversion to executable code was found in namespace scope.**

Explanation

SQL statements that result in executable code must be declared in the function scope.

User response

Move the SQL statement into the appropriate function scope.

CCN5590   **Dereference may not conform to the current aliasing rules.**

Explanation

The dereferenced expression may violate the ANSI aliasing rules.

User response

Change the pointer or reference type to a compatible type.

CCN5591   **The dereferenced expression has type "%1$s". "%2$s" may point to "%3$s" which has incompatible type "%4$s".**

Explanation

The pointer or reference type is not permitted to point to an object of type "%4$s".

In the message text:

"%1$s" is the type of the dereferenced expression. "%2$s" is the pointer or reference expression. "%3$s" is the object, which the expression may point to. "%4$s" is the type of the object pointed to.

User response

Change the pointer or reference type to a compatible type.

CCN5592   **Check assignment at line %1$s column %2$s of %3$s.**

Explanation

This message provides a traceback indicating how a pointer or reference may point to an object.

In the message text:

"%1$s" is the line of the assignment. "%2$s" is the column of the assignment. "%3$s" is the file of the assignment.

User response

Change the pointer or reference type to a compatible type.

CCN5595   **The modifier "%1$s" cannot be used on the last operand.**

Explanation

The modifier '%1$s' can be used any operand other than the last one.

In the message text:

"%1$s" is the modifier in the asm statement.

User response

Remove the modifier '%1$s' from the last operand.

CCN5596	The constraint "%1$s" cannot be used on the last operand.

Explanation

The constraint "%1$s" can be used on any operand other than the last one.

In the message text:

"%1$s" is the constraint in the asm statement.

User response

Remove the constraint "%1$s" from the last operand.

CCN5598	The operand expression is not an l-value.

Explanation

The operand must be an l-value.

User response

Provide an l-value output operand.

CCN5600	The reference to "%1$s" is ambiguous.

Explanation

More than one declaration was found for the reference.

In the message text:

"%1$s" is the ambiguous name.

User response

Fully qualify the reference.

CCN5601	The reference to "%1$s" is ambiguous because "%1$s" is declared in base classes "%2$s" and "%3$s".

Explanation

Multiple inheritance has supplied more than one declaration with the same name.

In the message text:

"%1$s" is the ambiguous reference. "%2$s" and "%3$s" are two base classes.

User response

Fully qualify the reference, or if it is a template, either change it to a template id, or change the base classes.

CCN5602	The reference to "%1$s" is ambiguous because "%1$s" can be accessed via multiple paths to base class "%2$s".

Explanation

Multiple inheritance has resulted in a declaration that can be reached in more than one way through the class hierarchy.

In the message text:

"%1$s" is the ambiguous reference. "%2$s" is the base class.

User response

Fully qualify the reference or change the base classes.

CCN5603	The template declaration "%1$s" cannot be found. An extra "template <>" may be specified on this declaration.

Explanation

Nested template explicit specializations and out-of-line declarations require a template scope for each level of nesting.

In the message text:

"%1$s" is the template declaration.

User response

Check and correct the template scopes on the declaration.

CCN5700	The previous message was produced while processing "%1$s".

Explanation

An informational message giving trace back information.

In the message text:

"%1$s" is the declaration (usually a template) that was being processed when the error occurred.

User response

See the primary message.

CCN5701   **The limit on nested template instantiations has been exceeded while instantiating "%1$s".**

Explanation

A template instantiation that requires another instantiation can set off a chain of instantiations with no end.

In the message text:

"%1$s" is the last instantiation done.

User response

Change the template implementation to avoid the recursion or write an explicit specialization that will stop the instantiation chain at a reasonable point.

CCN5702   **The template argument "%1$s" is not valid.**

Explanation

The template argument does not match the template parameter.

In the message text:

"%1$s" is the template argument.

User response

Correct the template argument.

CCN5704   **The definitions of "%1$s" and "%2$s" have the same linkage signature "%3$s".**

Explanation

The two definitions have the same mangled names and the linker will be unable to distinguish them.

In the message text:

"%1$s" and "%2$s" are the two declarations. "%3$s" is the linkage signature.

User response

Remove one of the definitions or change its linkage.

CCN5705   **The definition of "%1$s" has the same linkage signature, "%2$s", as a symbol from "%3$s".**

Explanation

Two definitions have the same mangled names and the linker will be unable to distinguish them.

In the message text:

"%1$s" is the declaration. "%2$s" is the linkage signature. "%3$s" is the library with the conflicting symbol.

User response

Remove one of the definitions or change its linkage.

CCN5706   **The symbol "%1$s" is already defined by "%2$s" in target "%3$s".**

Explanation

A symbol is being redefined by another compilation unit.

In the message text:

"%1$s" is the duplicate symbol. "%2$s" is the source file or source library. "%3$s" is the target executable, library, or object file.

User response

Remove one of the symbols so that only one definition exists.

CCN5707   **The symbol "%1$s" has the same signature as "%2$s" in target "%3$s".**

Explanation

A symbol is being redefined by another compilation unit.

In the message text:

"%1$s" is the duplicate symbol. "%2$s" is the name of the definition that is resolving to the same symbol as "1s". "%3$s" is the target executable, library, or object file to which "%2$s" belongs.

User response

Remove one of the symbols so that only one definition exists.

CCN5708   **The template argument %1$s does not match the corresponding template parameter of "%2$s".**

Explanation

Template arguments must match the type and kind of the template parameter.

In the message text:

%1$s is the template argument. "%2$s" is the template.

User response

Correct the template argument.

CCN5709	The wrong number of template arguments have been specified for "%1$s", from line %3$s of "%2$s".

Explanation

The number of template arguments must match the number of template parameters.

In the message text:

"%1$s" is the template. "%2$s" is the source file. %3$s is the line number.

User response

Remove the extra template arguments.

CCN5710	The static function "%1$s" is not defined, but is referenced from "%2$s".

Explanation

A referenced static function must be defined.

In the message text:

"%1$s" is the static function, "%2$s" is the referencing location.

User response

Define the function.

CCN5711	Too few template arguments have been specified.

Explanation

The number of template arguments must match the number of template parameters.

User response

Add the missing template arguments.

CCN5712	Too many template arguments have been specified.

Explanation

The number of template arguments must match the number of template parameters.

User response

Remove the extra template arguments.

CCN5713	The template argument "%1$s" is not valid for a non-type template parameter.

Explanation

A non-type template parameter cannot be satisfied with a type.

In the message text:

"%1$s" is the invalid argument.

User response

Change the template argument to a valid value.

CCN5714	The template argument must be a type, to match the template parameter.

Explanation

Only a type-id can be used for a type template argument.

User response

Change the template argument to a valid value.

CCN5715	The local type "%1$s" cannot be used in a template argument.

Explanation

A type defined in a function body or any type compounded from a local type cannot be used as a template argument.

In the message text:

"%1$s" is the local type.

User response

Change the argument to be a non-local type, or move the local type to namespace scope.

CCN5716	The template argument "%1$s" does not match the template parameter "%2$s".

Explanation

A template parameter must have a template argument and a regular type template parameter cannot have a template as an argument.

In the message text:

"%1$s" is the invalid argument, "%2$s" is the template parameter.

User response

Change the argument to correctly match the template parameter.

CCN5717	The template argument cannot use an unnamed type.

Explanation

An unnamed type or any type compounded from an unnamed type cannot be used as a template argument.

User response

Change the argument to be a non-local type, or give the type a name.

CCN5718	An implicit copy assignment operator cannot be created for class with a member of type "%1$s".

Explanation

The class does not have a user specified copy assignment operator and one cannot be generated because of the type of the members of the class.

In the message text:

The type of the member which prohibits the generation of an implicit copy assignment operator.

User response

Provide a copy assignment operator.

CCN5719	The previous message was produced while processing the implicit member function "%1$s".

Explanation

Informational message indicating which implicit member function caused the generation of the error or warning message.

In the message text:

The name of the member function.

User response

See the primary message.

CCN5720	Function "%1$s" has internal linkage but is undefined.

Explanation

A function was declared to have internal linkage, possibly because it was declared to be static, but it is not defined.

In the message text:

The name of the function that is not defined.

User response

Define the function.

CCN5721	The template "%1$s" must not be explicitly specialized after instantiation with the same set of template arguments.

Explanation

A use with no explicit specialization will cause an implicit instantiation. This will conflict with the explicit specialization.

In the message text:

"%1$s" is the explicit specialization.

User response

Move the use or the declaration of the explicit specialization.

CCN5722	The partial specialization "%1$s" must be declared before it is used.

Explanation

A use with no partial specialization will cause an implicit instantiation of the primary template. This will give different behavior than an instantiation of the partial specialization.

In the message text:

"%1$s" is the partial specialization.

User response

Move the use or the declaration of the partial specialization.

CCN5723	The inline function "%1$s" is referenced, but it is not defined.

Explanation

A referenced inline function must be defined.

In the message text:

"%1$s" is the inline function.

User response

Define the function.

CCN5724	The non-type template argument "%1$s" of type "%2$s" has wrapped.

Explanation

A non-type template argument has been provided that is outside the range for the argument type.

In the message text:

"%1$s" is the argument value and "%2$s" is its type.

User response

If this is not intended, change the argument value.

CCN5725	The physical size of an array is too large.

Explanation

The maximum allowable size for this target system has been exceeded.

User response

Reduce the size of the array.

CCN5726	The physical size of a class or union is too large.

Explanation

The maximum allowable size for this target system has been exceeded.

User response

Reduce the size of the class or union.

CCN5727	The static storage is too large.

Explanation

A limit on static storage has been exceeded.

User response

Decrease the amount of storage required.

CCN5728	The keyword _Packed must be used in a typedef.

Explanation

The _Packed type specifier can only be used in a typedef declaration.

User response

Use _Packed in a typedef declaration to declare the _Packed class type, then use the typedef name to declare the variable.

CCN5729	The keyword _Packed must be associated with a class definition.

Explanation

The _Packed specifier is only valid on a typedef declaration with a class definition.

User response

Define the _Packed class type in the typedef declaration.

CCN5730	Alias specification cannot be provided for a function definition.

Explanation

Alias specification can only be provided for a function declaration.

User response

Remove alias specification from the indicated function definition.

CCN5731	The external name "%1$s" must not conflict with the name in #pragma csect or the csect name generated by the compiler.

Explanation

The external name specified is identical to the name specified on a #pragma csect or the name generated by the CSECT option.

In the message text:

"%1$s" is the external name in conflict.

User response

Change the name on the #pragma csect, turn off the CSECT option, or change the external name.

CCN5732	The declarations "%1$s" and "%2$s" have the same linkage signature "%3$s".

Explanation

A symbol should only be accessed through a single declaration in a translation unit.

In the message text:

"%1$s" is the first declaration of the conflicting pair. "%2$s" is the second declaration of the conflicting pair. "%3$s" is the linkage signature of the two declarations.

User response

Remove one of the offending declarations.

CCN5800	The conversion from code page "%1$s" to "%2$s" cannot be initialized.

Explanation

The specified code page does not exist.

In the message text:

"%1$s" is the source code page. "%2$s' is the target code page.

User response

Change the specified code page to a valid one.

CCN5801	The character literal is empty.

Explanation

The character literal is invalid because it is empty.

User response

Change the character literal.

CCN5802	The character literal %1$s contains more than one character.

Explanation

The character literal is invalid because it has more than one character.

In the message text:

"%1$s" is the character literal in error.

User response

Change the character literal to a single character.

CCN5803	The value of the character literal %1$s contains more bytes than sizeof(int). Only the right-most bytes are retained.

Explanation

The character literal is invalid because it has too many bytes. The extra bytes to the left are ignored.

In the message text:

"%1$s" is the character literal in error.

User response

Change the character literal.

CCN5804	The characters "/*" are detected in a comment.

Explanation

The start of what may be a comment has been seen inside a comment. The first string "*/" will finish the comment which may result in unexpected behavior if this truly is a nested comment.

User response

Remove the nested comment or the string "/*" from the comment.

CCN5805	Division by zero occurs on the "#%1$s" directive.

Explanation

An attempt was made to divide by zero in a preprocessor directive.

In the message text:

"%1$s" is the preprocessor directive in the source code.

User response

Change the preprocessor directive to not divide by zero.

CCN5806	The parameter "%2$s" has already been used for the macro "%1$s".

Explanation

The same identifier has been used for more than one parameter for a macro.

In the message text:

"%1$s" is the name of the preprocessor macro in error. "%2$s" is the reused parameter from the macro in error.

User response

Change the parameter name.

CCN5807 The #elif directive has no matching #if, #ifdef, or #ifndef directive.

Explanation

The #elif directive requires a previous #if, #ifdef, or #ifndef. It may be that a #endif was added inappropriately.

User response

Remove the #elif directive.

CCN5808 The #else directive has no matching #if, #ifdef, or #ifndef directive.

Explanation

The #else directive requires a previous #if, #ifdef, or #ifndef. It may be that a #endif was added inappropriately.

User response

Remove the #else directive.

CCN5809 The source file is empty.

Explanation

Informational message indicating that the source file contains no preprocessing tokens.

User response

See the primary message.

CCN5810 An empty argument is specified for parameter "%2$s" of the macro "%1$s".

Explanation

The argument specified to the macro is empty.

In the message text:

"%1$s" is the name of the macro. "%2$2" is the parameter receiving the empty argument.

User response

Change the argument.

CCN5811 The #endif directive has no matching #if, #ifdef, or #ifndef directive.

Explanation

The #endif directive requires a previous #if, #ifdef, or #ifndef. It may be that a #endif was added inappropriately.

User response

Remove the #endif directive.

CCN5812 The escape sequence "%1$s" is out of range. Value is truncated.

Explanation

The specified escape sequence is not valid.

In the message text:

"%1$s" is the escape sequence from the source code.

User response

Change the escape sequence.

CCN5813 One or more #endif directives are missing at the end of the file.

Explanation

There must be a #endif for every #if, #ifdef, or #ifndef. It may be that a #endif was removed inappropriately.

User response

Add the missing #endif.

CCN5814 Expecting a macro name on the #%1$s directive but found "%2$s".

Explanation

The text specified for the macro name is invalid.

In the message text:

"%1$s" is the preprocessor directive. "%2$s" is the text found where the macro name was expected.

User response

Change the text for the macro name.

CCN5815 Expecting the end of the line on the #%1$s directive but found "%2$s".

Explanation

The end of line that was expected to terminate the preprocessing directive was not found.

In the message text:

"%1$s" is the preprocessor directive. "%2$s" is the unexpected input.

User response

Change the preprocessing directive.

CCN5816	Too many arguments are specified for the macro "%1$s". The extra arguments are ignored.

Explanation

The extra arguments specified for the macro are ignored.

In the message text:

"%1$s" is the name of the macro.

User response

Remove the extra arguments.

CCN5817	The comment which began on line %1$s did not end before the end of the file.

Explanation

The "*/" ending the comment was not found before the end of the file.

In the message text:

"%1$s" is the line number on which the comment began.

User response

Add "*/" to finish the comment.

CCN5818	The continuation sequence at the end of the file is ignored.

Explanation

End of file is unexpected after the continuation sequence.

User response

Remove the continuation sequence.

CCN5819	Unable to open the file %1$s. %2$s.

Explanation

The file could not be opened because of the reason indicated.

In the message text:

"%1$s" is the file name that could not be opened. "%2$s" is the text returned by the system when the file open failed.

User response

Ensure that the file can be opened.

CCN5820	Unable to read the file %1$s. %2$s.

Explanation

The file could not be read because of the reason indicated.

In the message text:

"%1$s" is the file name that could not be opened. "%2$s" is the text returned by the system when the file open failed.

User response

Ensure that the file exists and can be read.

CCN5821	The floating point literal "%1$s" is out of range.

Explanation

The floating point literal is not valid.

In the message text:

"%1$s" is the incorrect literal.

User response

Change the floating point literal.

CCN5822	The name "%1$s" must not be defined as a macro.

Explanation

The name cannot be used as a macro.

In the message text:

"%1$s" is the name of the reserved macro name.

User response

Change the name of the macro.

CCN5823   **The name "%1$s" must not be undefined as a macro.**

Explanation

The name cannot be undefined as a macro.

In the message text:

"%1$s" is the name of the reserved macro name.

User response

Change the name of the macro.

CCN5824   **The header of the #include directive is empty.**

Explanation

The #include directive is improperly specified.

User response

Change the #include specification.

CCN5825   **The character "%1$s" is not allowed.**

Explanation

The character is not valid.

In the message text:

"%1$s" is the character.

User response

Change the character.

CCN5826   **The use of the ## operator in the macro "%1$s" is not valid.**

Explanation

The use of the ## operator is not valid.

In the message text:

"%1$s" is the name of the macro in error.

User response

Change the ## operator.

CCN5827   **The constant expression on the #%1$s directive contains a syntax error at "%2$s".**

Explanation

There is a syntax error in the constant expression.

In the message text:

"%1$s" is the preprocessor directive. "%2$s" is the token that is causing the syntax error.

User response

Fix the syntax of the constant expression.

CCN5828   **The escape sequence "%1$s" is not known. The backslash is ignored.**

Explanation

The escape sequence is not valid and the backslash is ignored.

In the message text:

"%1$s" is the escape sequence.

User response

Remove the backslash or change the escape sequence to a valid one.

CCN5829   **The suffix of the floating point literal "%1$s" is not valid.**

Explanation

The floating point literal is improperly specified.

In the message text:

"%1$s" is the floating point literal.

User response

Change the floating point literal.

CCN5830   **The suffix of the integer literal "%1$s" is not valid.**

Explanation

The integer literal is improperly specified.

In the message text:

"%1$s" is the floating point literal.

User response

Change the integer literal.

CCN5831   **The parameter list for the macro "%1$s" contains a syntax error at "%2$s".**

Explanation

There is a syntax error in the parameter list for the macro.

In the message text:

"%1$s" is the name of the macro. "%2$s" is the token that is causing the syntax error.

User response

Fix the syntax error in the parameter list.

CCN5832	The value, "%1$s", of the wide character is not valid.

Explanation

The value of the wide character is not valid.

In the message text:

"%1$s" is the value of the wide character.

User response

Change the value of the wide character.

CCN5833	The multibyte character "%1$s" is unknown.

Explanation

The multibyte character is unknown.

In the message text:

"%1$s" is the multibyte character in error.

User response

Change the multibyte character.

CCN5834	A header name is expected on the #include directive but "%1$s" is found.

Explanation

The #include directive is not valid.

In the message text:

"%1$s" is the unexpected text found.

User response

Change the #include directive.

CCN5835	The file "%1$s" cannot be included because the maximum nesting of %2$s has been reached.

Explanation

The maximum number of nested include files has been reached.

In the message text:

"%1$s" is the file name. "%2$s' is the maximum include file nesting limit for the compiler.

User response

Remove some of the included files or change the include structure to not nest as deeply.

CCN5836	The #include file %1$s is not found.

Explanation

The specified include file was not found.

In the message text:

"%1$s" is the file name.

User response

Ensure that the file exists, change the name of the included file, or use the include path option to specify the path to the file.

CCN5837	An incomplete argument is specified for the parameter "%2$s" of the macro "%1$s".

Explanation

The argument to the macro is invalid.

In the message text:

"%1$s" is the name of the macro. "%2$s" is the macro parameter.

User response

Change the argument to the macro.

CCN5838	An incomplete parameter list is specified for the macro "%1$s".

Explanation

The parameter list to the macro is incomplete.

In the message text:

"%1$s" is the macro name.

User response

Change the parameter list.

CCN5839 **Preprocessor internal error in "%1$s". File "%2$s": Line %3$s.**

Explanation

An internal error has occurred in the preprocessor.

In the message text:

"%1$s" is the name of the compiler function at the time of the error. "%2$s" is the source file that was being processed at the time of the error. "%3$s" is the line number that was being processed at the time of the error.

User response

Contact your IBM C++ service representative.

CCN5840 **The integer literal "%1$s" is out of range.**

Explanation

The integer literal is not valid.

In the message text:

"%1$s" is the integer literal that is out of range.

User response

Change the integer literal.

CCN5841 **The wide character literal %1$s contains more than one character. The last character is used.**

Explanation

More than one character has been specified for a wide character literal.

In the message text:

"%1$s" is the literal.

User response

Remove the extra characters from the wide character literal.

CCN5842 **The line number %1$s on the #line directive must contain only decimal digits.**

Explanation

The #line directive contains an invalid number.

In the message text:

"%1$s" is the invalid line number specified in the #line directive.

User response

Change the number in the #line directive or remove the #line directive.

CCN5843 **Expecting a file name or the end of line on the #line directive but found "%1$s".**

Explanation

The #line directive is invalid.

In the message text:

"%1$s" is the unexpected text.

User response

Remove the extra symbols from the #line directive.

CCN5844 **Expecting a line number on the #line directive but found "%1$s".**

Explanation

The line number specified in the #line directive is invalid.

In the message text:

"%1$s" is the unexpected text.

User response

Change the #line directive.

CCN5845 **The #line value "%1$s" must not be zero.**

Explanation

The line number for a #line directive must not be zero.

In the message text:

"%1$s" is the invalid value specified in the #line directive.

User response

Change the line number for the #line directive.

CCN5846 **The #line value "%1$s" is outside the range 1 to 32767.**

Explanation

The line number for a #line directive is too large.

In the message text:

"%1$s" is the invalid value specified in the #line directive.

User response

Change the line number for the #line directive.

CCN5847	Expected an identifier but found "%2$s" in the parameter list for the macro "%1$s".

Explanation

The parameter to the macro is invalid.

In the message text:

"%1$s" is the macro name. "%2$s" is the unexpected text found.

User response

Change the parameter to the macro.

CCN5848	The macro name "%1$s" is already defined with a different definition.

Explanation

An attempt is being made to redefine the macro.

In the message text:

"%1$s" is the macro name.

User response

Change the name of the macro being defined.

CCN5849	The octal literal "%1$s" contains non-octal digits.

Explanation

The octal literal can only contain the digits 0-7.

In the message text:

"%1$s" is the octal literal.

User response

Change the literal.

CCN5850	Expecting "(" on the #%1$s directive, but found "%2$s"

Explanation

The "(" that was expected following the preprocessing directive was not found.

In the message text:

"%1$s" is the preprocessor directive. "%2$s" is the unexpected input.

User response

Change the preprocessing directive.

CCN5851	The #line directive has no effect.

Explanation

The context for the #line directive gives it no additional meaning.

User response

Delete the #line directive.

CCN5852	The #line value "%1$s" is outside the range 1 to 2147483647.

Explanation

The line number for a #line directive is too large.

In the message text:

"%1$s" is the invalid value specified in the #line directive.

User response

Change the line number for the #line directive.

CCN5857	The macro name "%1$s" is reserved but the directive is processed.

Explanation

The macro name is a reserved name.

In the message text:

"%1$s" is the macro name.

User response

Change the name of the macro.

CCN5858	The macro name "%1$s" is reserved but the directive is processed.

Explanation

The macro name is a reserved name.

In the message text:

"%1$s" is the macro name.

User response

Change the name of the macro to one that is not reserved.

CCN5859	#error directive: %1$s.

Explanation

A #error directive has been processed.

In the message text:

"%1$s" is the text that was specified by the #error directive in the source.

User response

Remove the #error directive.

CCN5860	A parameter name is expected after the # operator in the macro "%1$s" but "%2$s" is found.

Explanation

The right operand to the # operator is invalid.

In the message text:

"%1$s" is the macro name. "%2$s" is the unexpected text.

User response

Change the right operand to the # operator.

CCN5861	Too few arguments are specified for macro "%1$s". Empty arguments are used.

Explanation

Not enough arguments have been specified for the macro.

In the message text:

"%1$s" is the macro name.

User response

Add more arguments to the macro.

CCN5862	The unknown preprocessing directive "%1$s" is ignored.

Explanation

The preprocessing directive is unknown.

In the message text:

"%1$s" is the unknown directive.

User response

Change the preprocessing directive.

CCN5863	A character literal must end before the end of the source line.

Explanation

The character literal is improperly specified.

User response

Change the character literal.

CCN5864	A #include header must end before the end of the source line.

Explanation

The #include directive is improperly specified.

User response

Change the #include directive.

CCN5865	A character literal must end before the end of the source line.

Explanation

The character literal is improperly specified.

User response

Change the character literal.

CCN5866	A string literal must end before the end of the source line.

Explanation

The string literal is improperly specified.

User response

Change the string literal.

CCN5868	A string literal must end before the end of the source line.

Explanation

The string literal is improperly specified.

User response

Change the string literal.

CCN5869	%1$s digits are required for the universal character name "%2$s".

Explanation

Universal character names must follow the format \UNNNNNNNN or \uNNNN, where N is a hexadecimal digit.

In the message text:

"%1$s" is an integer and "%2$s" is a universal character name.

User response

Either pad or truncate the digits used for the universal character name.

CCN5870	The universal character name "%1$s" is not in the allowable range for an identifier.

Explanation

Hexadecimal values representing characters in the basic character set (base source code set) and the code points reserved by ISO/IEC 10646 for control characters are not allowed. The following characters are also disallowed: (1) Any character that has a short identifier that is less than 00A0. The exceptions are 0024 ($), 0040 (@), or 0060 (`). (2) Any character that has a short identifier that is in the code point range D800 through DFFF inclusive.

In the message text:

"%1$s" is a universal character name.

User response

Change the universal character name to an allowable one.

CCN5871	Incomplete or invalid multibyte character, conversion failed.

Explanation

The multibyte character is invalid.

User response

Change the multibyte character.

CCN5872	A string literal cannot be longer than 32765 characters.

Explanation

The string literal is too long.

User response

Change the string literal.

CCN5873	Syntax error in _Pragma operator: "%1$s" was expected but "%2$s" was found. The pragma is ignored.

Explanation

A syntax error has occurred and the first unexpected token is "%1$s". The only valid token at this point is "%2$s".

In the message text:

"%2$s" is the invalid text. "%1$s" is expected correct text.

User response

Change the incorrect token to the expected one.

CCN5874	An #include_next directive is found in a primary source file.

Explanation

The #include_next directive should only be used in the header files.

User response

Change the #include_next to an #include.

CCN5875	A header name is expected in the #include_next directive but "%1$s" is found.

Explanation

No header file name is provided after the #include_next directive.

In the message text:

"%1$s" is the unexpected text found.

User response

Specify the header file name. Enclose the system header names in angle brackets and the user header names in double quotes.

CCN5876	An #include_next header must end before the end of the source line.

Explanation

An #include_next directive was specified across two or more lines.

User response

Change the #include_next directive so that it and its arguments are contained on a single line.

CCN5877	The #include_next file %1$s is not found.

Explanation

The file specified in the #include_next directive cannot be found. See the documentation for the file search order.

In the message text:

%1$s is the file name.

User response

Ensure that the file exists. Change the name of the included file or use the include path option to specify the path to the file.

CCN5878	The header file name in the #include_next directive is empty.

Explanation

The header file name in the #include_next directive should not be empty.

User response

Specify a non-empty header file name in the #include_next directive.

CCN5879	#warning directive: %1$s.

Explanation

A #warning directive has been processed.

In the message text:

"%1$s" is the text that was specified by the #warning directive in the source.

User response

Remove the #warning directive.

CCN5880	The invalid character "%1$s" was found in a wide character or wide string literal. The character will be ignored.

Explanation

The wide character or wide string literal contains an invalid character that will be ignored.

In the message text:

"%1$s" is the invalid character.

User response

Remove the character.

CCN5881	The pragma GCC system_header directive is only permitted in an include file. The pragma will be ignored.

Explanation

The pragma should not be used in the primary source file so it will be ignored.

User response

Remove the pragma.

CCN5882	Expected ')' but found "%2$s" in the parameter list for the macro "%1$s".

Explanation

A variable argument parameter cannot appear anywhere but the end of a parameter list.

In the message text:

"%1$s" is the macro name. "%2$s" is the unexpected text found.

User response

Move the variable argument parameter to the end of the parameter list.

CCN5883	Use of __VA_ARGS__ in macro "%1$s" is unexpected; expected "%2$s".

Explanation

The ISO C99 variable argument identifier __VA_ARGS__ has been used in a GNU variadic macro.

In the message text:

"%1$s" is the macro name. "%2$s" is the variable argument parameter identifier.

User response

Replace __VA_ARGS__ with "%2$s".

CCN5884	The GNU variable argument identifier "%2$s" of macro "%1$s" is not permitted in the current langlvl mode.

Explanation

Possibly missing ',' or a GNU variable argument identifier has been specified in an illegal langlvl mode.

In the message text:

"%1$s" is the macro name. "%2$s" is the variable argument parameter identifier.

User response

Set the langlvl option appropriately.

CCN5885	The universal character is out of range for this platform.

Explanation

This platform only supports valid universal characters less than \u0100.

User response

Provide a valid universal character.

CCN5886	The universal character "%1$s" is not valid.

Explanation

The universal character is out of the allowable ranges.

In the message text:

"%1$s" is the invalid character.

User response

Provide a valid universal character.

CCN5887	The hexadecimal literal "%1$s" is not valid.

Explanation

The hexadecimal literal is incomplete or contains an invalid hex character.

In the message text:

"%1$s" is the invalid hexadecimal literal.

User response

Provide a valid hexadecimal literal.

CCN5888	The current option settings do not allow the use of "long long". The suffix of the integer literal "%1$s" is not valid.

Explanation

The suffix of the integer literal is "LL", but this is disallowed due to option settings.

In the message text:

"%1$s" is the integer literal.

User response

Delete the integer suffix or change the option settings to allow "long long".

CCN5889	No arguments have been provided for the variable argument parameter of macro "%1$s". Empty arguments are used.

Explanation

Not enough arguments have been specified for the macro.

In the message text:

"%1$s" is the macro name.

User response

Add more arguments to the macro.

CCN5891	Missing white space between the identifier "%1$s" and the replacement list.

Explanation

C++0x requires white space to seperate the identifier and its replacement list in the definition of an object-like macro.

In the message text:

"%1$s" is the identifier.

User response

Add white space after the identifier.

CCN5892	The preprocessor controlling expression evaluates differently between C++0x and non-C++0x langlvls.

Explanation

Note that the arithmetic changes in C++0x preprocessor may have different evaluations of the conditional inclusion directives.

User response

Add explicit suffix to the constant if necessary.

CCN5893	The header file name "%1$s" in the #include directive shall not start with a digit.

Explanation

C++0x requires that the first character of a header file name to be non-digit.

In the message text:

"%1$s" is the header file name.

User response

Rename the header if possible.

CCN5894	Keyword "%1$s" has a different meaning in non-C++0x language levels than under C++0x.

Explanation

Note that this keyword's functionality will change under C++0x.

In the message text:

%1$s is keyword name.

User response

If you are using C++0x language level mode, please refer to documentation for usage of this keyword.

CCN5895	The character "%1$s" has been truncated.

Explanation

The underlying type of wchar_t cannot hold the specified character.

In the message text:

"%1$s" is the character from the source code.

User response

Change the character or increase the size of the wchar_t type by compiling in 64-bit mode.

CCN5900	#include search attempted to open the file "%1$s".

Explanation

Informational message about the search path when attempting to find an include file.

In the message text:

"%1$s" is the file name.

User response

See the primary message.

CCN5901	The expression on the #%1$s directive evaluates to %2$s.

Explanation

Informational message about the condition directive value.

In the message text:

"#%1$s" is the directive name. "%2$s" is either 1 or 0. If the expression on the #%1$s is defined or evaluates to True, "%2$s" is 1; 0 otherwise.

User response

See the primary message.

CCN5902	The nesting level of the #%1$s directive is %2$s.

Explanation

Informational message about the conditional nesting level.

In the message text:

"#%1$s" is the directive name. "%2$s" is an integer, starting from 1. It indicates the nesting level of the condition directive #%1$s.

User response

See the primary message.

CCN5903	defined(%1$s) evaluates to %2$s.

Explanation

Informational message about the defined value.

In the message text:

"%1$s" is the directive name. "%2$s" is either 1 or 0. If %1$s is defined, %2$s is 1; %2$s is 0 otherwise.

User response

See the primary message.

CCN5904	Token skipping due to conditional compilation begins here.

Explanation

Informational message about token skipping due to conditional compilation.

User response

See the primary message.

CCN5905	Token skipping due to conditional compilation ends here.

Explanation

Informational message about token skipping due to conditional compilation.

User response

See the primary message.

CCN5921	"%1$s" is defined in the file "%2$s" on line %3$s.

Explanation

Informational message about where a macro is defined.

In the message text:

"%1$s" is the macro name. "%2$s" is the file name. "%3$s" is the line number.

User response

See the primary message.

CCN5922	#include_next search attempted to open the file "%1$s".

Explanation

Informational message about the search path when attempting to find an include file.

In the message text:

"%1$s" is the file name.

User response

See the primary message.

CCN5923	There is a #pragma extension without a matching #pragma extension (pop) in the same file. The "pop" has been inserted by the compiler at the end of the file.

Explanation

A #pragma extension should always have a matching #pragma extension (pop).

User response

Insert a #pragma extension (pop) to match the #pragma extension.

CCN6086	The initializer list in the compound literal expression must be a constant expression.

Explanation

If a compound literal expression is used outside a function body, its initializer list must be a constant expression.

User response

Change the initializer list to a constant expression.

CCN6087	The catch block(s) has no effect.

Explanation

The NOEXH option indicates that no exception will be thrown.

User response

Don't use NOEXH option or don't use catch blocks in the program.

CCN6088	The exception specification is being ignored.

Explanation

The NOEXH option indicates that no exception will be thrown.

User response

Don't use NOEXH option or don't use exception specification.

CCN6089   **The throw expression is being ignored.**

Explanation

The NOEXH option indicates that no exception will be thrown.

User response

Don't use NOEXH option or don't use throw expression.

CCN6090   **The destructor(s) of "%1$s" might not be called if an exception is thrown.**

Explanation

The NOEXH option does not prevent exceptions from other translation units; stack unwinding will fail to call the destructor(s).

In the message text:

"%1$s" is the object.

User response

Don't use NOEXH option or ensure that the object will not be subject to stack unwinding.

CCN6091   **The friend declaration "%1$s" specifies a default argument expression and is not a definition.**

Explanation

If a friend declaration specifies a default argument expression, that declaration must be a definition.

In the message text:

"%1$s" is the name of the function.

User response

Add the definition with the function declaration.

CCN6092   **The declaration "%1$s" is also declared as a friend with a default argument expression in file "%2$s", on %3$s.**

Explanation

If a friend declaration specifies a default argument expression, that declaration shall be the only declaration of the function or function template in the translation unit.

In the message text:

"%1$s" is the name of the function, "%2$s" is the file name, "%3$s" is the line and column number where the friend function is declared.

User response

Remove the default argument or move it to the non-friend declaration.

CCN6093   **The throw expression has no effect since exception handling is unsupported.**

Explanation

The current platform does not support exception handling.

User response

Do not use the throw expression.

CCN6099   **The value "%1$s" specified on pragma "%2$s" is not a valid architecture level. The pragma is ignored.**

Explanation

A valid architecture level is required on this pragma.

In the message text:

"%1$s" is the architecture level. "%2$s" is the name of the pragma.

User response

Correct the architecture level or remove the pragma.

CCN6100   **A local variable or compiler temporary variable is being used to initialize reference member "%1$s".**

Explanation

Initializing a reference member with a temporary or local variable is dangerous since it will result in a dangling reference if the object's life-span is longer than the temporary or local variable.

In the message text:

"%1$s" is the reference member.

User response

Initialize the member with another object.

CCN6101	A return value of type "%1$s" is expected.

Explanation

The function is expected to return a value but no return statement is given.

In the message text:

"%1$s" is the expected type.

User response

Add a return statement to the function.

CCN6102	"%1$s" might be used before it is set.

Explanation

The compiler cannot determine that the variable is initialized before it is used.

In the message text:

"%1$s" is the variable.

User response

Initialize the variable.

CCN6103	The address of a local variable or temporary is used in a return expression.

Explanation

The address of a local object is being returned by the function but this object's life-span will end at the function return, resulting in a dangling reference.

User response

Return a different value.

CCN6104	The condition evaluates to a constant value.

Explanation

The condition is a constant expression which may result in code that can never be reached or a loop that may not terminate.

User response

Change the condition to be non-constant.

CCN6105	The condition contains a non-parenthesized assignment.

Explanation

An assignment is being performed in a condition.

User response

Change the expression; this warning is often caused by an assignment being used when an equality comparison is desired.

CCN6106	The local type "%1$s" must not be used in a declaration with external linkage.

Explanation

The function has external linkage but is using a local type so the linkage signature of the function cannot be described.

In the message text:

"%1$s" is the type used in the source code declaration.

User response

Use a non-local type in the function prototype.

CCN6107	An object of abstract class "%1$s" cannot be created.

Explanation

The class has pure virtual functions so an object of this class type cannot be created.

In the message text:

"%1$s" is the class.

User response

Ensure that the class contains no pure virtual functions.

CCN6108	"%1$s" is not a valid type.

Explanation

The specified type is not a legal type.

In the message text:

"%1$s" is the type.

User response

Change the type.

CCN6109   **The use of undefined class "%1$s" is not valid.**

Explanation

The use requires that the type be defined and not just declared.

In the message text:

"%1$s" is the class.

User response

Define the class.

CCN6110   **The referenced type "%1$s" contains a circular reference back to "%2$s".**

Explanation

The two types contain references to each other that both require definitions.

In the message text:

"%1$s" and "%2$s" are the types in question.

User response

Change the first class to only require a declaration of the second class.

CCN6111   **Only function declarations can have default arguments.**

Explanation

An attempt has been made to have default arguments for a parameter in a declaration that is not a function declaration.

User response

Remove the default initializers.

CCN6112   **"%1$s" is a pure virtual function.**

Explanation

Informational message for listing pure virtual functions.

In the message text:

"%1$s" is the name of the function.

User response

See the primary message.

CCN6113   **The class template name "%1$s" must be followed by a < in this context.**

Explanation

The template must have its template arguments specified.

In the message text:

"%1$s" is the name of the template class.

User response

Add the < and the appropriate template arguments followed by >.

CCN6114   **"%1$s" is not allowed as a function return type.**

Explanation

The return type of the function is not valid.

In the message text:

"%1$s" is the type that the function is attempting to return.

User response

Change the function return type.

CCN6115   **"%1$s" cannot be declared to have type "void".**

Explanation

The type "void" is not valid for this declaration.

In the message text:

"%1$s" is the name of the declaration.

User response

Change the type.

CCN6116   **If "%1$s" is a function name, one of its parameters may contain an undeclared type name.**

Explanation

A function declaration that has an unknown type as a parameter may have been incorrectly parsed as a variable declaration with a paren-style initializer.

In the message text:

"%1$s" is the name of the attempted function or variable declaration.

User response

See the primary message.

CCN6117	"%1$s" cannot use the abstract class "%2$s" as the type of an object, parameter type, or return type.

Explanation

The class has pure virtual functions so an object cannot be created.

In the message text:

"%1$s" is what is attempting to use the abstract base class "%2$s".

User response

Change the type of the object being created.

CCN6118	The declaration of "%1$s" uses the undefined class "%2$s" when the class must be complete.

Explanation

The usage requires the class to be defined.

In the message text:

"%1$s" is the name of the declaration. "%2$s" is the type being declared.

User response

Define the class.

CCN6119	The weak declaration of "%1$s" must be public.

Explanation

Weak attribute must be attached to declarations that have external linkage.

In the message text:

"%1$s" is the function declaration.

User response

Remove offending attribute.

CCN6120	"using %1$s" must refer to a member of a base class.

Explanation

The using declaration must refer to a member of a base class.

In the message text:

"%1$s" is the argument of the using directive.

User response

Change the declaration.

CCN6121	"%1$s" is a class member and can be declared only in a member declaration.

Explanation

A using declaration for a class member shall be a member declaration

In the message text:

"%1$s" is a class member.

User response

Remove the using declaration, or move it into a class derived from the class that contains the member declaration.

CCN6122	A non-type template parameter cannot have type "%1$s".

Explanation

Only an integral, enumeration, pointer to object, pointer to function, lvalue reference, or pointer-to-member type, or a cv-qualified version of one of these types, is allowed as the type of a non-type template parameter.

In the message text:

"%1$s" is the invalid type.

User response

Correct the non-type template parameter.

CCN6123	An initializer is not allowed for "%1$s".

Explanation

An initializer has been specified for a declaration that does not create an object.

In the message text:

"%1$s" is the name of the declaration.

User response

Remove the initializer.

CCN6124   **A union cannot contain a static data member.**

Explanation

Static data members have external linkage. They cannot be used in unions, because members of a union share the same memory.

User response

Change the union into a class or struct, or remove the static data member.

CCN6125   **The data member "%1$s" cannot have the same name as its containing class.**

Explanation

Every data member of a class must have a name different from the name of the containing class

In the message text:

"%1$s" is the name of a class data member.

User response

Change the name of the data member so that it is not the same as the class name.

CCN6126   **The static data member "%1$s" is not allowed in a local class.**

Explanation

Since static data members have external linkage it makes no sense to have one inside a local class. If this were permitted, the static data member would be visible in scopes where the class itself is not visible.

In the message text:

"%1$s" is a data member of a local class.

User response

Remove the static data member or move the class to global scope.

CCN6127   **Only static data members with const integral or const enumeration type can specify an initializer in the class definition.**

Explanation

The declaration of a static data member is not a definition. The definition should appear in a namespace scope enclosing the class that contains this member. Only static data members of const integral or const enumeration type may be initialized inside the class declaration. In this case, they must still be defined in the enclosing scope without an initializer.

User response

Move the initializer to the definition in the containing scope, or make the type a const integral or const enumeration.

CCN6128   **The bit field "%1$s" must have integral or enumeration type.**

Explanation

A bit field is used to represent a sequence of bits. Only integral or enumeration types makes sense for bit fields.

In the message text:

"%1$s" is the name of the bit field.

User response

Change the type of the bit field or remove the bit field.

CCN6129   **The "mutable" specifier must not be applied to a member with type "%1$s".**

Explanation

The mutable specifier cannot be applied to const, static or reference members.

In the message text:

"%1$s" is the type of the data member.

User response

Remove the mutable specifier from the data member or change the type of the data member

---

CCN6130   **A static data member cannot be a direct or indirect member of an unnamed class.**

Explanation

Static data members are defined and accessed using the name of the class in which they are defined. If the

class has no name, the static data member cannot be defined or accessed.

User response

Give the class a name, or make the data member non-static.

CCN6131	A zero-length bit field must not have a name.

Explanation

Bit fields with zero-length are used to specify alignment of the next bit field at the boundary of an allocation unit. They have no data and are therefore not accessed for any reason.

User response

Change the length of the bit field or remove the name.

CCN6132	"%1$s" must not be a member of a union. "%2$s" has a non-trivial copy assignment operator.

Explanation

Unions can only contain members that do not have copy assignment operators.

In the message text:

"%1$s" is the declaration of the union member. "%2$s" is the name of the class that has a non-trivial copy assignment operator.

User response

Change the member to be a POD-type.

CCN6133	A union must not contain a member of type "%1$s".

Explanation

Reference variables are not allowed in unions.

In the message text:

"%1$s" is the type.

User response

Change the type of the member.

CCN6134	An anonymous %1$s must not have private or protected members.

Explanation

Only public members are allowed in anonymous aggregates.

In the message text:

%1$s is the keyword union, struct or class.

User response

Ensure that all members are public.

CCN6135	The anonymous %1$s member "%2$s" must not have the same name as its containing class.

Explanation

Every data member of a class must have a name different from the name of the containing class. Members of anonymous struct, class, or union are referenced as members of their containing class, so their name must also be different from the name of containing class.

In the message text:

"%1$s" is either union, struct or class. "%2$s" is the name of the member.

User response

Change the name of the member.

CCN6136	"%1$s" cannot be a union member, because "%2$s" has a non-trivial constructor.

Explanation

A trivial constructor is created by the compiler for a class with: no virtual functions and no virtual base classes. All the direct base classes of its class must have trivial constructors, and all of its nonstatic data members that are of class type have must have trivial constructors. An object with a non-trivial constructor may not be a member of a union.

In the message text:

"%1$s" is the declaration of the union member. "%2$s" is the name of the class that has a non-trivial constructor.

User response

Change the union to a struct or a class or remove the member which has a non-trivial constructor.

CCN6137   **"%1$s" cannot be a union member, because "%2$s" has a non-trivial destructor.**

Explanation

Unions can only contain members that do not have destructors.

In the message text:

"%1$s" is the declaration of the union member. "%2$s" is the name of the class that has a non-trivial destructor.

User response

Change the member to be a POD-type.

CCN6138   **Ellipsis (...) cannot be used for "%1$s".**

Explanation

An overloaded operator cannot have an ellipsis as a parameter.

In the message text:

"%1$s" is the function.

User response

Change the ellipsis parameter.

CCN6139   **An exception-specification can appear only in a function or pointer declaration.**

Explanation

An exception-specification is not valid for this type.

User response

Remove the exception-specification.

CCN6140   **The member "%1$s" must be declared in its containing class definition.**

Explanation

The member that is being defined out of line is not declared in the class.

In the message text:

"%1$s" is the member.

User response

Declare the variable or function as a member of the class.

CCN6141   **An anonymous %1$s can define only non-static data members.**

Explanation

Static members are not allowed in anonymous aggregates.

In the message text:

"%1$s" is the keyword union, struct, or class.

User response

Remove the static member declaration.

CCN6142   **"%1$s" is ill-formed because "%2$s" does not have a unique final overrider.**

Explanation

The virtual function has more than one final overrider because of virtual base classes.

In the message text:

"%1$s" is the name of the derived class. "%2$s" is the qualified name of the virtual function with no final overrider.

User response

Ensure that only base class has a final overrider for the function or define the virtual function in the class.

CCN6143   **"%1$s" cannot be used as a base class because it contains a zero-dimension array.**

Explanation

The base class cannot be used since it contains an array that has zero elements.

In the message text:

"%1$s" is the base class.

User response

Change the base class.

CCN6144   **All array dimensions for non-static members must be specified and be greater than zero.**

Explanation

An array dimension is missing or is negative.

User response

Ensure that all dimensions are specified as non-negative numbers.

CCN6145	A using-directive cannot appear in a class scope.

Explanation

Using directives can only be specified in namespace or lexical block scope.

User response

Remove the using directive.

CCN6146	The enumerator "%1$s" cannot have the same name as its containing class.

Explanation

This is a name collision.

In the message text:

"%1$s" is the enumerator.

User response

Change the name of either the enumerator or the class.

CCN6147	"%1$s" cannot be declared as inline or static.

Explanation

There are restrictions on "main" since it is the program starting point.

In the message text:

"%1$s" is the function name.

User response

Remove the inline or static specifiers.

CCN6148	The non-member function "%1$s" cannot be declared "%2$s".

Explanation

The specifier is only valid for member functions.

In the message text:

"%1$s" is the name of the function. "%2$s" is the specifier.

User response

Remove the specifier.

CCN6149	The value "%1$s" specified on pragma "%2$s" is lower than the effective architecture value "%3$s". The pragma is ignored.

Explanation

The nested architecture level must be specified in the increasing order.

In the message text:

"%1$s" is the invalid architecture level. "%2$s" is the name of the pragma. "%3$s" is the effective architecture level.

User response

Correct the architecture level by increasing its value or removing the pragma.

CCN6150	A constructor for "%1$s" cannot be declared "%2$s".

Explanation

The specifier is not valid for a constructor.

In the message text:

"%1$s" is the struct, or class. "%2$s" is the specifier.

User response

Remove the specifier.

CCN6151	When the first parameter to the constructor has type "%1$s", the constructor must have other parameters without default arguments.

Explanation

This is an ill-formed copy constructor since the first parameter is not a reference.

In the message text:

"%1$s" is the type.

User response

Change the first parameter to be a reference to make this a copy constructor.

CCN6152   **The destructor for "%1$s" cannot be declared "%2$s".**

Explanation

The specifier is not valid for a destructor.

In the message text:

"%1$s" is the struct, or class. "%2$s" is the specifier.

User response

Remove the specifier.

CCN6153   **A destructor must not have a return type or parameter.**

Explanation

A return type or parameter has been specified for a destructor.

User response

Remove the return type or parameter.

CCN6154   **The destructor "%1$s" must not be declared as a template.**

Explanation

A destructor must not be a member template.

In the message text:

"%1$s" is the destructor.

User response

Remove or change the destructor to be a regular non-template destructor.

CCN6155   **The static member function "%1$s" must not be declared "%2$s".**

Explanation

A static function cannot have cv-qualifiers.

In the message text:

"%1$s" is the name of the function. "%2$s" is the specifier.

User response

Remove the cv-qualifiers.

CCN6156   **A conversion operator must not have parameters.**

Explanation

A conversion operator has been specified with parameters.

User response

Remove the parameters.

CCN6157   **The conversion operator of type "%1$s" will never be directly called to perform a conversion.**

Explanation

A conversion operator has been specified with void type.

In the message text:

"%1$s" is the type.

User response

Change the void specifier to another type.

CCN6158   **The function template "%1$s" must not be declared as virtual.**

Explanation

A member function template cannot be virtual.

In the message text:

"%1$s" is the function.

User response

Change the function so that it is not virtual or not a template.

CCN6159   **The "%1$s" qualifier must not be applied to "%2$s".**

Explanation

The qualifier is not valid for this declaration.

In the message text:

"%1$s" is the qualifier. "%2$s" is the declarator.

User response

Remove the qualifier.

CCN6160   **The virtual function "%1$s" is not allowed in a union.**

Explanation

Unions cannot have virtual member functions.

In the message text:

"%1$s" is the name of the function.

User response

Remove the virtual specifier.

CCN6161	The default arguments for "%1$s" must not be followed by uninitialized parameters.

Explanation

All parameters following a parameter with a default initializer must also have default initializers.

In the message text:

"%1$s" is the name of the function.

User response

Add default initializers for all parameters after the first parameter with a default initializer.

CCN6162	The pure-specifier (= 0) is not valid for the non-virtual function "%1$s".

Explanation

The pure-specifier (= 0) is used to state that a virtual function does not have a definition. It has no meaning for non-virtual functions.

In the message text:

"%1$s" is the name of the function.

User response

Make the function virtual or remove the pure-specifier.

CCN6163	The exception-specification for "%1$s" is less restrictive than the exception-specification for "%2$s".

Explanation

The exception specification for an overriding function must not list more types than the exception specification for the original function.

In the message text:

"%1$s" is the overriding function. "%2$s" is the original function.

User response

Match the exception specification for the overriding function with the original function or modify the exception specification of the original function.

CCN6164	The return type for "%1$s" differs from the return type of "%2$s" that it overrides.

Explanation

When overriding a function, the name, parameters and the return type should match.

In the message text:

"%1$s" and "%2$s" are the names of the functions.

User response

Modify the return type of the overriding function to match the original function.

CCN6165	The virtual function "%1$s" is not a valid override of "%2$s" because the qualifiers are not compatible.

Explanation

The return for an override must be more cv-qualified than the function in the base class.

In the message text:

"%1$s" is the function. "%2$s" is the function being overridden.

User response

Add the missing qualifiers to the override.

CCN6166	The virtual function "%1$s" is not a valid override because "%2$s" is an inaccessible base class of "%3$s".

Explanation

The override is not correct because the base class containing the function is not accessible.

In the message text:

"%1$s" is the function. "%2$s" is the base class. "%3$s" is the derived class.

User response

Remove the override.

CCN6167   **The virtual function "%1$s" is not a valid override because "%2$s" is an ambiguous base class of "%3$s".**

Explanation

The override is not correct because there are multiple base classes containing the function.

In the message text:

"%1$s" is the function. "%2$s" is the base class. "%3$s" is the derived class.

User response

Remove the override.

CCN6168   **The virtual function "%1$s" is not a valid override because "%2$s" is not a base class of "%3$s".**

Explanation

The override is not correct because the return type is not complete nor the containing class.

In the message text:

"%1$s" is the function. "%2$s" is the base class. "%3$s" is the derived class.

User response

Change the return type to be a complete class or the containing class.

CCN6169   **The function template "%1$s" cannot have default template arguments.**

Explanation

Default template arguments are not allowed on a function template.

In the message text:

"%1$s" is the function template.

User response

Remove the default template arguments.

CCN6170   **Both "main" and "WinMain" are defined.**

Explanation

Only one of "main" and "WinMain" can be defined in a program.

User response

Remove either "main" or "WinMain".

CCN6171   **The friend function "%1$s" cannot be defined in a local class.**

Explanation

A class defined in a function body can not contain a definition of a friend function.

In the message text:

"%1$s" is the friend function.

User response

Remove the definition of the friend in the local class.

CCN6172   **More than one function "%1$s" has non-C++ linkage.**

Explanation

Only functions with C++ linkage can be overloaded.

In the message text:

"%1$s" is the function.

User response

Change the name of the function so that it is unique.

CCN6173   **"%1$s" is not a valid parameter type.**

Explanation

The type of the parameter is not valid.

In the message text:

"%1$s" is the type.

User response

Change the type of the parameter.

CCN6174   **The member "%1$s" is not declared as a template in its containing class definition.**

Explanation

This out-of-line template class member does not exist in the class template.

In the message text:

"%1$s" is the member.

User response

Declare the member in the class template or remove the out-of-line declaration.

CCN6175   **The class template partial specialization "%1$s" does not match the primary template "%2$s".**

Explanation

Either both the primary template and the partial specialization must be unions or neither of them must be unions.

In the message text:

"%1$s" is the partial specialization, "%2$s" is the primary template.

User response

Make the class key match.

CCN6176   **"%1$s" is declared with a conflicting linkage.**

Explanation

The linkage is not compatible with the linkage specified in a previous declaration.

In the message text:

"%1$s" is the declarator.

User response

Change the linkage of one of the declarations so that they are compatible.

CCN6177   **Only variables with static storage can be declared to have thread local storage.**

Explanation

The __thread is specified but the declaration is not for a variable, or the variable is not declared static.

User response

Remove the __thread specifier.

CCN6178   **"%1$s" is declared to have both %2$s and %3$s linkage.**

Explanation

The linkage is not compatible with the linkage specified in a previous declaration.

In the message text:

"%1$s" is the declarator. "%2$s" is the linkage specifier. "%3$s" is the linkage specifier.

User response

Change the linkage of one of the declarations so that they are compatible.

CCN6179   **"%1$s" contains conflicting linkages.**

Explanation

The linkage is not compatible with the linkage specified in a previous declaration.

In the message text:

"%1$s" is the declaration.

User response

Change the linkage of one of the declarations so that they are compatible.

CCN6180   **Namespace "%1$s" must be global.**

Explanation

A namespace can only be declared within another namespace or in the global namespace.

In the message text:

"%1$s" is the namespace.

User response

Move the namespace to be within another namespace.

CCN6181   **The number of function parameters exceeds the target operating system limit of %1$s.**

Explanation

Too many function parameters have been specified.

In the message text:

%1$s is the maximum number of function parameters allowed.

User response

Reduce the number of function parameters.

CCN6182   **"%1$s" must have two or more parameters.**

Explanation

The declaration of operator new does not have enough parameters.

In the message text:

"%1$s" is the function.

User response

Ensure that the function has at least two parameters.

CCN6183   **The non-member function "%1$s" must have at least one parameter of type class or enumeration, or a reference to class or enumeration.**

Explanation

The operator overload does not have the correct type for its parameters.

In the message text:

"%1$s" is the function.

User response

Change the types of the parameters.

CCN6184   **Wrong number of parameters for "%1$s".**

Explanation

The declaration for the operator overload does not have the correct number of parameters.

In the message text:

"%1$s" is the function.

User response

Change the declaration to have the proper number of parameters.

CCN6185   **"%1$s" must be a non-static member function.**

Explanation

The operator overload is only valid as a non-static member function.

In the message text:

"%1$s" is the function.

User response

Change the declaration to be a non-static member function.

CCN6186   **The last parameter for postfix "%1$s" must be of type "int".**

Explanation

The last parameter for the operator overload must be of type int.

In the message text:

"%1$s" is the function.

User response

Change the last parameter to be of type int.

CCN6187   **"%1$s" must not have default arguments.**

Explanation

The overloaded operator must not have default arguments.

In the message text:

"%1$s" is the function.

User response

Remove the default arguments.

CCN6188   **The return type for the "%1$s" must not be the containing class.**

Explanation

The return type for the overloaded function cannot be the containing class.

In the message text:

"%1$s" is the operator.

User response

Change the return type.

CCN6189   **The return type for "operator new" must be "void *".**

Explanation

The specified return type is invalid.

User response

Change the return type.

CCN6190   **The first parameter for "operator new" must have type "size_t".**

Explanation

The type of the first parameter is incorrect.

User response

Change the type of the first parameter.

CCN6191   **The first parameter of "operator new" cannot have a default argument.**

Explanation

It is invalid to specify a default argument for "operator new".

User response

Remove the default argument.

CCN6192   **"%1$s" must not be declared static in global scope.**

Explanation

Overloaded versions of "operator new" and "operator delete" must not be declared static.

In the message text:

"%1$s" is the function.

User response

Remove the static specifier.

CCN6193   **The member function "%1$s" must not be declared virtual.**

Explanation

"Operator new" and "operator delete" cannot be declared virtual in a member list.

In the message text:

"%1$s" is the member function.

User response

Remove the virtual specifier.

CCN6194   **"%1$s" must be a class member function or a global function.**

Explanation

The scope for the overloaded "operator new" or "operator delete" is invalid.

In the message text:

"%1$s" is the function.

User response

Remove the declaration.

CCN6195   **The return type for "operator delete" must be "void".**

Explanation

A return type other than "void" has been specified for "operator delete".

User response

Change the return type to be "void".

CCN6196   **The return type cannot be "%1$s" because "%2$s" does not have an "operator->" function.**

Explanation

The return type must have an "operator->" function.

In the message text:

"%1$s" is the type. "%2$s" is the class or struct.

User response

Add an "operator->" function to the return type.

CCN6197   **Parameter number %1$s for "operator delete" must have type "%2$s".**

Explanation

The parameter has a wrong type.

In the message text:

"%1$s" is the function parameter number. "%2$s" is the required type.

User response

Change the parameter to the required type.

CCN6198   **Too many parameters are specified for "operator delete".**

Explanation

There are too many parameters specified.

User response

Remove the extra parameters.

| CCN6199 | "main" must have a return type of type "int". |

Explanation

A return type other than "int" has been specified for "main".

User response

Change the return type of "int" to be "int".

| CCN6200 | An ellipsis (…) handler must not be followed by another handler. |

Explanation

An ellipsis handler will match all thrown objects, and the handlers are tried in the order that they are specified. Therefore the ellipsis handler must be last.

User response

Move the ellipsis handler to be the last handler.

| CCN6201 | A "new" expression with type "%1$s" must have an initializer. |

Explanation

A const type must be initialized even when it is allocated with new.

In the message text:

"%1$s" is the type.

User response

Add an initializer.

| CCN6202 | No candidate is better than "%1$s". |

Explanation

Informational message indicating one of the best matches for operator overloading.

In the message text:

"%1$s" is the match.

User response

See the primary message.

| CCN6203 | The conversion from "%1$s" to "%2$s" matches more than one conversion function. |

Explanation

There is more than one conversion sequence so it is an ambiguous conversion.

In the message text:

"%1$s" and "%2$s" are the types.

User response

Provide a closer matching conversion.

| CCN6204 | The conversion matches "%1$s". |

Explanation

Informational message indicating a matched conversion sequence.

In the message text:

"%1$s" is the conversion sequence.

User response

See the primary message.

| CCN6205 | The error occurred while converting to parameter %1$s of "%2$s". |

Explanation

Informational message about conversion sequences.

In the message text:

"%1$s" is the parameter number. "%2$s" is the function.

User response

See the primary message.

| CCN6206 | The class template instantiation of "%1$s" is ambiguous. |

Explanation

The instantiation cannot be performed since the template is not uniquely identified.

In the message text:

"%1$s" is the template.

User response

Qualify the instantiation to make it uniquely identify a template.

CCN6207   **The template arguments match "%1$s".**

Explanation

Informational message indicating what the template arguments match.

In the message text:

"%1$s" is the matched template.

User response

See the primary message.

CCN6208   **The use of "%1$s" is not valid.**

Explanation

The name is being incorrectly used.

In the message text:

"%1$s" is the invalid name.

User response

Fix the usage of the name.

CCN6209   **The name lookup in the context of "%1$s" resolved to "%2$s".**

Explanation

Informational message indicating the resolution of the name.

In the message text:

"%1$s" is the context. "%2$s" is the resolution.

User response

See the primary message.

CCN6210   **Name lookup in the context of the expression resolved to "%1$s".**

Explanation

Informational message indicating what the resolution of the name.

In the message text:

"%1$s" is the resolution.

User response

See the primary message.

CCN6211   **The conversion type must represent the same type in the context of the expression as in the context of the class of the object expression.**

Explanation

The conversion type is resolved in the left side of the member access and in the current scope and it can only resolve in one or it must resolve to the same entity in both.

User response

Change the context so that the lookups match.

CCN6212   **The type of the conversion function cannot be resolved.**

Explanation

Some names in the type of the conversion function are not declared.

User response

Change the conversion function so that all elements are declared.

CCN6213   **The temporary for the throw expression is of type "%2$s" and cannot be initialized with an expression of type "%1$s".**

Explanation

Throw expressions throw a copy (rather than the object itself) and the temporary cannot be initialized with the given expression.

In the message text:

"%2$s" is the type of the throw expression. "%1$s" is the initialization type.

User response

Change the initializer or provide appropriate constructors.

CCN6214   **The member expression resolves to the type "%1$s".**

Explanation

The left side of the class member access refers to type "%1$s".

In the message text:

"%1$s" is the type being accessed.

User response

Change the class member access expression.

CCN6215	"%1$s" must not have an initializer list.

Explanation

Only constructors can have constructor initializer lists and this function is not a constructor.

In the message text:

"%1$s" is the function.

User response

Remove the constructor initializer list.

CCN6216	The unqualified member "%1$s" must be qualified with "%2$s::" and preceded by an "&" to form an expression with type pointer-to-member.

Explanation

A pointer-to-member expression is of the form: "&className::member ".

In the message text:

"%1$s" is the member. "%2$s" are the qualifiers.

User response

Add the qualifiers and address operator.

CCN6217	The second and third operands of the conditional operators must not both be throw expressions.

Explanation

Only one of the second and third operands in a ternary operator can be a throw expression.

User response

Change one of the second and third operators to not be a throw expression or replace the ternary expression with a conditional statement.

CCN6218	When defining the implicitly declared function "%1$s", the header "<new>" should be included.

Explanation

The header "<new>" contains declarations that are necessary for creating some implicitly declared functions and must therefore be included using the #include directive.

In the message text:

"%1$s" is the function being implicitly declared.

User response

Include the header "<new>" using an include directive.

CCN6219	"%1$s" must be preceded by an "&" to form an expression with type pointer-to-member.

Explanation

A non-static member must be associated with an object.

In the message text:

"%1$s" is the member.

User response

Add the address operator.

CCN6220	The qualified type name "%1$s" used in the explicit destructor call does not match the destructor type "~%2$s".

Explanation

The form used to indicate a destructor in a pseudo-destructor call is not valid.

In the message text:

"%1$s" is the expected destructor specification. "~%2$s" is the destructor name.

User response

Change the specification of the destructor.

CCN6221	The explicit destructor call must be invoked for an object.

Explanation

An attempt is being made to call a destructor without an object.

User response

Call the destructor as a member access on an object.

CCN6222	The destructor type "%1$s" does not match the object type "%2$s".

Explanation

The destructor indicated does not match the type of the object.

In the message text:

"%1$s" is the type of the destructor. "%2$s" is the type of the object.

User response

Change the destructor to match the type of the object.

CCN6223	"%1$s" is not valid as an identifier expression.

Explanation

The form of the identifier is invalid.

In the message text:

"%1$s" is the invalid form for an identifier.

User response

Change the form to a valid form for an identifier.

CCN6224	"%1$s" cannot be dynamically cast to "%2$s" because "%1$s" does not declare or inherit virtual functions.

Explanation

Only polymorphic classes can be dynamically cast.

In the message text:

"%1$s" is the source class. "%2$s" is the target class.

User response

Remove the dynamic cast.

CCN6225	Name lookup did not find "%1$s" in the context of the template definition.

Explanation

This may cause an error when the template is instantiated. Declarations for non-dependent names are resolved in the template definition.

In the message text:

"%1$s" is the unresolved name.

User response

Correct the unresolved name by removing the reference or declaring it.

CCN6226	Declarations for non-dependent names are resolved in the template definition.

Explanation

This is a submessage.

User response

See the primary message.

CCN6227	"%1$s" does not depend on a template argument.

Explanation

This is a submessage.

In the message text:

"%1$s" is the name that is not dependent on the template.

User response

See the primary message.

CCN6228	Argument number %1$s is an lvalue of type "%2$s".

Explanation

Informational message describing the type of a parameter to a function.

In the message text:

"%1$s" is the argument number. "%2$s" is the lvalue type.

User response

See the primary message.

CCN6229	Argument number %1$s is an rvalue of type "%2$s".

Explanation

Informational message describing the type of a parameter to a function.

In the message text:

"%1$s" is the argument number. "%2$s" is the rvalue type.

User response

See the primary message.

CCN6230	Argument number 1 is the implicit "this" argument.

Explanation

Informational message describing the implicit "this" argument in a member function.

User response

See the primary message.

CCN6231	The conversion from argument number %1$s to "%2$s" uses %3$s.

Explanation

Informational message describing a conversion sequence.

In the message text:

%1$s is the argument number. "%2$s" is the parameter type. %3$s is more detailed text.

User response

See the primary message.

CCN6232	""%1$s""

Explanation

Informational message describing a standard conversion sequence.

In the message text:

"%1$s" is more detailed generated text.

User response

See the primary message.

CCN6233	""%1$s" followed by "%2$s""

Explanation

Informational message describing a standard conversion sequence.

In the message text:

"%1$s" is more detailed generated text. "%2$s" is more detailed generated text.

User response

See the primary message.

CCN6234	""%1$s" followed by "%2$s" followed by "%3$s""

Explanation

Informational message describing a standard conversion sequence.

In the message text:

"%1$s" is more detailed generated text. "%2$s" is more detailed generated text. "%3$s" is more detailed generated text.

User response

See the primary message.

CCN6235	the user-defined conversion "%1$s"

Explanation

Informational message describing a user-defined conversion sequence.

In the message text:

"%1$s" is the name of a user-defined conversion function.

User response

See the primary message.

CCN6236	the user-defined conversion "%1$s" followed by %2$s

Explanation

Informational message describing a user-defined conversion sequence.

In the message text:

"%1$s" is the name of a user-defined conversion function. %2$s is more detailed generated text.

User response

See the primary message.

CCN6237   **%1$s followed by the user-defined conversion "%2$s"**

Explanation

Informational message describing a user-defined conversion sequence.

In the message text:

%1$s is more detailed generated text. "%2$s" is the name of a user-defined conversion function.

User response

See the primary message.

CCN6238   **%1$s followed by the user-defined conversion "%2$s" followed by %3$s**

Explanation

Informational message describing a user-defined conversion sequence.

In the message text:

%1$s is more detailed generated text. "%2$s" is the name of a user-defined conversion function. %3$s is more detailed generated text.

User response

See the primary message.

CCN6239   **an ellipsis conversion sequence**

Explanation

Informational message about a conversion sequence.

User response

See the primary message.

CCN6240   **the resolved overloaded function "%1$s"**

Explanation

Informational message about a conversion sequence.

In the message text:

"%1$s" is the function.

User response

See the primary message.

CCN6255   **The local label "%1$s" has already been declared as a label.**

Explanation

An attempt was made to declare a local label in the same scope as an existing label or local label.

In the message text:

"%1$s" is a invalid local label.

User response

Remove the local label declaration.

CCN6257   **An rvalue of type "%1$s" cannot be converted to an rvalue of type __complex__.**

Explanation

There is no valid conversion sequence for converting the expression to an expression of type __complex__.

In the message text:

"%1$s" is the type of expression.

User response

Change the expression.

CCN6258   **Conversion from "%1$s" to "%2$s" may cause truncation.**

Explanation

The specified conversion from a wider to a narrower type may cause the loss of significant data.

In the message text:

"%1$s" is a C++ type "%2$s" is a C++ type

User response

Remove the conversion from a wider to a narrower type.

CCN6259   **The initializer list has too few initializers.**

Explanation

An initializer list should have the same number of initializers as the number of elements to initialize.

User response

Add some initializers or decrease the number of elements to initialize.

CCN6260   **An object of type "%2$s" cannot be constructed from an rvalue of type "%1$s".**

Explanation

There is no valid way to construct the desired object from the given type.

In the message text:

"%2$s" is the type being constructed. "%1$s" is the type of the expression.

User response

Change the expression.

CCN6261   **The qualified member "%1$s" should not be in parentheses when forming an expression with type pointer-to-member.**

Explanation

Informational message indicating that removing the parentheses may resolve the error.

In the message text:

"%1$s" is the member.

User response

See the primary message.

CCN6262   **The scope of "%1$s" extends only to the end of the for-statement.**

Explanation

Informational message indicating the scoping of variables introduced in for-statements. This behavior is different in the language standard than in previous levels of the working draft.

In the message text:

"%1$s" is the variable.

User response

Move the declaration above the for-statement.

CCN6263   **Build with "lang(ISOForStatementScopes, no)" to extend the scope of the for-init-statement declaration.**

Explanation

Informational message describing a compatibility option.

User response

See the primary message.

CCN6264   **The template argument must be preceded by an ampersand (&).**

Explanation

The template argument is expected to be the address of an object.

User response

Add the address operator.

CCN6265   **The template argument must be the address of an object or function with extern linkage.**

Explanation

For example string literals are not allowed because they have internal linkage.

User response

Correct the template argument.

CCN6266   **A template argument with type "%1$s" cannot be converted to a template parameter with type "%2$s".**

Explanation

Only certain standard conversion sequences can be applied.

In the message text:

"%1$s" is the argument type. "%2$s" is the parameter type.

User response

Correct the template argument type.

CCN6267   **"%1$s" is declared with internal linkage in source "%2$s".**

Explanation

Informational message about where an object is declared with internal linkage.

In the message text:

"%1$s" is the variable. "%2$s" is the source.

User response

See the primary message.

CCN6268	"%1$s" conflicts with the definition in source "%2$s" because "%3$s" has internal linkage.

Explanation

The variable or function is defined as static in another source file.

In the message text:

"%1$s" is the variable or function. "%2$s" is the source. "%3$s" is the other variable or function with internal linkage.

User response

Remove the static from the other definition.

CCN6269	The template argument for the non-type template parameter of type "%1$s" must be an integral constant expression.

Explanation

Only constant expressions are allowed for integral or enumeration non-type template arguments.

In the message text:

"%1$s" it the template parameter type.

User response

Correct the non-type template parameter.

CCN6270	A function or object name must be expressed as an id-expression.

Explanation

A function or object name used as a non-type template argument must be an id-expression with external linkage.

User response

Correct the template argument to be a name with external linkage.

CCN6271	The "sizeof" operator cannot be applied to a bit field.

Explanation

It is invalid to use the "sizeof" operator on a bit field.

User response

Remove the "sizeof" operator.

CCN6272	The incomplete class "%1$s" is not a valid "catch" type.

Explanation

Only complete types can be used in the type for catch handlers but the specified type has only been declared and not defined.

In the message text:

"%1$s" is the class.

User response

Define the type.

CCN6273	A pointer or reference to the incomplete class "%1$s" is not a valid "catch" type.

Explanation

Only pointers to complete types can be used in the type for catch handlers but the type has only been declared and not defined.

In the message text:

"%1$s" is the incomplete class type.

User response

Change the type in the catch or define the class.

CCN6274	The "catch(%1$s)" cannot be reached because of a previous "catch(%2$s)".

Explanation

Catch handlers are tried sequentially and this catch is unreachable because a previous handler catches everything that this handler can catch.

In the message text:

"%1$s" is the current handler. "%2$s" is the previous handler.

User response

Remove or change the handler.

CCN6275   Too many explicit template arguments are specified for "%1$s".

Explanation

The number and type of template arguments must match the template parameters.

In the message text:

"%1$s" is the template.

User response

Remove the extra template arguments.

CCN6276   The explicit template specialization "%1$s" matches more than one template.

Explanation

The explicit specialization of this function matches multiple function templates. Probably because of allowable non-type template argument conversions.

In the message text:

"%1$s" is the explicit specialization.

User response

Remove the explicit specialization, remove one of the primary templates, or add namespaces to separate the templates.

CCN6277   The explicit template specialization "%1$s" does not match any template.

Explanation

An explicit specialization must specialize a primary template.

In the message text:

"%1$s" is the explicit specialization.

User response

Declare the primary template or correct the explicit specialization.

CCN6278   The deduced type "%1$s" does not match the specialized type "%2$s".

Explanation

The template argument type deduced from the function call does not match the type in the specialization.

In the message text:

"%1$s" is the deduced type, "%2$s" is the specialized type.

User response

Explicitly specify the template arguments or change the call.

CCN6279   A return statement cannot appear in a handler of the function-try-block of a constructor.

Explanation

A return statement is in a handler for a function-try-block of a constructor.

User response

Remove the return statement.

CCN6280   An rvalue expression of type "%1$s" cannot be converted to type "%2$s".

Explanation

No conversion sequence exists for converting an rvalue expression of type "%1$s" to type "%2$s".

In the message text:

"%1$s" is the original type. "%2$s" is the target type.

User response

Change the types or provide conversion functions.

CCN6281   "offsetof" cannot be applied to "%1$s". It is not a POD (plain old data) type.

Explanation

"offsetof" cannot be applied to a class that is not a POD. POD types do not have non-static pointers-to-member, non-POD members, destructors nor copy assignment operators (ie, they are similar to C-style structs).

In the message text:

"%1$s" is the type.

User response

Change the type to be a POD type.

CCN6282	An enumerator from an enumeration that is in error is being referenced.

Explanation

This is a cascading error caused by an error in the definition of the enumeration.

User response

Fix the error in the definition of the enumeration.

CCN6283	"%1$s" is not a viable candidate.

Explanation

Informational message indicating that this was not a viable candidate for overload resolution.

In the message text:

"%1$s" is the potential resolution.

User response

See the primary message.

CCN6284	Predefined "%1$s" is not a viable candidate.

Explanation

Informational message indicating that this was not a viable candidate for overload resolution.

In the message text:

"%1$s" is the potential resolution.

User response

See the primary message.

CCN6285	The specialization matches "%1$s".

Explanation

Informational message indicating what a specialization matches.

In the message text:

"%1$s" is the matched specialization.

User response

See the primary message.

CCN6286	The specialization does not match "%1$s".

Explanation

Informational message indicating what a specialization cannot match.

In the message text:

"%1$s" is what the specialization cannot match.

User response

See the primary message.

CCN6287	"%1$s" has internal linkage but is undefined.

Explanation

A static member variable or static function must be defined.

In the message text:

"%1$s" is the undefined member variable or static function.

User response

Define the member variable or static function.

CCN6288	The explicit template instantiation "%1$s" matches more than one template.

Explanation

The explicit instantiation of this function matches multiple function templates. Probably because of allowable non-type template argument conversions.

In the message text:

"%1$s" is the explicit instantiation.

User response

Remove the explicit instantiation, remove one of the primary templates, or add namespaces to separate the templates.

CCN6289	The implicit object parameter of type "%2$s" cannot be initialized with an implied argument of type "%1$s".

Explanation

A function is being called implicitly and the parameters do not match the expected parameters.

In the message text:

"%2$s" is the implicit object parameter type. "%1$s" is the implied argument type.

User response

Provide an explicit conversion function.

CCN6290	An rvalue cannot be converted to an lvalue reference to a non-const or volatile type.

Explanation

The target of the conversion must be a non-volatile const lvalue reference or an rvalue reference.

User response

See the primary message.

CCN6291	To initialize the reference with an rvalue, "%1$s" must have a copy constructor with a parameter of type "%2$s".

Explanation

Informational message indicating that a copy constructor must be supplied.

In the message text:

"%1$s" is the type of the object. "%2$s" is the type of the parameter.

User response

See the primary message.

CCN6292	Static declarations are not considered for a function call if the function is not qualified.

Explanation

Informational message describing why a static function cannot be considered.

User response

See the primary message.

CCN6293	The explicit instantiation matches "%1$s".

Explanation

Informational message about matching of explicit instantiations.

In the message text:

"%1$s" is the matched explicit instantiation.

User response

See the primary message.

CCN6294	The explicit instantiation does not match "%1$s".

Explanation

Informational message about matching of explicit instantiations.

In the message text:

"%1$s" is the explicit instantiation that is not matched.

User response

See the primary message.

CCN6295	The explicit template instantiation "%1$s" does not match any template.

Explanation

There is no primary template matching this explicit template instantiation.

In the message text:

"%1$s" is the explicit template instantiation.

User response

Remove the explicit template instantiation or declare the primary template..

CCN6296	The const object "%1$s" requires "%2$s" to have a user-declared default constructor.

Explanation

This class has a const object so the class must have a user-declared default constructor.

In the message text:

"%1$s" is the const object. "%2$s" is the class.

User response

Provide a user default-constructor.

CCN6297	The const object "%1$s" needs an initializer or requires "%2$s"

to have a user-declared default constructor.

Explanation

This class has a const object so the class must have a user-declared default constructor.

In the message text:

"%1$s" is the const object. "%2$s" is the class.

User response

Provide a user default-constructor.

CCN6298	**"%1$s" needs to be declared in the containing scope to be found by name lookup.**

Explanation

Informational message about declaring friend classes in the containing scope for the class to be found by name lookup.

In the message text:

"%1$s" is the class.

User response

Declare the class in the enclosing scope.

CCN6299	**"%1$s" is undefined. Every variable of type "%2$s" will assume "%3$s" has no virtual bases and does not use multiple inheritance.**

Explanation

The pointer refers to an incomplete class so it will be assumed that the class has no virtual bases nor multiple inheritance.

In the message text:

"%1$s" is the undefined class. "%2$s" is the pointer type. "%3$s" is the class.

User response

Define the class.

CCN6300	**"%1$s" includes the file "%2$s".**

Explanation

This is a submessage. This message is used to specify that a certain file includes the file "%2$s".

In the message text:

"%1$s" and "%2$s" are the two files in the include chain.

User response

See the primary message.

CCN6301	**The previous error occurs during the processing of file "%1$s".**

Explanation

This is a submessage.

In the message text:

"%1$s" is the file.

User response

See the primary message.

CCN6302	**The conflicting declaration was encountered during the processing of the file "%1$s".**

Explanation

This message describes the include hierarchy that caused the preceding error.

In the message text:

"%1$s" is the file name.

User response

Remove the conflicting declaration.

CCN6303	**"%1$s" is not visible.**

Explanation

This message indicates that the declaration is not visible at the current location.

In the message text:

"%1$s" is the declaration.

User response

Move the declaration to a position prior to the current location.

CCN6304	**"%1$s" is not visible from "%2$s".**

Explanation

This message indicates that the declaration is not visible at the current location.

In the message text:

"%1$s" is the declaration. "%2$s" is the location.

User response

Move the declaration to a position prior to the current location.

CCN6305   **"%1$s" is not complete when included by "%2$s".**

Explanation

The class or struct is incomplete when included from a particular header file location.

In the message text:

"%1$s" is the class. "%2$s" is the header file.

User response

Instantiate the direct nullifier of the virtual function table operator.

CCN6306   **The deleted definition of function "%1$s" must not be out-of-line.**

Explanation

The delete keyword must not be specified on out-of-line declarations.

In the message text:

The deleted definition for function "%1$s" is invalid.

User response

Make the first declaration of function deleted and remove the out-of-line definition.

CCN6307   **The declaration of "%1$s" is deleted and cannot be used.**

Explanation

A function cannot be used if its declaration is deleted.

In the message text:

"%1$s" is the name of the function.

User response

Do not use a function whose declaration is deleted.

CCN6308   **"%1$s" is not a special member function, and therefore it cannot be defaulted.**

Explanation

Only special member functions can be used in defaulted definitions.

In the message text:

"%1$s" is an invalid defaulted definition.

User response

Remove the invalid defaulted definition.

CCN6309   **The deleted definition of function "%1$s" is invalid.**

Explanation

Deleted function definition has been used incorrectly.

In the message text:

"%1$s" is an invalid deleted definition.

User response

Remove the invalid deleted definition.

CCN6310   **The function "%1$s" differs from the expected signature, "%2$s", in a manner that prevents it from being explicitly defaulted.**

Explanation

The declared type of a defaulted function can differ from the otherwise-implicitly declared function only in limited ways.

In the message text:

"%1$s" is the user-declared function signature. "%2$s" is the signature of the otherwise-implicitly declared function.

User response

Change the declaration to match the expected signature or to differ from the expected signature in an allowed manner.

CCN6311   **To be explicitly defaulted on its first declaration, "%1$s" cannot have a parameter type which differs from the one in the expected signature, "%2$s".**

Explanation

The parameter type for a copy constructor or copy assignment operator cannot differ from the one in the

otherwise-implicitly declared function if the function is defaulted on its first declaration.

In the message text:

"%1$s" is the user-declared function signature. "%2$s" is the signature of the otherwise-implicitly declared function.

User response

Change the declaration to have the expected parameter type, or explicitly-default it in a later declaration instead.

CCN6312	The defaulted definition of function "%1$s" must not have default arguments.

Explanation

A function that is explicitly defaulted must not have default arguments.

In the message text:

"%1$s" is the name of the function.

User response

Remove the default arguments from the function.

CCN6313	The function "%1$s" is deprecated.

Explanation

A deprecated function has been used.

In the message text:

"%1$s" is a deprecated function.

User response

Do not use any deprecated function.

CCN6388	A positive integer constant expression must be specified for pragma %1s. The pragma is ignored.

Explanation

The parameter must be a positive integer constant expression.

In the message text:

"%1$s" is a pragma directive.

User response

Correct the pragma specification.

CCN6389	Pragma %1s must be specified only once for the same loop. This pragma is ignored.

Explanation

The pragma is ignored because it has been specified more than once for the same loop.

In the message text:

"%1$s" is a pragma directive.

User response

Remove the extra pragma specification.

CCN6390	Pragma %1s is ignored because it contradicts pragma %2s, which is previously specified for the same loop.

Explanation

The pragma is ignored because it is specified with invalid parameters that conflict with another pragma specified for the same loop.

In the message text:

"%1$s" and "%2$s" are two conflicting pragma directives.

User response

Change the pragma parameter to correct value.

CCN6391	External name "%1$s" has been truncated to "%2$s".

Explanation

The external name exceeds the maximum length and has been truncated. This may result in unexpected behavior if two different names become the same after truncation.

In the message text:

"%1$s" and "%2$s" are identifier names.

User response

Reduce the length of the external name.

CCN6392	The Csect option is ignored due to conflict with the name in #pragma csect.

Explanation

The compiler was unable to generate valid csect names which must be unique.

User response

Change the name in #pragma csect, turn off the CSECT option or add suboption to eliminate the conflict with the name in #pragma csect.

CCN6393	"pragma %1$s" must be specified in namespace scope. The pragma is ignored.

Explanation

The pragma is ignored because it has been specified in an invalid scope such as a function body or class member list.

In the message text:

"%1$s" is the name of the ignored pragma.

User response

Move the pragma to namespace scope.

CCN6394	The "pragma %1$s" and "pragma %2$s" are incompatible for the same declaration. The "pragma %3$s" is ignored.

Explanation

The "pragma "%1$s" is not supported with the use of "pragma %2$s".

In the message text:

"%1$s" and "2$s" are the names of two conflicting pragma derectives. "%3$s" is the name of the ignored pragma.

User response

Remove one of the pragma directives for the declaration.

CCN6395	The pragma argopt and pragma descriptor are incompatible for the same declaration.

Explanation

Only one of the pragmas is supported for each declaration.

User response

Remove either the pragma descriptor, or the pragma argopt for the declaration.

CCN6396	The value given for "pragma priority" must be a constant integral expression in the range between 101 and 65535. The pragma is ignored.

Explanation

The pragma is ignored when the system supports GNU Attributes because the value is not a constant integral expression in the range between 101 and 65535.

User response

Change the value to evaluate to the required range.

CCN6397	The Csect option is ignored due to a naming error.

Explanation

The compiler was unable to generate valid csect names.

User response

Use the pragma csect to name the code and static control sections.

CCN6398	The external name "%1$s" in pragma csect conflicts with another csect name.

Explanation

A pragma csect was specified with a name which has already been specified as a csect name on another pragma.

In the message text:

"%1$s" is an identifier.

User response

Ensure that the two csect names are unique.

CCN6399	There is more than one pragma csect statement.

Explanation

A duplicate pragma csect is ignored.

User response

Remove the duplicate pragma csect statement.

CCN6400   **The incorrect pragma is ignored.**

Explanation

The pragma is incorrect and is ignored.

User response

Correct the pragma.

CCN6401   **An unknown "pragma %1$s" is specified.**

Explanation

The specified pragma is not recognized.

In the message text:

"%1$s" is the name of the unknown pragma.

User response

Change the name of the pragma to one that is applicable to the compiler.

CCN6402   **The options for "pragma %1$s" are incorrectly specified: expected %2$s and found %3$s. The option is ignored.**

Explanation

The options for the pragma are not correctly specified and the pragma is ignored.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the expected option and "%3$s" is the found option.

User response

Change the options to the pragma as indicated.

CCN6403   **The function "%2$s" specified in "pragma %1$s" cannot be found. The pragma is ignored.**

Explanation

The pragma is ignored because it refers to a function that is not declared.

In the message text:

"%1$s" and "%2$s" are the names of the pragma and the undeclared function, respectively.

User response

Change the pragma to refer to a declared function or declare the function.

CCN6404   **The parameter "%1$s" specified for "pragma %2$s" is not valid. The pragma is ignored.**

Explanation

The pragma is ignored because the parameter specified is not valid.

In the message text:

%1$s and "%2$s" are the invalid parameter and the pragma, respectively.

User response

Change the pragma parameter.

CCN6405   **Syntax error in "pragma %1$s": expected "%2$s" and found "%3$s". The pragma is ignored.**

Explanation

The pragma is ignored because there is a syntax error in the pragma directive.

In the message text:

"%1$s" is the name of the pragma. "%2$s" and "%3$s" are the expected text and the incorrect input, respectively.

User response

Correct the syntax of the pragma specification.

CCN6406   **"pragma %1$s" is already specified. The pragma is ignored.**

Explanation

The pragma is ignored because it has already been specified.

In the message text:

"%1$s" is the name of the pragma that is ignored.

User response

Remove the pragma specification.

CCN6407   **The function "%2$s" specified in "pragma %1$s" does not have an implementation. The pragma is ignored.**

Explanation

The pragma is ignored because it requires that the specified function be defined but it is only declared.

In the message text:

"%1$s" is the name of the ignored pragma and "%2$s" is the name of the function that must be defined.

User response

Define the function.

CCN6408	"pragma %1$s" has no effect. The pragma is ignored.

Explanation

The pragma is ignored because it has no effect. It may be that the pragma specifies options that are already in effect.

In the message text:

"%1$s" is the name of the ignored pragma.

User response

See the primary message.

CCN6409	"pragma %1$s" is not supported on the target platform. The pragma is ignored.

Explanation

The pragma is ignored because it is not valid on the target platform.

In the message text:

"%1$s" is the name of the ignored pragma.

User response

See the primary message.

CCN6410	The function "%2$s" specified in "pragma %1$s" is not uniquely identified. The pragma is ignored.

Explanation

The pragma is ignored either because the function specified is overloaded or because it is declared in multiple namespaces. It is not clear which function is being specified.

In the message text:

"%1$s" is the name of the ignored pragma. "%2$s" is the name of the overloaded function.

User response

Either use namespace qualifiers and parameter type signatures to uniquely identify the function or remove the pragma.

CCN6411	"pragma %1$s" must be specified in global scope. The pragma is ignored.

Explanation

The pragma is ignored because it has been specified in an invalid scope such as a function body or class member list.

In the message text:

"%1$s" is the name of the ignored pragma.

User response

Move the pragma to global scope.

CCN6412	The declaration "%2$s" specified in "pragma %1$s" cannot be found. The pragma is ignored.

Explanation

The pragma is ignored because it names a variable or type that has not been declared.

In the message text:

"%1$s" is the name of the ignored pragma and "%2$s" is the name of the variable or the type indicated in the pragma.

User response

Change the pragma to refer to a declared variable or type or declare the indicated variable or type.

CCN6413	The conflicting pragma is specified on line %1$s of "%2$s".

Explanation

This message shows the coordinates of the conflicting pragma.

In the message text:

"%1$s" is the line number. "%2$s" is the file name.

User response

Remove the pragma specification.

CCN6414	The function "%2$s" specified in "pragma %1$s" is a member function. The pragma is ignored.

Explanation

Member functions are not allowed for the pragma specified.

In the message text:

"%1$s" is the pragma name. "%2$s" is the function name.

User response

Specify a non-member function in the pragma or remove the pragma.

CCN6415	The declaration "%2$s" specified in "pragma %1$s" is a member variable. The pragma is ignored.

Explanation

Member variables are not allowed for the pragma specified.

In the message text:

"%1$s" is the pragma name. "%2$s" is the declaration.

User response

Specify a non-member variable in the pragma or remove the pragma.

CCN6416	The declaration "%2$s" specified in "pragma %1$s" is a structure tag. The pragma is ignored.

Explanation

Structure tags are not allowed for the pragma specified.

In the message text:

"%1$s" is the pragma name. "%2$s" is the declaration.

User response

Fix the declaration in the pragma or remove the pragma.

CCN6417	The declaration "%2$s" specified in "pragma %1$s" must have "%3$s" linkage. The pragma is ignored.

Explanation

The pragma is only valid for declarations with specific linkage.

In the message text:

"%1$s" is the pragma name. "%2$s" is the declaration specified in the pragma. "%3$s" is the required linkage for the pragma.

User response

Specify a declaration with the correct linkage or remove the pragma.

CCN6418	The declaration "%1$s" specified in pragma "%2$s" is not compatible with the declaration "%3$s", which is also specified in the pragma. The pragma will be ignored.

Explanation

The two declarations specified in the pragma are incompatible.

In the message text:

"%1$s", and "%3$s" are declarations, "%2$s" is the pragma name.

User response

Change the declarations or remove the pragma.

CCN6420	The packing boundary for "pragma pack" must be 1, 2, 4, 8, or 16. The pragma is ignored.

Explanation

A "pragma pack" has been specified with an invalid boundary.

User response

Change the pack boundary for the "pragma pack" to one of the accepted boundaries or remove the pragma.

CCN6421	Attempting to pop an empty "pragma pack" stack. The current pack setting may be invalid.

Explanation

The specified "pragma pack" stack is empty. A pop operation is not permitted.

User response

Remove the pop operation, or ensure that the "pragma pack" stack has been set up correctly.

CCN6422   **The identifier does not exist within the "pragma pack" stack. The current alignment may change.**

Explanation

The current alignment may change because the identifier does not exist on the pragma pack stack.

User response

Change the name of the identifier specified in the pragma.

CCN6423   **The declaration in "pragma map" has already been mapped to "%1$s". The pragma is ignored.**

Explanation

The pragma is ignored because the declaration has already been mapped.

In the message text:

"%1$s" is the previous mapping of the declaration.

User response

Remove the pragma or change the declaration.

CCN6424   **Priority values in successive "pragma priority" statements must increase.**

Explanation

The priority specified is lower than a priority specified in a previous pragma.

User response

Increase the priority specified in the pragma.

CCN6425   **The value given for the "pragma priority" is in the range reserved for the system.**

Explanation

The priority specified in the pragma is in the range reserved for the system. This may cause unexpected behavior because the declaration may have a higher priority than system variables.

User response

Lower the specified priority.

CCN6426   **The function "%1$s" in "pragma alloc_text" is already specified. The pragma is ignored.**

Explanation

The pragma is ignored because the function has already been specified in a previous pragma alloc_text.

In the message text:

"%1$s" is the name of the function specified in the pragma.

User response

Remove the pragma.

CCN6427   **The specified object model "%1$s" is not known. The pragma is ignored.**

Explanation

The pragma is ignored because the object model is not recognized.

In the message text:

"%1$s" is the unrecognized object model.

User response

Change the specified object model to one that is known.

CCN6428   **The "pragma object_model" stack is empty. The pragma is ignored.**

Explanation

The pragma is ignored because the object model stack is empty.

User response

Remove the pragma or ensure that the stack is not empty.

CCN6429   **The identifier "%1$s" in "pragma import" is already specified on line %2$s of "%3$s". The pragma is ignored.**

Explanation

The pragma is ignored because the identifier has already been specified in a previous pragma import.

In the message text:

"%1$s" is the name of the repeated identifier and %2$s and "%3$s" are the coordinates of the previous pragma.

User response

Remove the pragma.

CCN6430   **The identifier "%1$s" in "pragma export" is already specified. The pragma is ignored.**

Explanation

The pragma is ignored because the identifier has already been specified in a previous pragma export.

In the message text:

"%1$s" is the name of the repeated identifier.

User response

Remove the pragma.

CCN6431   **The "pragma enum" stack is empty. The pragma is ignored.**

Explanation

The pragma is ignored because the pragma enum stack is empty.

User response

Remove the pragma or ensure that the pragma stack is not empty.

CCN6432   **The function "%1$s" in "pragma alloc_text" is already specified with "pragma code_seg".**

Explanation

The pragma is in conflict with a previous pragma code_seg.

In the message text:

"%1$s" is the name of the function indicated in the pragma.

User response

Remove the current or the previous pragma.

CCN6433   **The function "%1$s" in "pragma weak" is already specified. The pragma is ignored.**

Explanation

The pragma is ignored because it has already been specified in a pragma weak.

In the message text:

"%1$s" is the name of the function specified in the pragma.

User response

Remove the pragma.

CCN6434   **The message id "%1$s" in "pragma report" is not a valid. The pragma is ignored.**

Explanation

The pragma is ignored because the message id is not valid.

In the message text:

"%1$s" is the message id that must be changed.

User response

Change the message id.

CCN6435   **The function "%1$s" in "pragma mc_func" is already specified. The pragma is ignored.**

Explanation

The pragma is ignored because the function has already been specified in a pragma mc_func.

In the message text:

"%1$s" is the name of the function specified in the pragma.

User response

Remove the pragma.

CCN6436   **The function "%1$s" in "pragma reg_killed_by" is already specified. The pragma is ignored.**

Explanation

The pragma is ignored because the function has already been specified in a pragma reg_killed_by.

In the message text:

"%1$s" is the name of the function specified in the pragma.

User response

Remove the pragma.

CCN6437	"pragma reg_killed_by" must be used with a corresponding "pragma mc_func".

Explanation

The function specified in the pragma must have been previously specified in a pragma mc_func.

User response

Provide the pragma mc_func before the pragma reg_killed_by.

CCN6438	The file "%1$s" should be specified in an "#include" directive or as a source file in the configuration file.

Explanation

Informational message indicating that the file should be an included file or it should be specified in the configuration file.

In the message text:

"%1$s" is the name of the file that should be included.

User response

Ensure that the file is specified in an include directive.

CCN6439	Two or more expressions must be specified in "pragma disjoint". The pragma is ignored.

Explanation

The pragma is ignored because it must have two or more expressions specified.

User response

Ensure that at least two expressions are specified in the pragma.

CCN6440	The expressions "%1$s" and "%2$s" specified in "pragma disjoint" have incompatible types. The pragma is ignored.

Explanation

The pragma is ignored because the types specified in the two expressions are incompatible.

In the message text:

"%1$s" and "%2$s"are the two incompatible expressions, one of which must be changed.

User response

Change one of the expressions to have a compatible type with the other.

CCN6441	The expression "%1$s" specified in "pragma disjoint" is not a valid type. The pragma is ignored.

Explanation

The pragma is ignored because the type specified in the expression is not correct.

In the message text:

"%1$s" is the expression specifying the invalid type.

User response

Change the expression to specify a valid type.

CCN6442	The "pragma align" stack is empty. The pragma is ignored.

Explanation

The pragma is ignored because the pragma align stack is empty.

User response

Remove the pragma or ensure that the pragma align stack is not empty.

CCN6443	"pragma %1$s" overrides the original option value.

Explanation

Informational message indicating that the pragma is overriding the option value.

In the message text:

"%1$s" is the name of the pragma that is overriding the option value.

User response

See the primary message.

CCN6444	The "pragma namemangling" stack is empty. The pragma is ignored.

Explanation

The pragma is ignored because the pragma namemangling stack is empty.

User response

Remove the pragma or ensure that the pragma namemangling stack is not empty.

CCN6445	The size specified for "pragma pointer_size" must be 32 or 64. The pragma is ignored.

Explanation

The pragma is ignored because the size specified was not 32 or 64.

User response

Change the size specified to be 32 or 64.

CCN6446	The "pragma pointer_size" stack is empty.

Explanation

The pragma is ignored because the pragma pointer_size stack is empty.

User response

Remove the pragma or ensure that the pragma pointer_size stack is not empty.

CCN6447	The argument "%2$s" specified in "pragma %1$s" is not a defined class.

Explanation

The pragma is ignored because the argument does not specify a defined class.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the name of the class that must be defined.

User response

Change the argument or ensure that the class is defined.

CCN6448	"A pragma IsHome" is defined for "%1$s", but there is no matching "pragma HasHome". The pragma is ignored.

Explanation

The pragma is ignored because there must be a previously specified pragma HasHome for the argument.

In the message text:

"%1$s" is the argument that must have a corresponding pragma HasHome.

User response

Remove the pragma or ensure that there is a previous corresponding pragma HasHome.

CCN6449	More than one "pragma IsHome" for "%1$s" in different targets.

Explanation

There are more than one pragma IsHome specified for the arguments in different targets.

In the message text:

"%1$s" is the argument that has multiple pragma IsHome directives.

User response

Remove the extra pragma IsHome directives.

CCN6450	"pragma %1$s" has already been specified for function "%2$s". The pragma is ignored.

Explanation

The pragma is ignored because it has already been specified for the same function.

In the message text:

"%1$s" is the pragma name. "%2$s" is the qualified name of the function.

User response

Remove the pragma or change the declaration.

CCN6451	Using #pragma pack(%1$s), the C++ compiler may generate a different class layout than the C compiler generates. Use #pragma pack(%1$s:C_Compat) to get the C compiler's behavior.

Explanation

The class layout with #pragma pack(%1$s) may be different than the class layout generated by the C

compiler . Use #pragma pack(%1$s:C_Compat) to get the same class layout.

In the message text:

"%1$s" is the pragma pack value.

User response

Use #pragma pack(%1$s:C_Compat) to get the C compiler's behavior, but it may cause backward incompatibility with an older version of the C++ compiler.

CCN6455	Member "%1$s" is not declared as specified in pragma "%2$s". The pragma is ignored.

Explanation

The declaration of the member in the pragma does not match the declaration for that member in the member's class.

In the message text:

"%1$s" is the class member. "%2$s" is the name of the pragma.

User response

Fix the declaration in the pragma or remove the pragma.

CCN6456	Only dot member access is allowed in pragma "%1$s". The pragma is ignored.

Explanation

Pragma "%1$s" is only allowed to use class member access with the dot operator.

In the message text:

"%1$s" is the name of the pragma.

User response

Change the pragma to use dot member access or remove the pragma.

CCN6457	Member "%1$s" is at offset "%2$s", not at offset "%3$s" as specified in pragma assert_field_offset.

Explanation

The assertion in the pragma assert_field_offset has been violated. The member is not at the specified offset.

In the message text:

"%1$s" is the member name. "%2$s" is the actual offset. "%3$s" is the offset specified in the pragma.

User response

Fix the offset or remove the pragma.

CCN6460	"%1$s" has not been declared before the pragma pointer directive.

Explanation

"%1$s" must be declared before the pragma.

In the message text:

"%1$s" is the type.

User response

Add a declaration for "%1$s" before the pragma or remove the pragma.

CCN6461	"%1$s" is not a 16 byte void pointer.

Explanation

The pragma has an argument.

In the message text:

"%1$s" is the argument to the pragma.

User response

Fix the argument to the pragma or remove the pragma.

CCN6463	"%1$s" has been used in a declaration, the pragma is ignored.

Explanation

The name has already been used previously and cannot be used again by the pragma.

In the message text:

"%1$s" is the argument to the pragma.

User response

Fix the argument to the pragma or remove the pragma.

CCN6464	"%1$s" is not a typedef name.

Explanation

The pragma requires a typedef name as an argument and the one provided is not one.

In the message text:

"%1$s" is the argument to the pragma.

User response

Fix the argument to the pragma or remove the pragma.

CCN6465	Instruction sequence for "pragma mc_func" contains the character "%1$s" that is not a hexadecimal digit.

Explanation

The pragma requires a hexadecimal argument and one has not been provided.

In the message text:

"%1$s" is the invalid character specified in the pragma.

User response

Fix the instruction sequence for the pragma or remove the pragma.

CCN6466	Instruction sequence for "pragma mc_func" contains odd number of hexadecimal digits.

Explanation

The pragma requires an argument which is an instruction sequence consisting of an even number of hexadecimal digits.

User response

Fix the instruction sequence for the pragma or remove the pragma.

CCN6467	The include directive for the primary source file "%1$s" is ignored.

Explanation

It was not possible for the compiler to process the file as a primary source file.

In the message text:

"%1$s" is the source file name.

User response

Remove the include directive from the configuration file.

CCN6469	The function "%2$s" specified in "pragma %1$s" cannot be found.

Explanation

Name lookup failed for the function specified in the pragma.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the function name.

User response

Fully qualify the function, specify a different function, or remove the pragma.

CCN6470	The source file "%1$s" is being included by the source file "%2$s", which has different options in effect.

Explanation

The source file "%1$s" has been specified as a primary source file in the configuration file and it's options do not match the options specified by another primary source file that includes "%1$s".

In the message text:

"%1$s' is the included source file. ""%2$s" is the source file including "%1$s""

User response

Change the options to be consistent or change "1s" to not be a primary source file.

CCN6492	No argument is specified for "pragma define". The pragma is ignored.

Explanation

The pragma requires an argument and one was not specified.

User response

Specify an argument or remove the pragma.

CCN6493	Duplicate argument "%1$s" in "pragma disjoint". The pragma is ignored.

Explanation

The argument indicated was duplicated in the argument list specified for the pragma.

In the message text:

"%1$s" is the duplicate argument specified in the pragma.

User response

Remove the duplicate argument or remove the pragma.

CCN6494	The suboption "%1$s" for "pragma %2$s" is not supported on the target platform. The pragma is ignored.

Explanation

The suboption for the pragma indicated is not supported on this operating system.

In the message text:

"%1$s" is the name of the suboption that is unsupported. "%2$s" is the name of the pragma.

User response

Specify a different suboption or remove the pragma.

CCN6495	Unexpected text "%2$s" found in "pragma %1$s". The pragma is ignored.

Explanation

A syntax error has been found while processing the pragma, causing it to be ignored.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the text causing the syntax error.

User response

Fix the syntax of the pragma or remove the pragma.

CCN6496	Unexpected text "%2$s" found in "pragma %1$s". The rest of the pragma directive is ignored.

Explanation

A syntax error has been found while processing part of the pragma, causing part of it to be ignored.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the text causing the syntax error.

User response

Fix the syntax of the pragma or remove the pragma.

CCN6497	An implicit "}" does not find a matching implicit 'extern "C" {'. An extra "}" may be present.

Explanation

An unmatched "}" was detected while processing a linkage specification.

User response

Remove the extra "}" if one exists.

CCN6498	The function "%2$s" specified in "pragma %1$s" has already been defined. The pragma is ignored.

Explanation

The pragma specified must be placed before the definition of the function to which it refers.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the name of the function.

User response

Move the pragma to before the definition of the function or remove the pragma.

CCN6499	The function "%2$s" specified in "pragma %1$s" is virtual. The pragma is ignored.

Explanation

The pragma specified requires an argument that is not a virtual function.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the name of the function.

User response

Change the pragma to specify a non-virtual function or remove the pragma.

CCN6600	"main" should have a return type of type "int".

Explanation

A return type other than "int" has been specified for "main".

User response

Change the return type of "main" to be "int".

CCN6601   **A local class cannot have member templates.**

Explanation

Member templates can only be defined in namespace scope classes.

User response

Remove the template from the local class, or move the class to non-local scope.

CCN6602   **The partial specialization "%1$s" cannot have template parameters that have default values.**

Explanation

Default template arguments are not allowed on partial specializations.

In the message text:

"%1$s" is the partial specialization.

User response

Remove the default template arguments.

CCN6603   **Default template parameter arguments cannot be followed by uninitialized template parameters.**

Explanation

Just like function parameters, all template parameters following a template parameter with a default argument must also have default arguments.

User response

Add the missing default arguments or remove the existing one.

CCN6604   **The template parameter "%1$s" cannot be used in a partially specialized non-type argument expression.**

Explanation

The use of a template parameter in an expression for a non-type template argument in partial specialization is not allowed.

In the message text:

"%1$s" is the template parameter.

User response

Correct the non-type template argument expression.

CCN6605   **The argument list for the partial specialization "%1$s" is equivalent to the implicit argument list of the primary template.**

Explanation

A partial specialization must specialize something in the argument list.

In the message text:

"%1$s" is the partial specialization.

User response

Change the argument list of the partial specialization.

CCN6606   **The non-type template parameter, "%1$s", must have an integral, enumeration, pointer to object, pointer to function, lvalue reference, or pointer-to-member type.**

Explanation

No other types are allowed.

In the message text:

"%1$s" is the non-type template parameter.

User response

Correct the non-type template parameter type.

CCN6607   **All array dimensions for "%1$s" should be specified and should be greater than zero.**

Explanation

An array dimension is missing or is negative.

In the message text:

"%1$s" is the array.

User response

Ensure that all dimensions are specified as non-negative numbers.

CCN6608   **An anonymous %1$s should only define non-static data members.**

Explanation

Static data members and non-data members are not allowed in anonymous aggregates.

In the message text:

%1$s is the keyword union, struct, or class.

User response

Remove the member declaration.

CCN6609	A using declaration cannot be used to declare "%1$s".

Explanation

The using declaration cannot be used here.

In the message text:

"%1$s" is the declarator.

User response

Remove the using declaration.

CCN6610	"%1$s" must not be declared as import and defined.

Explanation

The "_Import" specifier cannot be specified on a definition.

In the message text:

"%1$s" is the function.

User response

Remove the "_Import" specifier.

CCN6611	The current option settings do not allow the use of "long long".

Explanation

The declaration type is "long long" but this type is disallowed due to option settings.

User response

Change the type of the declaration or the option settings to allow "long long".

CCN6612	The static variable "%1$s" is not visible where "%2$s" is used in a #include directive.

Explanation

A static variable is being referenced in an include file and is not visible.

In the message text:

"%1$s" is the static variable. "%2$s" is the header file.

User response

Remove the static specifier from the declaration.

CCN6613	The static function "%1$s" is not visible where "%2$s" is used in a #include directive.

Explanation

A static function is being referenced in an include file and is not visible.

In the message text:

"%1$s" is the static function. "%2$s" is the header file.

User response

Remove the static specifier from the declaration.

CCN6614	"%1$s" must be the last data member in its class because "%2$s" contains a zero-dimension array.

Explanation

Only the last non-static data member can have a zero dimension.

In the message text:

"%1$s" is the member. "%2$s" is the union, struct, or class.

User response

Move the declaration to be the last in the class.

CCN6615	Only the first array bound can be omitted.

Explanation

For a multi-dimensional array, the compiler can determine the size of the first bound based on the number of initializers. It is unable to compute any other omitted array bounds.

User response

Specify all array bounds or leave only the first bound unspecified.

CCN6616   **A pointer-to-member should not be converted from the virtual base "%1$s" to the derived class "%2$s".**

Explanation

The conversion is from a virtual base class to a derived class.

In the message text:

"%1$s" is the virtual base. "%2$s" is the derived class.

User response

See the primary message.

CCN6617   **The incomplete type "%1$s" is not allowed in an exception-specification .**

Explanation

Only complete types are allowed in an exception-specification.

In the message text:

"%1$s" is the incomplete type.

User response

Correct the exception specification type list.

CCN6618   **"%1$s" is not allowed in an exception-specification because "%2$s" is incomplete.**

Explanation

Only pointers to complete types are allowed in pointer exception-specification types.

In the message text:

"%1$s" is the pointer type. "%2$s" is the incomplete type.

User response

Correct the exception-specification type list.

CCN6619   **The type "%1$s" is not valid in this context.**

Explanation

The type "void" is not valid for this declaration.

In the message text:

"%1$s" is the type.

User response

Change the type.

CCN6620   **"%1$s" must be declared to have "stdcall" linkage.**

Explanation

The "stdcall" specifier must be specified.

In the message text:

"%1$s" is the function.

User response

Add the "stdcall" specifier.

CCN6621   **The explicit specialization "%1$s" must be declared in the nearest enclosing namespace scope of the template.**

Explanation

The explicit specialization declaration must be in the namespace scope of the nearest enclosing namespace of the primary template or a namespace in the enclosing namespace set.

In the message text:

"%1$s" is the explicit specialization.

User response

Move the explicit specialization declaration to a correct scope.

CCN6622   **The explicit specialization "%1$s" must be defined in a namespace that encloses the declaration of the explicit specialization.**

Explanation

An explicit specialization must be defined at namespace scope, in the same or an enclosing namespace as the declaration.

In the message text:

"%1$s" is the explicit specialization.

User response

Move the explicit specialization definition to the correct scope.

CCN6623	The explicit specialization "%1$s" cannot have default function arguments.

Explanation

Default function arguments are not allowed on an explicit specialization.

In the message text:

"%1$s" is the explicit specialization.

User response

Remove the default function arguments.

CCN6624	The partial specialization "%1$s" must be declared in the same scope as the primary template or in a namespace scope that encloses the primary template.

Explanation

A partial specialization declaration must be in the same scope or in an enclosing namespace scope of the primary template.

In the message text:

"%1$s" is the partial specialization.

User response

Move the partial specialization declaration to the correct scope.

CCN6625	The explicit specialization "%1$s" must not be declared in the scope of a template.

Explanation

An explicit specialization must be declared in the namespace containing the primary template.

In the message text:

"%1$s" is the explicit specialization.

User response

Remove the explicit specialization.

CCN6626	At least one template argument in a partial specialization must depend on a template parameter.

Explanation

A partial specialization cannot be fully specialized.

User response

Change the declaration to an explicit specialization or change the template arguments to be partially specialized.

CCN6627	A statement type that follows the pragma "%1$s" is invalid. The pragma is ignored.

Explanation

A valid statement type is required to be followed by the pragma.

In the message text:

"%1$s" is the name of the pragma.

User response

Use the correct statement type or remove the pragma.

CCN6628	Every template parameter for a constructor template must be used in the parameter list of the constructor.

Explanation

There is no way to specify an explicit template argument list for a constructor template.

User response

Change the template parameter list of the constructor template.

CCN6629	Every template parameter for a conversion function template must be used in the return type.

Explanation

There is no way to specify an explicit template argument list for a conversion function template.

User response

Change the template parameter list of the conversion function template

CCN6630	Every template parameter for a partial specialization must be used in the template argument list.

Explanation

The extra template parameters are not used so they are not allowed.

User response

Change the parameter list of the partial specialization.

CCN6631	A template parameter should not be used in its own default argument.

Explanation

A template parameter can be used in subsequent template parameters and their default arguments.

User response

Change or remove the default argument.

CCN6632	The length of the identifier exceeds the maximum limit of "%1$s" for a name with "%2$s" linkage.

Explanation

The identifier name is too large.

In the message text:

"%1$s" is the maximum permitted identifier length. "%2$s" is the linkage specifier.

User response

Replace the identifier with a smaller identifier.

CCN6633	The name "%1$s" is not a recognized built-in declaration.

Explanation

The function specified is not a built-in function.

In the message text:

"%1$s" is the function name.

User response

Change the declaration so that it does not specify that the function is built in.

CCN6634	An array element must not have type "%1$s".

Explanation

The type of the array is invalid.

In the message text:

"%1$s" is the type.

User response

Change the type of the array.

CCN6635	There cannot be a reference to a reference.

Explanation

A reference to a reference is invalid.

User response

Remove the extra reference.

CCN6636	There cannot be a pointer to a reference.

Explanation

A pointer to a reference is invalid.

User response

Change the declaration.

CCN6637	There cannot be a pointer-to-member with reference type.

Explanation

A pointer to a member reference is invalid.

User response

Change the declaration.

CCN6638	There cannot be an array of references.

Explanation

The element type of an array cannot be a reference type, void type, function type, or an abstract class type.

User response

Change the element type of the array to a valid type.

CCN6639	The behavior of long type bit fields has changed from previous releases of this compiler. In 64-bit mode, long type bit fields now default to long, not int.

Explanation

The bit field will default to long, this is a change in behavior.

User response

No response required.

CCN6640   **Cannot take the address of the machine-coded function "%1$s".**

Explanation

It is invalid to take the address of a machine-coded function.

In the message text:

"%1$s" is the function.

User response

Change the expression.

CCN6641   **The use of vector type is invalid for architecture level "%1$s".**

Explanation

The effective architecture value specified on either #pragma arch_section or a compiler option does not support vector.

In the message text:

"%1$s" is the architecture level specified on #pragma arch_section or a target architecture option.

User response

Use architecture which supports vector processing, or remove the use of vector.

CCN6642   **The packed attribute is valid only for class and struct nonstatic data members. The attribute is ignored.**

Explanation

The packed attribute has no effect on static members or function or namespace scoped variables.

User response

Remove the packed attribute.

CCN6643   **"main" cannot be declared as a template function.**

Explanation

"main" implicitly has "C" linkage; a template function may not have "C" linkage.

User response

Do not define "main" as a template function.

CCN6644   **The unnamed bit field is too small: %1$s bits are needed for "%2$s".**

Explanation

The size of the unnamed bit field is not large enough to contain all of the possible values.

In the message text:

%1$s is the number of bits. "%2$s" is the name of the enumerated type.

User response

Increase the size of the unnamed bit field.

CCN6645   **The bit field "%1$s" is too small: %2$s bits are needed for "%3$s".**

Explanation

The size of the bit field is not large enough to contain all of the possible values.

In the message text:

"%1$s" is the bit field. %2$s is the number of bits. "%3$s" is the name of the enumerated type.

User response

Increase the size of the bit field.

CCN6646   **The explicit instantiation of member "%1$s" must have a definition.**

Explanation

The definition must be available in order for an instantiation to be done.

In the message text:

"%1$s" is the member.

User response

Define the static member.

CCN6648   **Clause "%1$s" is not valid in "# pragma omp %2$s".**

Explanation

The clause is not valid in #pragma omp.

In the message text:

"%1$s" is the omp clause "%2$s" is the specifier of #pragma omp.

User response

Remove the clause.

CCN6649	The "%1$s" clause was already specified.

Explanation

The clause was specified and should not be re-specified.

In the message text:

"%1$s" is the omp clause.

User response

Remove the clause.

CCN6650	"# %1$s %2$s" is not valid for the "%3$s" statement.

Explanation

#pragma %1$1 is not valid for the statement.

In the message text:

"%1$1s" is the pragma construct name. "%2$s" is the specifier of #pragma construct. "%3$s" is the statement.

User response

Remove the #pragma %1$1 or change the statement.

CCN6652	"%1$s" is not allowed in a structured block.

Explanation

The statement is not allowed in a structured block

In the message text:

"%1$s" is the statement.

User response

Remove the statement.

CCN6653	Branching out of a structured block is not allowed.

Explanation

The label statement must be within the lexical block

User response

Don't branch out of a structured block

CCN6654	Branching into a structured block is not allowed.

Explanation

The label statement must be out of the lexical block

User response

Don't branch in a structured block

CCN6655	The for-init-statement is missing, the for loop is not in the canonical form.

Explanation

The for-init-statement is missing, the for loop is not in the canonical form.

User response

Check the for-init-statement.

CCN6656	The for-init-statement of the for loop is not in the canonical form.

Explanation

The for-init-statement of the for loop is not in the canonical form.

User response

Check the for-init-statement.

CCN6657	The iteration variable must be a signed or unsigned integer or random access iterator type.

Explanation

The iteration variable must be a signed or unsigned integer or random access iterator type.

User response

Check if the iteration variable is a signed or unsigned integer or random access iterator type.

CCN6658	The condition is missing, the for loop is not in the canonical form.

Explanation

The condition is missing, the for loop is not in the canonical form.

User response

Add the condition in the for loop.

CCN6659	The condition of the for loop is not in the canonical form.

Explanation

The condition of the for loop is not in the canonical form.

User response

Check the condition in the for loop.

CCN6660	The increment expression is missing, the for loop is not in the canonical form.

Explanation

The for loop is missing the increment expression.

User response

Add the increment expression in the for loop.

CCN6661	The increment expression of the for loop is not in the canonical form.

Explanation

The increment expression of the for loop is not in the canonical form.

User response

Check the increment expression in the for Loop.

CCN6663	Incorrect assignment of a restrict qualified pointer. Only outer-to-inner scope assignments between restrict pointers are allowed. This may result in incorrect program behavior.

Explanation

Only outer-to-inner scope assignments between restrict pointers are allowed.

User response

Check the assignment.

CCN6664	The variable "%1$s" has undefined data scope.

Explanation

The variable should have a defined data scope.

In the message text:

"%1$s" is the variable.

User response

Specify a data scope for the variable.

CCN6665	The value of the expression must be greater than zero.

Explanation

The value of the expression must be greater than zero.

User response

Change the chunk size to a positive value.

CCN6666	An "ordered" directive must be within a dynamic extent of a "for" or "parallel for" construct.

Explanation

An "ordered" directive must be within a dynamic extent of a "for" or "parallel for" construct.

User response

Check if the "ordered" directive is within a "for" or "parallel for" construct.

CCN6667	The related "for" or "parallel for" construct must have an "ordered" clause.

Explanation

The related "for" or "parallel for" construct must have an "ordered" clause.

User response

Check if the related "for" or "paraller for" construct have an "ordered" clause.

CCN6668	The "ordered" directive must not be executed more than once.

Explanation

The "ordered" directive must not be executed more than once.

User response

Check if the "ordered" directive is executed more than once.

CCN6670	Invalid statement type in "atomic" construct.

Explanation

Invalid statement type in "atomic" construct.

User response

Change the statement type in "atomic" construct.

CCN6671	Invalid expression type in "atomic" construct.

Explanation

Invalid expression type in "atomic" construct.

User response

Check the expression type in "atomic" construct.

CCN6672	Expression in "atomic" construct is not scalar type.

Explanation

Expression in "atomic" construct is not scalar type.

User response

Change the expression to scalar type.

CCN6673	The variable "%1$s" has already been specified in one of the data scope clauses.

Explanation

The same variables appear on same named clauses.

In the message text:

"%1$s" is the variable

User response

Change the variable.

CCN6674	The variable "%1$s" must not have a reference type.

Explanation

A reference type is not applied for the variable.

In the message text:

"%1$s" is the variable.

User response

Change the variable's type.

CCN6675	The variable "%1$s" must not have an incomplete type.

Explanation

The variable should have a complete type.

In the message text:

"%1$s" is the variable.

User response

Check the variable's type.

CCN6676	The class of variable "%1$s" must have a default constructor.

Explanation

The class should have a default constructor.

In the message text:

"%1$s" is the class name.

User response

Add a default constructor in the class.

CCN6677	The __callback keyword is not associated with a function pointer.

Explanation

The __callback keyword is restricted to qualify function pointers.

User response

Change the declaration or remove the __callback keyword.

CCN6678	Critical constructs with the same name cannot be nested.

Explanation

Critical directives with the same name are not allowed to be nested inside each other.

User response

Change the name of a critical directive.

CCN6679	"%1$s" cannot be nested within "%2$s".

Explanation

Some directives are not permitted in the dynamic extent of other directives.

In the message text:

"%1$s" is the inside directive name, "%2$s" is the outside directive name.

User response

Check both directives.

CCN6680	The smallest statement that contains a "%1$s" directive must be a block.

Explanation

The smallest parent statement of the directive must be a block.

In the message text:

"%1$s" is the directive name.

User response

Remove the directive or change the parent statement to a block.

CCN6682	The variable "%1$s" must not have a pointer type.

Explanation

The variable can not have a pointer type.

In the message text:

"%1$s" is the variable name.

User response

Check the declaration.

CCN6683	The variable "%1$s" is already listed in a reduction clause.

Explanation

The variable should not be listed in a reduction clause twice.

In the message text:

"%1$s" is the variable name.

User response

Don't list the variable in a reduction clause more than once.

CCN6684	Variable "%1$s" must be shared in the enclosing context.

Explanation

Variable listed in the reduction clause must be shared in the enclosing context.

In the message text:

"%1$s" is the variable name.

User response

Check the scope of the variable in the reduction clause.

CCN6685	Variable "%1$s" must not be listed in both a shared and a reduction clause.

Explanation

A variable must not be listed in a "shared" clause.

In the message text:

"%1$s" is the variable name.

User response

Check the scope of the variable in the reduction clause.

CCN6686	Variable "%1$s" must not be const-qualified.

Explanation

The variable should not be const-qualified.

In the message text:

"%1$s" is the variable name.

User response

Remove the const qualifier of the variable.

CCN6687	The type of variable "%1$s" is not valid for the reduction operator.

Explanation

The variable must not be: a reference, a pointer, or const-qualified.

In the message text:

"%1$s" is the variable name.

User response

Change the declaration of the variable.

CCN6688	The copy assignment operator of class "%1$s" in OMP "%2$s" clause does not exist.

Explanation

Missing the copy assignment operator of the class.

In the message text:

"%1$s" is the class name, "%2$s" is the OMP clause.

User response

Add a copy assignment operator in the class.

CCN6689	The variable "%1$s" is not a threadprivate variable.

Explanation

The variable is not a threadprivate variable.

In the message text:

"%1$s" is the variable name.

User response

Change the variable to a threadprivate variable.

CCN6690	The variable "%1$s" cannot have a const-qualified type in the OMP "%2$s" clause.

Explanation

The variable must not have a const-qualified type in the OMP clause.

In the message text:

"%1$s" is the variable name, "%2$s" is the OMP clause.

User response

Remove the const-qualified type of the variable.

CCN6691	"%1$s" cannot be nested within "%2$s" when both bind to the same parallel region.

Explanation

The constructs that bind to the same parallel are not allowed to be nested inside each other.

In the message text:

"%1$s" is the inside construct name, "%2$s" is the outside construct name.

User response

Do not nest the constructs.

CCN6692	The class "%1$s" of threadprivate variable "%2$s" declared with an explicit initializer must have a copy constructor.

Explanation

The class declared with an explicit initializer should have a copy constructor.

In the message text:

"%1$s" is the class name, "%2$s" is the variable name.

User response

Add a copy constructor in the class.

CCN6693	Local labels can only be preceded by other local label declarations in a lexical block.

Explanation

The local label declaration is not the first statement in the lexical block or it is not strictly preceded in the lexical block by local label declarations.

User response

Move the local label declaration before any other statements in the lexical block.

CCN6694	The class of variable "%1$s" has an ambiguous default constructor.

Explanation

The default constructor of the class is ambiguous.

In the message text:

"%1$s" is the variable name.

User response

Check the default constructor of the class.

CCN6695   **The class of variable "%1$s" has an ambiguous copy constructor.**

Explanation

The assignment operator of the class is ambiguous.

In the message text:

"%1$s" is the variable name.

User response

Check the copy constructor of the class.

CCN6696   **The class of variable "%1$s" has an ambiguous copy assignment operator.**

Explanation

The assignment operator of the class is ambiguous.

In the message text:

"%1$s" is the variable name.

User response

Check the assignment operator of the class.

CCN6697   **The copy constructor of class "%1$s" in the OMP "%2$s" clause does not exist.**

Explanation

Missing copy constructor of a class in the OMP clause.

In the message text:

"%1$s" is the class name, "%2$s" is the clause name.

User response

Add a copy constructor in the class.

CCN6698   **The current option settings do not allow the use of "%1$s".**

Explanation

The "%1$s" is not supported by the option.

In the message text:

"%1$s" is the unsupported feature.

User response

Use the option to support "%1$s".

CCN6699   **Pragma "%1$s" may not be supported in next releases. Pragma "%2$s" provides the same functionality, and should be used.**

Explanation

An old sub-option was used with the pragma. The sub-option may not be supported in next releases.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the name of another pragma.

User response

Use a new sub-option which provides the same functionality.

CCN6700   **Reduction variable "%1$s" may not be accessed in an explicit task.**

Explanation

Reduction variable cannot be accessed in an explicit task.

User response

No user response is required.

CCN6800   **The divisor for the modulus or division operator must not be zero.**

Explanation

A division-by-zero condition has been detected.

User response

Change the expression.

CCN6801   **The result of expression evaluation resulted in an overflow.**

Explanation

An overflow condition has been detected.

User response

Change the expression.

CCN6802   **The result of expression evaluation resulted in an underflow.**

Explanation

An underflow condition has been detected.

User response

Change the expression.

CCN7500   **The option "%1$s" is not supported.**

Explanation

The command line contained an option that is not supported. Note that some option parameters must not have spaces between the option and the parameter.

In the message text:

"%1$s" is an option.

User response

Remove the option. Check the syntax of the options.

CCN7501   **Suboption "%1$s" for option "%2$s" is not supported on the target platform.**

Explanation

The option has been specified with a suboption that is not supported on the target platform.

In the message text:

"%1$s" is the suboption. "%2$s" is the option.

User response

Change the suboption, or remove the option.

CCN7502   **Missing value for option "%1$s".**

Explanation

The option was missing a required parameter. See the "User's Guide" for details on the option.

In the message text:

"%1$s" is an option name

User response

Add a value for the option.

CCN7503   **Unrecognized value "%1$s" specified with option "%2$s".**

Explanation

An inappropriate value was used with the option.

In the message text:

"%1$s" is the value specified with the option,. "%2$s" is the option name.

User response

Remove the unrecognized value.

CCN7504   **"%1$s" is not a valid suboption for "%2$s". The option is ignored.**

Explanation

The command line contained an option with an invalid suboption.

In the message text:

"%1$s" is the suboption, "%2$s" is the option.

User response

Remove the suboption.

CCN7505   **The value given for the "priority" option is in the range reserved for the system.**

Explanation

Priority values less than -2147482624 are reserved for system purposes.

User response

Change the priority value so that it is greater than -2147482624.

CCN7506   **"%1$s" is no longer supported. The option is ignored.**

Explanation

The command line contained an option that is no longer supported by this release.

In the message text:

"%1$s" is the outdated option.

User response

Remove the option.

CCN7507   **Options "%1$s" and "%2$s" are not compatible.**

Explanation

The specified options cannot be used together.

In the message text:

"%1$s" and "%2$s" are both option names.

User response

Change option values.

CCN7508   Suboption "%1$s" for option "%2$s" is no longer supported. The suboption is ignored.

Explanation

The command line contained a suboption that is no longer supported by this release.

In the message text:

"%1$s" is the suboption. "%2$s" is the option.

User response

Remove the suboption.

CCN7509   The suboption specified for the "%1$s" option is not allowed when the "%2$s" option is specified.

Explanation

The suboption specified in the first option conflicts with the second option. The first option is ignored.

In the message text:

"%1$s" and "%2$s" are option names.

User response

Correct the conflicting option or suboption.

CCN7510   Insufficient memory.

Explanation

The available memory has been exhausted.

User response

Provide more memory.

CCN7511   Either the default or user-defined maximum number of error messages has been exceeded.

Explanation

There have been too many errors to continue.

User response

Fix the previous errors.

CCN7512   Compiler cannot create temporary files. The file system may be full or not writable.

Explanation

The intermediate code files could not be created. Please verify that the target file system exists, is writable, and is not full.

User response

Ensure that the designated location for temporary objects exists, is writable, and is not full.

CCN7513   An error was detected while writing to an temporary file. The file system may be full.

Explanation

An error occurred writing to an intermediate code file. Please verify that the target file system exists, is writable, and is not full.

User response

Ensure that the designated location for temporary objects exists, is writable, and is not full.

CCN7517   The template registry file "%1$s" could not be opened.

Explanation

A template registry file is created when the templateregistry compiler option is enabled.

In the message text:

"%1$s" is the template registry file name designated by the templateregistry compiler option.

User response

Ensure that file system permissions allow files to be written, and that sufficient file system resources exist to permit the creation of this file.

CCN7518   Error reading template registry file "%1$s".

Explanation

The template registry file is corrupt.

In the message text:

"%1$s" is the template registry file name designated by the templateregistry compiler option

User response

Delete the template registry file and recompile all of the source files using this registry.

CCN7519   **Error writing to template registry file "%1$s".**

Explanation

A template registry file is created when the templateregistry compiler option is enabled.

In the message text:

"%1$s" is the template registry file name designated by the templateregistry compiler option.

User response

Ensure that file system permissions allow files to be written, and that sufficient file system resources exist to permit the creation of this file. If you receive this error while using an existing templateregistry file from a previous compilation, delete the templateregistry file, then recompile your source.

CCN7520   **""%1$s""**

Explanation

This is a generic message.

In the message text:

"%1$s" is the message.

User response

The primary message describes a unique situation. All information should be found there.

CCN7521   **The template definition "%1$s" is no longer provided in module "%2$s". Dependent modules should be recompiled to generate the necessary definition.**

Explanation

A template definition is no longer available in the current module.

In the message text:

"%1$s" is the template definition and "%2$s" is the module.

User response

Recompile dependent modules to regenerate the template definition.

CCN7522   **The compiler is operating in 32-bit mode. The option "%1$s" is ignored.**

Explanation

This option is valid only in 64-bit mode.

In the message text:

"%1$s" is the option value.

User response

Remove the unrecognized value.

CCN7524   **"%1$s" is not compatible with "%2$s". "%3$s" is being set.**

Explanation

The option is not compatible with another option so it is ignored.

In the message text:

"%1$s", "%2$s" and "%3$s" are all option names.

User response

Remove one of the options.

CCN7525   **Sub-option is not allowed in the "%1$s" option.**

Explanation

Sub-option is not allowed in the specified option.

In the message text:

"%1$s" is the option name.

User response

Remove the sub-option.

CCN7526   **Option "%1$s" does not have a negative form and is ignored.**

Explanation

This option cannot be used in negative form.

In the message text:

"%1$s" is the option name.

User response

Change the negative form to positive by removing "no" string from the specified option or remove the option altogether.

CCN7527   **Invalid delimiter following the string "%1$s".**

Explanation

The delimiter used to separate one string from another is not valid.

In the message text:

"%1$s" is the string preceding invalid delimiter.

User response

The valid string delimiter is character ':'. Please use the valid delimiter.

CCN7528   **Invalid string "%1$s" specified and is ignored.**

Explanation

Refer to the option specification to review which characters are allowed to form a valid string for this option.

In the message text:

"%1$s" is the invalid string.

User response

Remove or correct the invalid string.

CCN7529   **Unable to access options file "%1$s".**

Explanation

The compiler could not access the specified options file. It was either unable to open it or unable to read it.

In the message text:

"%1$s" is the options file name specified on OPTFILE option.

User response

Ensure the options file name and other specifications are correct. Ensure that the access authority is sufficient. Ensure that the file being accessed has not been corrupted.

CCN7530   **Options file "%1$s" is already specified. All subsequent occurrences are ignored.**

Explanation

An options file can only be specified once.

In the message text:

"%1$s" is the options file name.

User response

Ensure that the options file is specified only once.

CCN7531   **Options file "%1$s" is too big. The current size is %2$s bytes.**

Explanation

The options file exceeded the size limit.

In the message text:

"%1$s" is the options file name. %2$s is the size of options file in bytes.

User response

Reduce the size of the options file and rerun.

CCN7532   **No matching %1$s quote in "%2$s".**

Explanation

The specified string has missing quote which makes the string invalid.

In the message text:

%1$s is a quote character. "%2$s" is the string with missing quote.

User response

Remove the string or add the missing quote to make the string valid.

CCN7533   **No matching parenthesis in "%1$s".**

Explanation

The specified string has missing matching parenthesis which makes the string invalid.

In the message text:

"%1$s" is the string with missing parenthesis.

User response

Remove the string or add the missing parenthesis to make the string valid.

CCN7534   **The "pragma %1$s" stack is empty. The pragma is ignored.**

Explanation

The pragma is ignored because the "pragma %1$s" stack is empty.

In the message text:

"pragma %1$s" is a keyword.

User response

Remove the pragma or ensure that the "pragma %1$s" stack is not empty.

CCN7535　　　**The attribute "%1$s" redeclared with different value.**

Explanation

The attribute "%1$s" redeclared with different value.

In the message text:

"%1$s" is an attribute name.

User response

Remove the attribute specifier.

CCN7536　　　**The declaration "%1$s" has greater visibility than the "%2$s".**

Explanation

The declaration "%1$s" has greater visibility than the "%2$s".

In the message text:

"%1$s" and "%2$s" are declaration names.

User response

Remove the attribute specifier.

CCN7537　　　**There might be performance loss if "%1$s" has "%2$s" visibility and specified by "%3$s".**

Explanation

If any of the functions that are marked as imported resolve to statically bound objects, there will be performance loss.

In the message text:

"%1$s" is the symbol name, "%2$s" is one of the protected/hidden/internal visibility attribute, %3$s is option name

User response

The generated code may be larger and run more slowly than the default code sequence generated for functions.

CCN7538　　　**The name mangling for "%1$s" is not implemented.**

Explanation

The expression is used in a context where it needs to be mangled; however, there is no mangled form implemented for one of the language constructs used therein.

In the message text:

"%1$s" is the expression that needs to be mangled.

User response

Change the source code so that "%1$s" can be mangled.

CCN7539　　　**"%1$s" is reserved as a keyword. To disable it, use "%2$s".**

Explanation

A C++0x reserved keyword is present in C++98/03 code.

In the message text:

%1$s is the C++0x keyword, "%2$s" is the option to disable the keyword.

User response

Disable the keyword or do not use it.

CCN7541　　　**The value given for priority option is not in the range "%1$s" - "%2$s". It has been reset to %3$s.**

Explanation

priority values must be in the range "%1$s" - "%2$s".

User response

Change the priority option value so that it is in the range "%1$s" - "%2$s".

CCN7542　　　**FUNCEVENT(ENTRYCALL) option cannot take function name with template arguments. FUNCEVENT(NOENTRYCALL) is set.**

Explanation

FUNCEVENT(ENTRYCALL) option cannot distinguish between different function template instantiations.

User response

To generate entry calls for template function, specify FUNCEVENT(ENTRYCALL) to generate entry calls for all functions.

CCN7599	The compiler could not open the output file "%1$s".

Explanation

The file "%1$s" could not be opened.

In the message text:

%1$s is a file name.

User response

Ensure the output file name is correct. Also, ensure that the location of the output file has sufficient storage available. If using a network file system, ensure that the network is working properly and you have permission to write to the file system.

CCN7601	Goto statements should not be used.

Explanation

Goto statements often lead to difficult to maintain code.

User response

Remove the goto statements.

CCN7602	Ellipsis notation should not be used.

Explanation

Using ellipsis prevents type checking of arguments.

User response

Use an explicit argument list.

CCN7607	"%1$s" should probably define a constructor.

Explanation

"%1$s" does not have a constructor defined.

In the message text:

"%1$s" is a class name.

User response

Define a constructor for "%1$s".

CCN7608	"%1$s" should probably define a destructor.

Explanation

"%1$s" does not have a destructor defined.

In the message text:

"%1$s" is a class name.

User response

Define a destructor for "%1$s".

CCN7609	"%1$s" should probably define a copy constructor.

Explanation

"%1$s" does not have a user defined copy constructor.

In the message text:

"%1$s" is a class name.

User response

Define a copy constructor for "%1$s".

CCN7611	Argument "%1$s" is not used in function "%2$s".

Explanation

The argument "%1$s" is specified but not needed.

In the message text:

"%1$s" is an argument and "%2$s" is the name of the function.

User response

Consider removing the argument from the paramater list of the function.

CCN7612	"%1$s" is set but not used in function "%2$s".

Explanation

A variable has been explicitly initialized or assigned but is not referenced.

In the message text:

"%1$s" is the variable that is set but not used and "%2$s" is the function where the variable resides.

User response

Remove the variable if there are no side-effects.

CCN7613	The destructor in the base class "%1$s" of "%2$s" should be made virtual.

Explanation

A virtual destructor in the base class ensures that the proper destructor is called.

In the message text:

"%1$s" is the base class to change. "%2$s" is the derived class.

User response

Declare the destructor with the virtual keyword.

CCN7614	A user-defined copy constructor/ assignment operator should be created in "%1$s" to handle a pointer data member.

Explanation

The compiler generated copy constructor and assignment operator does a bitwise member copy.

In the message text:

"%1$s" is the class that has a pointer to data member.

User response

Create a copy constructor and an assignment operator.

CCN7616	"%1$s" does not assign values to all data members in the class.

Explanation

Checks that all data members in a class are assigned to when user defined assignment operators are present.

In the message text:

"%1$s" is the offending class.

User response

Assign value to data member.

CCN7617	"%1$s" was not initialized.

Explanation

The data member was not initialized.

In the message text:

"%1$s" is a data member.

User response

Initialize the member.

CCN7618	"%1$s" should be initialized using the member initialization list.

Explanation

Initializing a data member is faster than assignment in the constructor.

In the message text:

"%1$s" is the data member to initialize.

User response

Initialize the data member in the constructor list.

CCN7619	"%1$s" should be initialized in the same order as it is declared in "%2$s". It should be initialized after "%3$s".

Explanation

Data members are initialized in the order they are declared. The initialization list should reflect this order.

In the message text:

"%2$s" is the class name. "%1$s" and "%3$s" are its data members. "%3$s" is a data member that is after "%1$s" in the class definition.

User response

Re-order the initialization list to be the same as the declaration order.

CCN7620	"%1$s" is a non-const namespace variable and may cause problems in multithreaded code.

Explanation

Variables in namespace scope that are not protected by a mutex may behave unexpectedly in multithreaded code.

In the message text:

"%1$s" is a variable in namescope scope.

User response

Don't use variables in namespace scope for multithreaded code.

CCN7621	"%1$s" is a global variable and may cause problems in multithreaded code.

Explanation

Global variables that are not protected by a mutex may behave unexpectedly in multithreaded code.

In the message text:

"%1$s" is a global variable.

User response

Don't use global variables for multithreaded code.

CCN7622	"%1$s" is a static local variable and may cause problems in multithreaded code.

Explanation

Static local variables that are not protected by a mutex may behave unexpectedly in multithreaded code.

In the message text:

"%1$s" is a static local variable.

User response

Don't use static local variables for multithreaded code.

CCN7623	"%1$s" is a static member variable and may cause problems in multithreaded code.

Explanation

Static member variables that are not protected by a mutex may behave unexpectedly in multithreaded code.

In the message text:

"%1$s" is a static member variable.

User response

Don't use static member variables for multithreaded code.

CCN7624	64-bit portability : possible truncation of pointer through conversion of pointer type into int type.

Explanation

Conversion from an 8 byte pointer type into a 4 byte int type could be incorrect.

User response

Change the int type to long.

CCN7625	64-bit portability : possible truncation of array through conversion of array type into int type.

Explanation

Conversion from an 8 byte array type into a 4 byte int type could be incorrect.

User response

Change the int type to long.

CCN7626	64-bit portability : possible truncation of function through conversion of function type into int type.

Explanation

Conversion from an 8 byte function type into a 4 byte int type could be incorrect.

User response

Change the int type to long.

CCN7627	64-bit portability : possible incorrect pointer through conversion of integral type into pointer.

Explanation

Casting an integral type smaller than 8 bytes to a 64-bit pointer will set the upper bytes to all zeros, or all ones; likely an invalid pointer.

User response

Explicitly cast to larger integral type before casting to pointer.

CCN7628	64-bit portability : possible loss of digits through conversion of long type into int type.

Explanation

Conversion from an 8 byte long type into a 4 byte int type could be incorrect.

User response

Change the int type to long.

CCN7629	64-bit portability : possible change of result through conversion of unsigned int type into long int type. In 32-bit mode, values greater than INT_MAX would be truncated, but not in 64-bit mode.

Explanation

In 32-bit mode, values greater than INT_MAX would be truncated and could be incorrect.

User response

Make sure that values <= INT_MAX.

CCN7630	64-bit portability : possible difference in results. In 32-bit mode, values < INT_MIN or > INT_MAX would be truncated, but not in 64-bit mode.

Explanation

In 32-bit mode, values < INT_MIN or > INT_MAX could be incorrect.

User response

Make sure that values <= INT_MAX and values >= INT_MIN.

CCN7631	64-bit portability : possible difference in results. Values < 0 would give different results in 64-bit mode, values > UINT_MAX would be truncated in 32-bit mode but not in 64-bit mode.

Explanation

Possible difference in results if value < 0.

User response

Make sure that values >= 0.

CCN7632	64-bit portability : possible difference in results. Values > INT_MAX would be truncated in

32-bit mode but not in 64-bit mode.

Explanation

Values > INT_MAX could be incorrect in 32-bit mode.

User response

Make sure that values <= INT_MAX.

CCN7633	64-bit portability : possible difference in results. Values > UINT_MAX would be truncated in 32-bit mode but not in 64-bit mode.

Explanation

Values > UINT_MAX could be incorrect in 32-bit mode.

User response

Make sure that values <= UINT_MAX.

CCN7634	64-bit portability : possible difference in results if value is negative.

Explanation

Values < 0 would give different results in 64-bit mode.

User response

Make sure that values >= 0.

CCN7635	"%1$s" is not used in function "%2$s".

Explanation

The variable "%1$s" is not used in function "%2$s".

In the message text:

"%1$s" is a variable name. "%2$s" is a function.

User response

Either use the variable or remove it appropriately.

CCN7636	Global variable "%1$s" is not used.

Explanation

A global variable was declared but not used.

In the message text:

"%1$s" is a global variable.

User response

Remove the variable.

CCN7637 **Null statement.**

Explanation

This C++ statement has no effect.

User response

Remove the statement.

CCN7638 **The condition evaluates to a constant value.**

Explanation

An expression in a condition will not change during execution.

User response

Remove the condition.

CCN7639 **Precision will be lost in assignment to bit field "%1$s".**

Explanation

The size of the value assigned to the bit field is too large.

In the message text:

"%1$s" is the name of the bit field.

User response

Increase the size of the bit field or reduce the value assigned.

CCN7640 **The statement is unreachable.**

Explanation

Statements that are unreachable are never executed.

User response

Remove unreachable statements.

CCN7641 **Auto compiler temporary of type "%1$s" has been generated.**

Explanation

A temporary variable was generated by the compiler to hold an intermediate result.

In the message text:

"%1$s" is the type of the temporary variable.

User response

Modify expression to remove the need for the compiler generated temporary.

CCN7642 **The constant expression is larger than the size of the bit field type.**

Explanation

This may result in unexpected behavior.

User response

Choose a different bit field type or reduce the size of the bit field.

CCN7643 **The function %1$s declared with attribute "noreturn" or pragma leaves may return.**

Explanation

The noreturn function should have reachable call to noreturn function.

In the message text:

"%1$s" is the function name.

User response

Make sure the noreturn function has reachable call to noreturn function.

CCN7644 **Pointer type "%1$s" and type "%2$s" are not compatible in the current aliasing mode.**

Explanation

This may break ANSI aliasing rules.

In the message text:

"%1$s" is a type. "%2$s" is a type.

User response

Make sure that there is no need to do this cast in the code. Or use different aliasing mode to ensure optimization correctness.

CCN7645 **Array "%1$s", was not initialized in its declaration.**

Explanation

The array is not initialized when it is declared.

In the message text:

"%1$s" is an array.

User response

Make sure that the array is initialized when it is declared.

CCN7646   **Label "%1$s" defined but not used.**

Explanation

The label is defined but it is never referenced.

In the message text:

"%1$s" is a label.

User response

Make sure that the label is referenced. Or the label could be incorrect.

CCN7647   **The "vector" keyword must be the first type specifier used.**

Explanation

The "vector" keyword must be the first type specifier used.

User response

Element specifier must come after vector specifier.

CCN7648   **Deprecated type specifier "long" in AltiVec type, use "int" instead.**

Explanation

Deprecated type specifier "long" in AltiVec type, use "int" instead.

User response

Long to be deprecated in future release of Altivec PIM.

CCN7649   **The condition always evaluates to true.**

Explanation

An expression in a condition will not change during execution.

User response

Remove the condition.

CCN7650   **The condition always evaluates to false.**

Explanation

An expression in a condition will not change during execution.

User response

Remove the condition.

CCN7651   **64-bit portability : possible compilation error due to explicit conversion of pointer type to integral type.**

Explanation

Conversion from an 8-byte pointer type to an integral type smaller than 8 bytes is not allowed

User response

Change the target type of the conversion to a larger integral type, such as long or long long.

CCN7652   **The right-hand side of a bitwise shift expression should be positive and less than the width in bits of the promoted left operand.**

Explanation

This expression may not be portable.

User response

Change the shift expression.

CCN8100   **"%1$s" specified in "%2$s" is not a valid numeric value. The option is ignored.**

Explanation

The specified option was ignored because the argument was not a valid numeric value.

In the message text:

"%1$s" is the invalid numeric value. "%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8101   **The numeric value "%1$s" specified in "%2$s" is out of bounds. The option is ignored.**

Explanation

The specified option was ignored because the argument was not a numeric value within the range specified by this option.

In the message text:

"%1$s" is the out-of-bounds value specified. "%2$s" is the option being ignored.

User response

Verify the allowable values for this option.

CCN8102   **The alignment value "%1$s" specified in "%2$s" is not a power of two. The option is ignored.**

Explanation

The specified option was ignored because the alignment specified was not a power of two.

In the message text:

"%1$s" is the invalid alignment value. "%2$s" is the option being ignored.

User response

Verify the allowable values for this option.

CCN8103   **"%1$s" specified in "%2$s" is not recognized. The option is ignored.**

Explanation

The specified option was ignored because the specified argument was not recognized.

In the message text:

"%1$s" is the unrecognized argument. "%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8104   **The message number %1$s specified in "%2$s" is not a valid message ID. The option is ignored.**

Explanation

The specified option was ignored because the message ID is not valid.

In the message text:

"%1$s" is the invalid message ID. "%2$s" is the option being ignored.

User response

Verify the syntax of the option and the message ID.

CCN8105   **A non-empty string is required but "%1$s" appears in "%2$s". The option is ignored.**

Explanation

The specified option was ignored because it was expecting a string with characters in it.

In the message text:

"%1$s" is the invalid argument. "%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8106   **An option argument is required but is not found in "%2$s". The option is ignored.%1$s**

Explanation

The specified option was ignored because it expected an argument which was not provided.

In the message text:

"%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8107   **"%1$s" specified in "%2$s" contains embedded spaces. The option is ignored.**

Explanation

The specified option was ignored due to embedded spaces in the argument.

In the message text:

"%1$s" is the argument containing embedded spaces. "%2$s" is the option being ignored.

User response

Verify the syntax of the option and the value passed as an argument.

CCN8108   **The option argument "%1$s" specified in "%2$s" is not valid. The option is ignored.**

Explanation

The specified option was ignored because the argument specified was not valid.

In the message text:

"%1$s" is the invalid argument. "%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8109   **The section attributes "%1$s" specified in "%2$s" are not valid. The option is ignored.**

Explanation

The specified option was ignored because the section attributes argument was not valid.

In the message text:

"%1$s" is the invalid section attributes argument. "%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8110   **An unnecessary argument "%1$s" is found in "%2$s". The option is ignored.**

Explanation

The specified option was ignored because an unnecessary argument was specified.

In the message text:

"%1$s" is the unnecessary argument. "%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8111   **"%1$s" specified in "%2$s" requires an additional option argument. The option is ignored.**

Explanation

The specified option was ignored because the argument requires more information.

In the message text:

"%1$s" is the argument that requires more information. "%2$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8120   **The AlignAddr value "%1$s" is less than the AlignFile value "%2$s".**

Explanation

The AlignAddr value must be greater than the AlignFile value.

In the message text:

"%1$s" is the AlignAddr value. "%2$s" is the AlignFile value.

User response

Change the values.

CCN8121   **"%1$s" in "%2$s" is not a valid object model name. The option is ignored.**

Explanation

The option specified was ignored because the specified object model name was not valid.

In the message text:

"%1$s" is the invalid object model name specified. "%2$s" is the option being ignored.

User response

Verify the option syntax.

CCN8122   **"%1$s" is in conflict with "%2$s". The option is ignored.**

Explanation

The options specified are not valid if they are specified together.

In the message text:

"%1$s" and "%2$s" are the conflicting options.

User response

Verify the options and remove or modify one of them.

CCN8123   **The string "%1$s" in "%2$s" is not a valid identifier. The option is ignored.**

Explanation

The specified option was ignored because it expected a valid identifier.

In the message text:

"%1$s" is the invalid identifier specified. "%2$s" is the option being ignored.

User response

Verify the syntax of the option and the string specified.

CCN8124   **The string "%1$s" in "%2$s" is not a valid keyword. The option is ignored.**

Explanation

The specified option was ignored because it expected a string containing a valid keyword.

In the message text:

"%1$s" is the invalid string specified. "%2$s" is the option being ignored.

User response

Verify the syntax of the option and the string specified.

CCN8125   **The option argument "%1$s" specified in "%2$s" is longer than %3$s characters. The option is ignored.**

Explanation

The specified option was ignored because the argument was too long.

In the message text:

"%1$s" is the invalid argument. "%2$s" is the option being ignored. "%3$s" is the maximum length.

User response

Verify the syntax and constraints of the option.

CCN8126   **The option "%1$s" requires architecture level "%2$s" or above. The option "%1$s" is ignored.**

Explanation

The specified option is ignored because it needs the required architecture level.

In the message text:

"%1$s" is the ignored option and "%2$s" is the required architecture level.

User response

Remove the option or specify the required architecture level.

CCN8127   **Option "%2$s" must be specified for option "%1$s".**

Explanation

The specified option does not take effect because a required option is not specified.

In the message text:

"%1$s" is the specified option, "%2$s" is the option required by "%1$s".

User response

Remove the option or specify its required option.

CCN8130   **The value "%1$s" in option "%2$s" is reserved for system use. The value is not accepted.**

Explanation

The specified value is not accepted because it is reserved by the system.

In the message text:

"%1$s" is the value. "%2$s" is the option.

User response

Change the specified value.

CCN8131   **The global option directive "%1$s" must not be placed inside braces. The option is ignored.**

Explanation

The specified option directive is a global directive that applies to the target rather than to individual files.

In the message text:

"%1$s" is the option directive being ignored.

User response

Move the option to the global scope.

CCN8132	The global option directive "%1$s" is not allowed because it modifies a previous directive. The option is ignored.

Explanation

The specified option directive is ignored because it conflicts with a previous directive.

In the message text:

"%1$s" is the global option directive being ignored.

User response

Verify the meaning of the option directives specified to see that they do not conflict.

CCN8133	No include path is specified for the option "%1$s". The option is ignored.

Explanation

The specified option was ignored because it expected an include path as a an argument.

In the message text:

"%1$s" is the option being ignored.

User response

Verify the syntax of the option.

CCN8134	Error in setting option "%1$s" for extension source "%2$s". Configuration value "%3$s" has the wrong format.

Explanation

This is a warning message about compiler extension source options.

In the message text:

"%1$s" is the option. "%2$s" is the extension source. "%3$s" is the configuration value.

User response

If you are using that extension, use the correct option for that extension.

CCN8135	Default value of option "%1$s" in the .ice file has the wrong format "%2$s".

Explanation

The .ice file contains an invalid default value for the specified option.

In the message text:

"%1$s" is the option which has an invalid default value in the .ice file.

User response

Verify the syntax used to specify defaults in the .ice file.

CCN8136	Options "%1$s" and "%2$s" are in conflict.

Explanation

The specified options cannot be specified together because they conflict.

In the message text:

"%1$s" and "$2$s" are the conflicting options.

User response

Verify the option settings and remove or modify one of the conflicting options.

CCN8137	OBJECT_MODE setting "%1$s" is not recognized and is not a valid setting for the compiler.

Explanation

The specified OBJECT_MODE setting is not valid.

In the message text:

"%1$s" is the invalid setting.

User response

Verify the valid settings for OBJECT_MODE.

CCN8138	OBJECT_MODE = 32_64 is not a valid setting for the compiler.

Explanation

The 32_64 OBJECT_MODE setting is not supported.

User response

Verify the valid settings for OBJECT_MODE.

CCN8139	The global option "%1$s" should be applied to all sources and targets.

Explanation

A global option is an option that applies to all sources and targets rather than just one specified source file.

In the message text:

"%1$s" is the global option.

User response

Move the global option so that it applies to all targets and sources.

CCN8140	"%1$s" is not compatible with 64-bit object mode. The default value "%2$s" is being set.

Explanation

The specified option is not valid for 64-bit object mode, so the specified default is being set.

In the message text:

"%1$s" is the option that is not valid for 64-bit object mode. "%2$s" is the default value being set.

User response

Verify the options that are valid for 64-bit object mode or switch to 32-bit object mode.

CCN8141	"%1$s" is not compatible with 32-bit object mode. The default value "%2$s" is being set.

Explanation

The specified option is not valid for 32-bit object mode so the specified default is being set.

In the message text:

"%1$s" is the option which is not valid for 32-bit object mode. "%2$s" is the default value being set.

User response

Verify the options that are valid for 32-bit object mode or switch to 64-bit object mode.

CCN8142	"%1$s" is not compatible with "%2$s". "%3$s" is being set.

Explanation

The specified option values cannot be specified together because they are not compatible. A valid option is being set instead.

In the message text:

"%1$s" and "%2$s" are the incompatible option values. "%3$s" is the setting chosen by the compiler.

User response

Verify the option values, and either remove or modify them so that they are compatible.

CCN8143	"%1$s" option is specified, but no floating point traps are being detected.

Explanation

Floating point traps are enabled but no traps have been specified.

In the message text:

"%1$s" is the option.

User response

Remove the option.

CCN8144	The option "%1$s" requires "%2$s". The option is ignored.

Explanation

The specified option is ignored because it needs the required option.

In the message text:

"%1$s" is the ignored option, "%2$s" is the required option.

User response

Remove the option or specify the required option.

CCN8145	"main" cannot be exported. The directive is ignored.

Explanation

"main" is ignored because it cannot be exported.

User response

Remove "main".

CCN8146	Expected text "%1$s" was not encountered on option "%2$s". The option is ignored.

Explanation

option argument should have "%1$s".

In the message text:

"%1$s" is the unexpected text, "%2$s" is the ignored option.

User response

Verify the syntax of the option.

CCN8147	The compiler is operating in 32-bit mode. The option "%1$s" is ignored.

Explanation

The compiler should operate in 64-bit mode.

In the message text:

"%1$s" is the ignored option.

User response

To use the specified option, turn on 64 bit mode.

CCN8148	The current codeset "%1$s" is not utf-8. The option "%2$s" is ignored.

Explanation

The current locale should be utf-8.

In the message text:

"%1$s" is the current codeset. "%2$s" is the ignored option.

User response

Change the current locale in utf-8 to use this option.

CCN8150	The option "%1$s" requires one of the following "%2$s". The option is ignored.

Explanation

The specified option is ignored because it needs one of the other required options.

In the message text:

"%1$s" is the ignored option, "%2$s" is the list of required options.

User response

Remove the option or specify one of the required options.

CCN8151	The option "%1$s" sets "%2$s".

Explanation

The second option is set when the first option is specified.

In the message text:

"%1$s" is the explicitly set option, "%2$s" is the implicitly set option.

User response

If the implicitly set option is not desirable and the explicitly set option isn't required, remove the explicitly set option.

CCN8153	The correct way of representing the imaginary part of a complex number is by using "%1$s"

Explanation

The standard does not support using suffix i or j to represent the imaginary part of a complex number. Use "%1$s".

In the message text:

"%1$s" is the right format for representing the imaginary part of a complex number.

User response

Remove suffix i or j.

CCN8154	C++ complex types might be supported differently by this compiler than by other compilers. If you are compiling this program with more than one compiler, using complex types might result in program incompatibility.

Explanation

The complex data type is a nonstandard C++ extension.

User response

Do not use the predefined complex data type if portability is a key requirement. Use the library complex type instead.

CCN8155	The use of long in a vector type is not allowed in 64-bit mode.

Explanation

Long is not allowed in vector type in 64-bit mode.

User response

Use int instead of long for vector type.

CCN8157	Option "%1$s" may be in conflict with OpenMP.

Explanation

Option "%1$s" may cause behavior that is different than the OpenMP API specification.

In the message text:

"%1$s" is the explicitly-set option which causes the conflicting behavior.

User response

Remove the option.

CCN8158	The option "%1$s" only has an effect when preprocessed output is generated. The option is ignored.

Explanation

The specified option is ignored because it only has effect when preprocessed output is generated.

In the message text:

"%1$s" is the ignored option.

User response

Specify an option configuration that generates preprocessed output.

CCN8159	"%1$s" is deprecated when used together with "%2$s". The option is accepted but may not be in a future release.

Explanation

Combined use of the specified options is not recommended.

In the message text:

"%1$s" and "$2$s" are the option values, which are deprecated when used together.

User response

Verify the option values, and either remove or modify them to avoid this particular option combination.

CCN8160	The 'class' keyword is no longer required in this context for friend declarations under C++0x.

Explanation

C++0x introduced a change in syntax to class friend declarations, this new syntax is incompatible with the old C++98 friend syntax.

User response

Remove the 'class' keyword from the friend class declaration.

CCN8161	"%1$s" is deprecated. The option is accepted but may not be in a future release.

Explanation

Use of the specified option is not recommended.

In the message text:

"%1$s" is the deprecated option.

User response

Remove the option.

CCN8162	The "%1$s" option has been deprecated and might be removed in a future release. Consider using the "%2$s" environment variable instead.

Explanation

The option "%1$s" is not supported and should not be used. See the Compiler Reference for the "%2$s" environment variable.

In the message text:

"%1$s" is the option value and "%2$s" is the environment variable value.

User response

Change option "%1$s" to utilize the "%2$s" environment variable.

CCN8163	The "%1$s" option can prevent the "%2$s" option from detecting the use of some variables before they are set.

Explanation

The "%1$s" option can hide uninitialized variables from the "%2$s" option.

In the message text:

"%1$s" and "%2$s" are the option vlaues, "%1$s" can prevent "%2$s" from functioning.

User response

Verify the "%1$s" option does not prevent the "%2$s" option.

CCN8164	Vector long type is not allowed. Use vector int instead.

Explanation

Long is not allowed in vector type.

User response

Use int or long long instead of long for vector type.

CCN8165	The %1$s schedule type has been deprecated and might be removed in a future release. Consider using the %2$s schedule type. The schedule type is reset to %2$s.

Explanation

The schedule type %1$s is deprecated and would be reset to option %2$s. See the Compiler Reference for %2$s schedule type.

In the message text:

%1$s and %2$s are schedule type names.

User response

Change schedule type %1$s to utilize %2$s schedule type.

CCN8200	Class "%1$s" has base classes with different object models.

Explanation

The object model deals primarily with the layout of class hierarchies. All classes in the same inheritance hierarchy must have the same object model.

In the message text:

"%1$s" is the name of the derived class.

User response

Modify either the base class or the derived class so that both have the same object model.

CCN8201	Class "%1$s" is specified with a different object model than its base classes. The object model

specified in its base classes will be used.

Explanation

The object model deals primarily with the layout of class hierarchies. All classes in the same inheritance hierarchy must have the same object model.

In the message text:

"%1$s" is the name of the derived class.

User response

Modify either the base class or the derived class so that both have the same object model.

CCN8202	Class "%1$s" has different object model between its formal template class and its base classes.

Explanation

The object model deals primarily with the layout of class hierarchies. All classes in the same inheritance hierarchy must have the same object model. Any formal templates (primary templates or partial specializations) must also have the same object model.

In the message text:

"%1$s" is the name of the instance class.

User response

Modify either the base class or the formal template class so that both have the same object model.

CCN8204	The direct base "%1$s" inaccessible in "%2$s" due to ambiguity.

Explanation

A base class is inaccessible because it is ambiguous.

In the message text:

"%1$s" is the name of the base class, "%2$s" is the name of the derived class

User response

Remove the ambiguous base class from the class hierarchy.

CCN8205	The covariant return type is not supported on the specific platform, the function "%1$s"

has two covariant return types, "%2$s" and "%3$s".

Explanation

Covariant return type is not implemented on this platform.

In the message text:

"%1$s" is the function name, "%2$s" and "%3$s" are the two covariant return type names.

User response

Remove the covariant return type for the function.

CCN8400	**"%1$s" is undefined. The delete operator will not call a destructor.**

Explanation

The class is declared but not defined so a constructor will not be called when the object is deleted at this point.

In the message text:

"%1$s" is the class.

User response

Define the class.

CCN8401	**The address of the destructor "%1$s" cannot be taken.**

Explanation

An attempt has been made to take the address of a destructor.

In the message text:

"%1$s" is the destructor.

User response

Change the code to not take the address of the destructor.

CCN8402	**The explicit reference to the destructor "%1$s" can only be used in an explicit destructor call.**

Explanation

Destructors do not have names and can only be referred to in declarations and in pseudo-destructor calls.

In the message text:

"%1$s" is the destructor.

User response

Remove the reference to the destructor.

CCN8403	**An expression with type pointer-to-member function must be bound to an object or a pointer to an object when it is used with the function call operator ().**

Explanation

A pointer-to-member function must have an object to refer to when calling the function.

User response

Change the code so that the function is being called on an object or a pointer to an object.

CCN8404	**All the arguments must be specified for "%1$s" because its default arguments have not been checked yet.**

Explanation

The function is recursive and is using the default arguments. Because they have not been processed yet, they must be specified.

In the message text:

"%1$s" is the function.

User response

Specify the parameters to the function call.

CCN8405	**An empty initializer list cannot be used to initialize an unbounded array.**

Explanation

The array is unbounded and its size is not known so an empty initializer list cannot be used.

User response

Specify the size of the array or use a non-empty initializer list.

CCN8406	**Build with the "%1$s" compiler option to extend the scope of the for-init-statement declaration.**

Explanation

Informational message about the option for extending scope of the variable in the for statement.

In the message text:

"%1$s" is the compiler option that can extend the scope of the variables declared in the for statement.

User response

See the primary message.

CCN8407	The local macro "%1$s" is not visible in the current source.

Explanation

Informational message about a local macro.

In the message text:

"%1$s" is the macro.

User response

See the primary message.

CCN8408	The condition declaration cannot have type "%1$s".

Explanation

The type of the variable declared in the condition is not valid.

In the message text:

"%1$s" is the type.

User response

Change the type of the declaration in the condition to bool.

CCN8409	The condition declaration cannot be initialized with a brace list initializer.

Explanation

A declaration in a condition cannot be initialized with a brace list.

User response

Change the initializer so that it is not in brace list format.

CCN8410	The left side of the "%1$s" operator must be an lvalue.

Explanation

The operand on the left side is not an object that can be assigned a value.

In the message text:

"%1$s" is the operator.

User response

Change the left operand to an object that can be assigned a value.

CCN8411	A dynamic cast is present, but the correct RTTI option is not specified.

Explanation

The compilation unit must be compiled with RTTI enabled.

User response

Use the correct RTTI compiler option, or remove the dynamic cast.

CCN8412	A typeid is present, but the correct RTTI option is not specified.

Explanation

The compilation unit must be compiled with RTTI enabled.

User response

Use the correct RTTI compiler option, or remove the type ID.

CCN8413	The "__alignof__" operator cannot be applied to a bit field.

Explanation

An attempt to use the __alignof__ operator on a bit field has been made.

User response

Remove the use of the __alignof__ operator.

CCN8414	The identifier "__VA_ARGS__" is allowed only in the replacement list of a function-like macro that has an ellipsis, "...", in the parameter list.

Explanation

An attempt was made to use __VA_ARGS__ without an ellipsis in the macro's parameter list.

User response

Remove the use of __VA_ARGS__ or add an ellipsis.

CCN8415	This expression cannot be used as a typeof expression.

Explanation

The expression is inappropriate for use with the typeof extension.

User response

Change the expression.

CCN8418	The non-"%1$s" member function "%2$s" is called for "%3$s".

Explanation

Only the same cv-qualified member functions can be called with a more qualified or the same cv-qualified type of object.

In the message text:

"%1$s" is the cv-qualifier. "%2$s" is the function. "%3$s" is the object.

User response

Change the member function to be of the same cv-qualification or change the object to be non-cv-qualified.

CCN8419	A pointer to non-"%1$s" member function type "%2$s" is called for "%3$s".

Explanation

Only the same cv-qualified const member functions can be called with a more qualified or the same cv-qualified type of pointer-to-member.

In the message text:

"%1$s" is the cv-qualifier. "%2$s" is the function. "%3$s" is the type.

User response

Change the member function to be of the same cv-qualification or change the pointer-to-member to be non-cv-qualified.

CCN8421	A flexible array member must be the last member of an outer struct to be initialized.

Explanation

Initialization of a flexible array member is not allowed if the flexible array member is not the last member of the outer structure in a struct nest.

User response

Remove the initialization of the offending flexible array member.

CCN8422	A variable length array may not be initialized.

Explanation

An initialization list may not be used to initialize a variable length array.

User response

Remove the variable length array initialization list.

CCN8423	References to variable length arrays are not supported in C++.

Explanation

Variable length array references are not defined within the C99 language specification and are not suported in C++.

User response

Do not use variable length array reference types.

CCN8424	Variable length arrays may not be a data member of a structure, union, or class.

Explanation

Variable length arrays can only be automatic variables.

User response

Remove the data member declaration or change its type.

CCN8425	A variable length array of unknown size is not allowed in this context.

Explanation

An array bound '[*]' has been provided in an invalid context.

User response

Specify an expression for the variable length array bound.

CCN8426	The argument of the sizeof operator is (or contains) a zero extent array or flexible array member of size zero.

Explanation

This is an informational message describing the default size for a zero extent array and for a flexible array member.

User response

See the primary message.

CCN8427	A goto statement may not jump into the scope of a variable of variably modified type.

Explanation

The goto statement branches into the scope of a variable length array.

User response

Remove the goto statement or change the location of the label.

CCN8428	A variable length array of non-POD (plain old data) element type is not permitted.

Explanation

Support for variable length arrays in C++ is limited to C99 functionality.

User response

Use malloc or new to dynamically allocate an array.

CCN8429	The format of the designated initializer is incorrect.

Explanation

A designated initializer should contain a designator, followed by an expression to initialize it.

User response

Change the designated initializer syntax.

CCN8430	Casting to an array type is not permitted.

Explanation

A cast expression may not specify an array type.

User response

Remove the array type cast or correct the type.

CCN8431	A template may not be instantiated with a variably modified type.

Explanation

Template instantiation with a variably modified type is not permitted.

User response

Remove the template instantiation or correct the type.

CCN8432	"goto %1$s" bypasses the variable length array definition "%2$s".

Explanation

The goto statement skips over the definition of a variable length array.

In the message text:

"%1$s" is the label. "%2$s" is the missed variable length array.

User response

Move the label before the variable length array definition.

CCN8433	The "%1$s" statement bypasses the variable length array definition "%2$s".

Explanation

A case in the switch statement contains variable length arrays that are not contained within a compound statement.

In the message text:

"%1$s" is the case or default statement. "%2$s" is the bypassed variable length array.

User response

Add a pair of braces {} to enclose the code containing the variable length array.

CCN8434   The type is not allowed for the "%1$s" unary operation.

Explanation

Only integral type, floating point type or complex floating point type are allowed for the "%1$s" unary operation.

In the message text:

"%1$s" is the unary operation.

User response

Change the type to integral, floating point type or complex floating point type.

CCN8439   The constructor initializer is unexpected. This constructor delegates at line %1$s, column %2$s.

Explanation

A delegating constructor should have only one constructor initializer. The initialization can only be done within a non-delegating constructor. In other words, a delegating constructor cannot both delegate and initialize.

In the message text:

"%1$s" is the line number for the constructor initializer that specifies the target constructor. "%2$s" is the column number for the constructor initializer that specifies the target constructor.

User response

Remove the extra initializers from the constructor initializer list.

CCN8440   "%1$s" delegates to itself.

Explanation

A delegating constructor should not delegate to itself directly or indirectly. Attempts to do so lead to infinite recursion.

In the message text:

"%1$s" is a delegating constructor that delegates to itself directly or indirectly.

User response

Modify an affected constructor to be non-delegating or modify the chain of delegations to reach a non-delegating constructor.

CCN8441   "%1$s" delegates to "%2$s".

Explanation

Informational message indicating part of a delegation chain.

In the message text:

"%1$s" is a delegating constructor. "%2$s" is the constructor that is the direct target of "%1$s".

User response

See the primary message.

CCN8442   This expression cannot be used as a decltype expression.

Explanation

The expression is inappropriate for use with decltype.

User response

Change the expression.

CCN8443   The function "%1$s" given to decltype contains an ambiguous set of overloaded functions.

Explanation

Decltype must not be given an ambiguous set of overloaded functions as part of the argument.

In the message text:

"%1$s" is a function.

User response

Resolve the ambiguity or do not use decltype on the overloaded function set.

CCN8444   Auto deduced type cannot be incomplete type.

Explanation

An incomplete type was specified in the initializer of auto deduced type and it is not allowed.

User response

Specify a complete type in the initializer of auto deduced type.

CCN8445 **An rvalue has been bound to an lvalue reference to a non-const or volatile type.**

Explanation

The C++ language does not permit an rvalue to be bound to a non-const or volatile lvalue reference; this binding is not portable.

User response

Change the reference to be a non-volatile const lvalue reference.

CCN8446 **The generated temporary will not be treated as a local variable due to the presence of label definition "%1$s".**

Explanation

A branch to the label may bypass construction of the temporary. The temporary is destroyed in the standard compliant way.

In the message text:

"%1$s" is the name of the problematic label definition.

User response

Remove the label definition or enclose the temporary construction in an appropriate lexical block.

CCN8447 **The generated temporary will not be treated as a local variable due to the presence of computed goto on line "%1$s", column "%2$s".**

Explanation

The computed goto may bypass the destruction of the temporary. The temporary is destroyed in the standard compliant way.

In the message text:

"%1$s" is the line of the computed goto in the user source. "%2$s" is the column of the computed goto in the user source.

User response

Remove the computed goto or enclose the temporary construction in an appropriate lexical block.

CCN8460 **The parameter of a catch block cannot be an rvalue reference.**

Explanation

The C++ language does not permit an rvalue reference type in an exception-declaration.

User response

Change the type of the parameter for the catch block.

CCN8461 **The rvalue reference type, "%1$s", is not allowed in an exception-specification.**

Explanation

The C++ language does not permit an rvalue reference type in an exception-specification.

In the message text:

"%1$s" is the rvalue reference type.

User response

Correct the exception-specification type list.

CCN8462 **A prvalue expression of type "%1$s" cannot be converted to type "%2$s".**

Explanation

Conversion from a prvalue expression of type "%1$s" to type "%2$s" cannot be done in this context.

In the message text:

"%1$s" is the original type. "%2$s" is the target type.

User response

Change the types or provide conversion functions.

CCN8463 **An lvalue expression of type "%1$s" cannot be converted to type "%2$s".**

Explanation

Conversion from an lvalue expression of type "%1$s" to type "%2$s" cannot be done in this context.

In the message text:

"%1$s" is the original type. "%2$s" is the target type.

User response

Change the types or provide conversion functions.

CCN8464   **An xvalue expression of type "%1$s" cannot be converted to type "%2$s".**

Explanation

Conversion from an xvalue expression of type "%1$s" to type "%2$s" cannot be done in this context.

In the message text:

"%1$s" is the original type. "%2$s" is the target type.

User response

Change the types or provide conversion functions.

CCN8465   **An object or reference of type "%2$s" cannot be initialized with a prvalue of type "%1$s".**

Explanation

This object or reference must be initialized with a glvalue or a prvalue of a type other than "%1$s".

In the message text:

"%2$s" is the type of the object. "%1$s" is the type of the prvalue.

User response

Change the type of the object or reference.

CCN8466   **A parameter of type "%2$s" cannot be initialized with a prvalue of type "%1$s".**

Explanation

This parameter must be initialized with a glvalue or a prvalue of a type other than "%1$s".

In the message text:

"%2$s" is the type of the parameter. "%1$s" is the type of the prvalue.

User response

Change the type of the parameter.

CCN8467   **A return value of type "%2$s" cannot be initialized with a prvalue of type "%1$s".**

Explanation

The return value must be initialized with a glvalue or a prvalue of a type other than "%1$s".

In the message text:

"%2$s" is the return type. "%1$s" is the type of the prvalue.

User response

Change the return type.

CCN8468   **An object or reference of type "%2$s" cannot be initialized with an lvalue of type "%1$s".**

Explanation

This object or reference must be initialized with an rvalue or an lvalue of a type other than "%1$s".

In the message text:

"%2$s" is the type of the object. "%1$s" is the type of the lvalue.

User response

Change the type of the object or reference.

CCN8469   **A parameter of type "%2$s" cannot be initialized with an lvalue of type "%1$s".**

Explanation

This parameter must be initialized with an rvalue or an lvalue of a type other than "%1$s".

In the message text:

"%2$s" is the type of the parameter. "%1$s" is the type of the lvalue.

User response

Change the type of the parameter.

CCN8470   **A return value of type "%2$s" cannot be initialized with an lvalue of type "%1$s".**

Explanation

The return value must be initialized with an rvalue or an lvalue of a type other than "%1$s".

In the message text:

"%2$s" is the return type. "%1$s" is the type of the lvalue.

User response

Change the return type.

CCN8471 **An object or reference of type "%2$s" cannot be initialized with an xvalue of type "%1$s".**

Explanation

This object or reference must be initialized with an expression that is not an xvalue of type "%1$s".

In the message text:

"%2$s" is the type of the object. "%1$s" is the type of the xvalue.

User response

Change the type of the object or reference.

CCN8472 **A parameter of type "%2$s" cannot be initialized with an xvalue of type "%1$s".**

Explanation

This parameter must be initialized with an expression that is not an xvalue of type "%1$s".

In the message text:

"%2$s" is the type of the parameter. "%1$s" is the type of the xvalue.

User response

Change the type of the parameter.

CCN8473 **A return value of type "%2$s" cannot be initialized with an xvalue of type "%1$s".**

Explanation

The return value must be initialized with an expression that is not an xvalue of type "%1$s".

In the message text:

"%2$s" is the return type. "%1$s" is the type of the xvalue.

User response

Change the return type.

CCN8474 **The scoped enumeration variable "%1$s" cannot be converted to "%2$s".**

Explanation

Scoped enums cannot be implicitly converted to any other type.

In the message text:

"%1$s" is the scoped enum variable that is being implicitly converted to type "%2$s".

User response

Use a scoped enum only in the context of a scoped enum of the same type to avoid the need for a conversion.

CCN8475 **The case expression "%1$s" should be of type "%2$s".**

Explanation

The case expression "%1$s" has a type different from "%2$s", which is the type of the switch condition.

In the message text:

"%1$s" is the case statement, "%2$s" is the scoped enum type of the condition in the switch statement.

User response

Scoped enums of matching types can only be used throughout a switch/case statement.

CCN8600 **"%1$s" operator cannot be overloaded.**

Explanation

The attempted operator overload is not valid.

In the message text:

"%1$s" is the operator.

User response

Change the declaration to overload a different operator.

CCN8601 **Forward declaration of the enumeration "%1$s" is not allowed.**

Explanation

Enumerations cannot have forward declarations.

In the message text:

"%1$s" is the enumeration.

User response

Define the enumeration before attempting to use an elaboration of the enumeration.

CCN8602   **The first non-matching token was encountered on line %1$s, column %2$s. A project cannot contain more than one definition of a class unless each definition consists of the same sequence of tokens.**

Explanation

Informational message indicating the first token that differs in the two class definitions.

In the message text:

"%1$s" is the line number. "%2$s" is the column number.

User response

See the primary message.

CCN8603   **The parameter must not be specified with this scheduling type.**

Explanation

This schedule clause kind does not allow a chunk_size parameter.

User response

Remove the chunk_size expression from the schedule clause.

CCN8606   **"restrict" can only qualify a pointer or reference type. The "restrict" keyword is ignored.**

Explanation

The "restrict" qualifier is only allowed to adorn a pointer or a reference type.

User response

Apply the "restrict" keyword to a pointer or reference type.

CCN8607   **The "__callback" keyword can only adorn a pointer to a function. The keyword is ignored.**

Explanation

The "__callback" keyword is only allowed to adorn a pointer to a function.

User response

Remove the __callback or apply the "__callback" to a pointer to a function.

CCN8608   **The "__ptr32" qualifier cannot be applied to a pointer that is in the return type of a function or in a parameter to a function.**

Explanation

The "__ptr32" qualifier is not allowed on a pointer that is part of a function type. That is, a pointer that is part of a function return type or part of a function parameter type.

User response

Remove the __ptr32 qualifier.

CCN8609   **The linkage keyword "%1$s" is deprecated and has no meaning. The keyword is ignored.**

Explanation

The linkage keyword has no meaning and is ignored.

In the message text:

"%1$s" is the deprecated linkage keyword.

User response

Remove the linkage keyword.

CCN8610   **The pascal string is too long. It will be truncated to 255 bytes in length.**

Explanation

The pascal string can be a maximum of 255 bytes in length.

User response

Shorten the pascal string.

CCN8611   **The name "%1$s" can only be used to declare a constructor.**

Explanation

The constructor for a class cannot be used as a type specifier.

In the message text:

"%1$s" is the constructor.

User response

Declare the constructor or specify a valid type.

CCN8612   **The hexadecimal floating point constant "%1$s" cannot be represented exactly in its evaluated format.**

Explanation

Due to limits on the number of significant digits, the hexadecimal floating point constant is rounded.

In the message text:

"%1$s" is the hexadecimal floating constant.

User response

Change the hexadecimal floating point constant so that it fits in the evaluation format.

CCN8613   **A variable length array type cannot be used in a compound literal expression.**

Explanation

A compound literal expression can only have an array type of unknown size or fixed size.

User response

Remove the variable array type from the compound literal.

CCN8614   **The static keyword or type qualifiers are ignored unless they are in the outermost array index of a function parameter.**

Explanation

The array index contains the static keyword or type qualifiers. When the static keyword or type qualifiers are used to specify the dimension of an array, they can only be used for the declaration of function parameters and only in the outermost array dimension.

User response

Remove the static keyword or type qualifiers.

CCN8615   **The attribute "%1$s" cannot be applied to this variable. The attribute is ignored.**

Explanation

The attribute is generally not supported for the type of variable.

In the message text:

"%1$s" is the attribute specified.

User response

Remove the section attribute specifier.

CCN8616   **A different section was specified for "%1$s"; the new specification is used.**

Explanation

The new section specification overrides the previous one.

In the message text:

"%1$s" is an identifier.

User response

Remove the previous specification of attribute "section".

CCN8617   **The attribute "section" has been specified more than once; the last specification is used.**

Explanation

The identified attribute was specified more than once; the last specification is used.

User response

Remove the duplicate attribute specifier.

CCN8618   **The class template name "%1$s" did not match an injected class name and must be followed by a template parameter list.**

Explanation

The template must have its template parameter list specified.

In the message text:

"%1$s" is the name of the template class.

User response

Add the < and the appropriate template parameter list followed by >.

CCN8619   **The anonymous enumeration declaration does not declare a name.**

Explanation

An anonymous enumeration has been specified without an enumerator list.

User response

Either name the indicated enumeration or specify its enumerators.

CCN8621   **The type attribute "%1$s" is ignored because it is not supported for this type.**

Explanation

The identified attribute is attached to the type of the declarator, but it is not supported for this type.

In the message text:

"%1$s" is the invalid attribute.

User response

Remove the attribute specifier or, if you wish to specify it on a variable, attach it to the variable by placing it after the variable declarator.

CCN8622   **The expression must be an integral non-volatile expression.**

Explanation

Only an integral expression can be used in this context, but a non-integral expression is specified.

User response

Change the expression to be an integral expression.

CCN8623   **A character string literal cannot be concatenated with a wide string literal.**

Explanation

You can only concatenate character string literals or wide string literals, but not both together.

User response

Change the string concatenation.

CCN8625   **The asm label specification cannot be applied to a constructor or**

destructor declaration. The asm specification is ignored.

Explanation

An asm label cannot be specified in this type of declaration.

User response

Remove the asm label specifier.

CCN8626   **The asm label specification cannot be applied to a local object or static object. The asm specification is ignored.**

Explanation

An asm label cannot be specified in this type of declaration.

User response

Remove the asm label specifier.

CCN8627   **The asm label specification cannot be applied to a non-static data member. The asm specification is ignored.**

Explanation

An asm label cannot be specified in this type of declaration.

User response

Remove the asm label specifier.

CCN8628   **The asm label specification cannot be applied to a template function declaration or any declaration contained within a template. The asm specification is ignored.**

Explanation

An asm label cannot be specified in this type of declaration.

User response

Remove the asm label specifier.

CCN8629   **The asm label specification cannot be applied to a typedef declaration. The asm specification is ignored.**

Explanation

An asm label cannot be specified in this type of declaration.

User response

Remove the asm label specifier.

CCN8630	The asm label specification cannot be applied to a virtual member function declaration. The asm specification is ignored.

Explanation

An asm label cannot be specified in this type of declaration.

User response

Remove the asm label specifier.

CCN8631	The asm label specification cannot be applied to a friend declaration. The asm specification is ignored.

Explanation

An asm label cannot be specified in this type of declaration.

User response

Remove the asm label specifier.

CCN8633	Decimal floating-point constant "%1$s" is out of range.

Explanation

The compiler detected a decimal floating-point overflow or underflow while scanning a constant or performing constant arithmetic folding.

In the message text:

"%1$s" is a decimal floating-point constant.

User response

Change the decimal floating-point constant so that it fits in the evaluation format or use a format with higher precision. See the XL C/C++ Language Reference for details on the valid format.

CCN8634	Decimal floating-point constant "%1$s" cannot be represented exactly in its evaluated format.

Explanation

Due to limits on the number of significant digits, the decimal floating-point constant is rounded.

In the message text:

"%1$s" is a decimal floating-point constant.

User response

Change the decimal floating-point constant so that it fits in the evaluation format or use a format with higher precision. See the XL C/C++ Language Reference for details on the valid format.

CCN8638	The __thread keyword can only be specified for external or static data.

Explanation

The __thread keyword is used in an invalid context.

User response

Remove the __thread attribute.

CCN8639	The attribute "%1$s" is specified for "%2$s"; the attribute "%3$s" is ignored.

Explanation

The attribute "%1$s" has the higher priority than the attribute "%3$s".

In the message text:

"%1$s" and "%3$s" are attribute names. "%2$s" is an identifier.

User response

Use only attribute %1$s" or attribute "%3$s" for the variable.

CCN8640	A different attribute common/nocommon was specified; the new specification is used.

Explanation

The new attribute common/nocommon specification overwrites the previous one.

User response

Remove the previous specification of attribute common/nocommon.

CCN8641   **The type "%1$s" is not supported on the target architecture.**

Explanation

The target architecture does not natively support machine instructions for the specified type.

In the message text:

"%1$s" is a type.

User response

Remove the reference to the type.

CCN8643   **No static initialization may refer to the address of a thread-local variable.**

Explanation

The __thread keyword is used in an invalid context.

User response

Remove the __thread attribute.

CCN8644   **The "[" has no matching "]".**

Explanation

There is an imbalance of left and right brackets.

User response

Ensure that each left bracket has a matching right bracket.

CCN8645   **Attribute "%1$s" has been specified more than once; the last specification is used.**

Explanation

The new attribute specification overwrites the previous one.

In the message text:

"%1$s" is an attribute name.

User response

Remove the previous specification of attribute "%1$s".

CCN8646   **"auto" is a storage class in this context and cannot be specified with any other storage class.**

Explanation

"auto" is a storage class in this context and it conflict with another storage class.

User response

Remove extra storage class specifers so there is just one specifier.

CCN8647   **Expecting single token "auto" for trailing return type placeholder.**

Explanation

Auto type specifier placeholder can be single token only in conjuction with trailing return type.

User response

Please specify "auto" for trailing return type placeholder.

CCN8648   **"auto" type specifier can only be specified for static members with constant initializers.**

Explanation

"auto" is a type specifer in this context and it is not supported for non-static class members.

User response

Remove the "auto" type specifier or make the member static.

CCN8649   **"auto" is a type specifier in this context. "%1$s" must have an initializer for automatic type deduction.**

Explanation

"auto" is a type specifer in this context, an initializer must be supplied to deduce the type.

In the message text:

"%1$s" is the variable name.

User response

Specify an initializer for "%1$s".

CCN8650   **The second operand to a static_assert expression must be a string literal.**

Explanation

The second operand to a static_assert expression represents the diagnostic message to be displayed upon assertion failure.

User response

Replace the second operand to the static_assert expression with a string literal.

CCN8651	The argument to decltype must be an expression.

Explanation

The argument given to the decltype operator is not an expression.

User response

Pass an expression to decltype.

CCN8652	Decltype is being used but decltype is not enabled. To enable decltype use -qlanglvl=decltype.

Explanation

Decltype has not been enabled but is present.

User response

Enable decltype or do not use the decltype operator.

CCN8653	Auto type is not supported in top-level array type.

Explanation

Auto type has been specified in top-level array type and is unsupported.

User response

Do not use auto type in top-level array types.

CCN8654	Auto type is not supported for brace list initialization.

Explanation

Auto type has been specified in brace list initialization and is unsupported.

User response

Do not use auto type in brace list initialization.

CCN8655	Auto type is not supported here.

Explanation

Auto type has been specified in an unsupported construct.

User response

Do not use auto type in this construct.

CCN8656	A non-function friend declaration must consist of a friend specifier followed by an elaborated-type-specifier, simple-type-specifier or typename-specifier.

Explanation

C++0x allows only specific syntactic forms for non-function friend declarations.

User response

Use one of the allowed forms. It may be necessary to use multiple friend declarations.

CCN8657	Variable declared with auto type cannot appear in its initializer.

Explanation

Variable declared with auto type was used in its initializer.

User response

Do not use variable declared with auto type in its initializer.

CCN8658	Auto type cannot be declared with more than one type token.

Explanation

Variable declared with auto type has been declared with additional type tokens.

User response

Remove additional type tokens.

CCN8659	Auto type specifier is not supported on functions without trailing return type.

Explanation

Trailing return type is missing for this function declarator.

User response

Please apply trailing-return-type in conjunction with auto type.

CCN8660   **The "explicit" specifier must be applied only to declarations of constructors or conversion operators within a class declaration.**

Explanation

The "explicit" specifier is being applied to something other than a constructor or conversion operator that is being declared in-line in the class.

User response

Remove the "explicit" specifier.

CCN8661   **The underlying type "%1$s" is not an integral type.**

Explanation

The underlying type of the enumeration should be an integral type.

In the message text:

"%1$s" is the underlying type used by the user.

User response

Please use an integral type as the underlying type.

CCN8662   **The scoped enum should have a name.**

Explanation

Scoped enums cannot be anonymous.

User response

Give the scoped enum a name or make it unscoped.

CCN8663   **The keyword "%1$s" is not allowed in an enum elaborated type specifier.**

Explanation

The elaborated type of scoped enums is the same as that of unscoped enums.

In the message text:

The keyword "%1$s" is not allowed in the elaborated type.

User response

Remove the extra keyword "%1$s".

CCN8664   **Enumerator "%1$s" has value "%2$s", which does not fit in the underlying type "%3$s".**

Explanation

Enumerator values should be representable by the enumeration's underlying type.

In the message text:

"%1$s" is the enumerator's name, "%2$s" is the enumerator value, and "%3$s" is the underlying type of the enumeration.

User response

Change the underlying type to fit the value, or change enumerator value if possible.

CCN8665   **Unexpected specifiers in enumeration's underlying type found.**

Explanation

The underlying type of an enumeration should not contain any specifiers.

User response

Remove any specifiers from the enumeration's underlying type.

CCN8666   **The "noreturn" specifier must be applied only to function declarations.**

Explanation

The "noreturn" specifier is being applied to something other than a function.

User response

Remove the "noreturn" specifier.

CCN8667   **"%1$s" cannot be declared as noreturn.**

Explanation

There are restrictions on "main" since it is the program starting point.

In the message text:

"%1$s" is the function name.

User response

Remove the noreturn specifier.

CCN8668	Forward declared unscoped enums should have a fixed underlying type.

Explanation

Unscoped enumerations with no explicit underlying type cannot be forward declared.

User response

Add an underlying type to the forward declaration of the unscoped enum.

CCN8669	"%1$s" is a scoped enumerator and cannot be named by a using-declaration.

Explanation

using-declarations cannot name scoped enumerators.

In the message text:

"%1$s" is the enumerator that is being named by the using-declaration.

User response

Make the enumerator unscoped, or remove the whole using-declaration.

CCN8670	"%1$s" is invalid. Qualified enumeration names cannot be forward declared.

Explanation

Forward declaration of enums cannot use nested-name-specifiers in the enum name.

In the message text:

"%1$s" is the qualified name used in the forward declaration.

User response

Remove the nested-name-specifier from "%1$s".

CCN8671	Unable to deduce the type of variable "%1$s" from expression "%2$s".

Explanation

The compiler was unable to deduce the type of the auto variable from the initializer expression provided.

In the message text:

"%1$s" is the name of the variable, "%2$s" is the initializer expression provided to deduce the type of the variable.

User response

Provide a correct initializer exprssion or specify the type of the variable.

CCN8672	Vector initializer list is non-portable.

Explanation

Vector initializer list is non-portable.

User response

Use parenthesis-style initialization instead.

CCN8701	The "pragma datamodel" stack is empty. The pragma is ignored.

Explanation

An attempt has been made to restore the previous pragma setting, but this is the first instance of the pragma.

User response

Remove the pragma.

CCN8702	Invalid syntax for pragma datamodel.

Explanation

The compiler has detected an invalid pragma datamodel syntax.

User response

Correct the syntax.

CCN8703	pragma datamodel(LLP64 | P128) seen without matching pragma datamodel(pop).

Explanation

At the end of compilation there was an extra pragma datamodel on the stack.

User response

Ensure that all pragma datamodel directives have a matching pragma datamodel(pop).

CCN8704   **The base class has a different data model than this derived class.**

Explanation

Base and derived classes must have identical data models.

User response

Change the data model of one of the classes.

CCN8705   **Cannot initialize a static __ptr64 with a __ptr128 value.**

Explanation

A __ptr64 variable is being initialized with a constant value when the storage model indicates such values are __ptr128s.

User response

Use a different initialization value or a different storage model.

CCN8706   **The declaration "%2$s" specified in "pragma %1$s" cannot be found.**

Explanation

Name lookup failed for the variable or type specified in the pragma.

In the message text:

"%1$s" is the name of the pragma. "%2$s" is the variable or type name.

User response

Change the pragma to refer to a declared variable or type or declare the indicated variable or type.

CCN8707   **The pragma map has been applied to function "%1$s", which has internal linkage. The pragma is ignored.**

Explanation

An internal linkage function cannot be mapped.

In the message text:

"%1$s" is the function name.

User response

Change the function's linkage or remove the pragma.

CCN8708   **The divisor for the modulus or division operator cannot be zero.**

Explanation

The result of the calculation is undefined.

User response

Change the value of the divisor or change the operator.

CCN8709   **The pragma "%1$s" directive must occur before the first C++ statement in the program. The directive is ignored.**

Explanation

The pragma must precede any C++ statement in the program.

In the message text:

"%1$s" is the pragma name.

User response

Move the pragma directive before any C++ statement.

CCN8710   **The pragma "%1$s" is ignored because the "%2$s" option is not specified.**

Explanation

The pragma must only be used when the option is specified.

In the message text:

"%1$s" is the pragma name, "%2$s" is the missing option.

User response

Remove the pragma or specify the option.

CCN8711   **Detected "%1$s" : "%2$s"**

Explanation

The pragma runopts has invalid arguments.

In the message text:

"%1$s" is the message number, "%2$s" is the message text.

User response

Correct the arguments.

CCN8712   **The pragma enum is not allowed in the middle of a declaration of an enumeration. This pragma is in effect after the enumeration declaration.**

Explanation

The pragma enum cannot appear inside a declaration of an enumeration.

User response

Place the pragma before or after the enum declaration.

CCN8713   **The pragma "%1$s" is ignored because the locale compiler option is not specified.**

Explanation

The locale compiler option is required for pragma "%1$s"

In the message text:

"%1$s" is the pragma name.

User response

Remove all the pragma &1 directives or specify the locale compiler option.

CCN8715   **The pragma runopts is not implemented with 64-bit mode.**

Explanation

The pragma runopts is not supported with 64-bit mode in the current release.

User response

Remove the pragma runopts if compiled in 64-bit mode.

CCN8716   **The "pragma wsizeof" stack is empty. The pragma is ignored.**

Explanation

The pragma is ignored because the wsizeof stack is empty.

User response

Remove the pragma or ensure that the stack is not empty.

CCN8717   **The "pragma %1$s" is not allowed in namespace scope. The pragma is ignored.**

Explanation

The pragma is ignored because it is specified in namespace scope.

In the message text:

"%1$s" is the pragma name.

User response

Use this pragma in a global scope.

CCN8718   **The unroll and nounroll pragmas must be applied to a for-loop or block_loop construct. The pragma is ignored.**

Explanation

The pragma is ignored because it is not applied to a for loop.

User response

Remove the offending pragma.

CCN8719   **Only one unroll directive may be specified on a single loop or block_loop. The pragma is ignored.**

Explanation

The pragma is ignored because it conflicts with another pragma specified on the same loop.

User response

Remove the conflicting pragma.

CCN8720   **The unroll pragma unrolling factor must be a positive scalar integer initialization expression. The pragma is ignored.**

Explanation

The pragma is ignored because its unrolling factor is not a positive scalar integer.

User response

Change the pragma factor to a positive scalar integer.

CCN8721   **The "pragma pass_by_value" stack is empty. The pragma is ignored.**

Explanation

The pragma is ignored because the pass_by_value stack is empty.

User response

Remove the pragma or ensure that the stack is not empty.

CCN8722	The declaration "%2$s" specified in pragma "%1$s" must be a variable. The pragma is ignored.

Explanation

The pragma is ignored because the declaration in the variable list is not a variable.

In the message text:

"%1$s" is the pragma name. "%2$s" is a declaration.

User response

Remove the declaration from the pragma declaration list.

CCN8723	The variable "%2$s" specified in pragma "%1$s" must be not be a member variable. The pragma is ignored.

Explanation

The pragma is ignored because the variable in the variable list is a member of a class.

In the message text:

"%1$s" is the pragma name. "%2$s" is a variable name.

User response

Remove the variable from the pragma declaration list.

CCN8724	The ccsid code page number "%1$s" specified in #pragma convert is not valid. The pragma is ignored.

Explanation

A valid ccsid suboption for #pragma convert is needed.

In the message text:

"%1$s" is a ccsid code page number.

User response

Specify a different ccsid number or remove the pragma.

CCN8725	Missing "%1$s" "%2$s" directive(s). The matching "%3$s" is/are no longer in effect.

Explanation

The "%3$s" is ignored because it needs a matching "%1$s".

In the message text:

"%1$s" is either convert or convlit. "%2$s" is either pop or resume. "%3$s" is either convert or convlit.

User response

Add the missing "%1$s" pragma or remove the "%3$s" pragma.

CCN8726	An empty pragma directive was found. The pragma directive is ignored.

Explanation

The pragma directive is ignored because there is no pragma specified in the directive.

User response

Remove the pragma directive or complete the pragma.

CCN8727	The "pragma nameManglingRule" stack is empty. The pragma is ignored.

Explanation

The pragma is ignored because the pragma nameManglingRule stack is empty.

User response

Remove the pragma or ensure that the pragma nameManglingRule stack is not empty.

CCN8728	The declaration "%2$s" specified in "pragma %1$s" has already been defined. The pragma is ignored.

Explanation

The identifier found in the pragma cannot be redefined.

In the message text:

"%1$s" is the name of the pragma and "%2$s" is the name of the identifier.

User response

Remove the definition of the identifier in question or remove the pragma.

CCN8735	The use of pragma once is deprecated. It may not be supported in future releases of the compiler or on other platforms.

Explanation

Ths pragma is provided for support of legacy code on some platforms. The continued use of this pragma is discouraged.

User response

Replace the use of this pragma with a C/C++ style header guard.

CCN8736	The loopid pragma can only be specified once per loop. The pragma is ignored.

Explanation

The pragma is ignored because this loop already has one loopid specified.

User response

Remove the offending pragma.

CCN8737	The loopid pragma may only be specified before a for loop or block_loop directive. The pragma is ignored.

Explanation

The pragma is ignored because the statement that follows it is not applicable to the loopid directive.

User response

Remove the offending pragma.

CCN8738	A loopid pragma must be unique within its enclosing scope. The pragma is ignored.

Explanation

The pragma is ignored because it is within the enclosing scope of another pragma loopid which has the same loopid name.

User response

Remove the offending pragma.

CCN8739	A block_loop pragma should precede a for loop or another block_loop directive. The pragma is ignored.

Explanation

The pragma is ignored because the statement that follows it is not applicable to the block_loop directive.

User response

Remove the offending pragma or move it to the correct place.

CCN8740	The block_loop directive is invalid because loopid, "%1$s", is not found within the loop nest. The pragma is ignored.

Explanation

The pragma is ignored because the loopid specified is not a valid loopid.

In the message text:

%1$s is the loopid name.

User response

Remove the offending pragma.

CCN8741	A block_loop directive may only be specified on a perfect loop nest. The pragma is ignored.

Explanation

The pragma is ignored because it is not applied to a perfect loop nest.

User response

Remove the offending pragma or fix the loop nest.

CCN8742	The loopid identifier name specified in this directive is not valid. The pragma is ignored.

Explanation

The pragma is ignored because the loopid identifier name is not in the proper format.

User response

Remove the offending pragma or fix the loopid identifier name.

CCN8743	The nosimd pragma can only be specified before a for, while, or do loop. The pragma is ignored.

Explanation

The pragma is ignored because it is not specified before a loop.

User response

Remove the offending pragma.

CCN8744	The novector pragma can only be specified before a for, while, or do loop. The pragma is ignored.

Explanation

The pragma is ignored because it is not specified before a loop.

User response

Remove the offending pragma.

CCN8745	The blocking factor specified for the block_loop directive must be a positive integral value. The pragma is ignored.

Explanation

The pragma is ignored because the blocking factor is not in the proper format.

User response

Remove the offending pragma or fix the blocking factor.

CCN8746	The unroll pragmas may not be applied to %1$s. The pragma is ignored.

Explanation

The pragma is ignored because it is not applied to a for loop.

In the message text:

%1$s is the statement to which the unroll pragma applies.

User response

Remove the offending pragma.

CCN8747	The Loopid directive has been referred to by more than one block_loop directive.

Explanation

A loop may only be blocked by one block_loop directive.

User response

Remove the offending block_loop directive, or block a different loop.

CCN8749	The conflicting mapping is specified on line %1$s of "%2$s".

Explanation

Informational message about the coordinates of the conflicting pragma map or asm label.

In the message text:

"%1$s" is the line number, "%2$s" is the file name.

User response

Correct the mapped names for consistency.

CCN8751	There is no matching #pragma extension for "pop" argument in the same file. The pragma will be ignored.

Explanation

An attempt has been made to apply #pragma extension (pop) without a matching pragma extension in the same file.

User response

Remove the pragma.

CCN8753	pragma "%1$s" must be specified at file scope. The pragma is ignored.

Explanation

The pragma directive is ignored because it has been specified at an invalid scope.

In the message text:

"%1$s" is the name of the pragma.

User response

Move the pragma directive to file scope.

CCN8754	The pragma "%1$s" directive must occur before the first CICS statement in the program. The directive is ignored.

Explanation

The pragma must precede any CICS statement or CICS keyword in the program.

In the message text:

"%1$s" is the name of the pragma.

User response

Move the pragma directive before any CICS statement or CICS keyword.

CCN8755	The "pragma report" stack is empty. The pragma is ignored.

Explanation

The pragma is ignored because the pragma report stack is empty.

User response

Remove the pragma or ensure that the pragma report stack is not empty.

CCN8756	An error was detected while writing to the temporary file "%1$s". The file system may be full.

Explanation

An error occurred writing to an intermediate code file. Please verify that the target file system exists, is writable, and is not full.

In the message text:

"%1$s" is a file name.

User response

Ensure that the designated location for temporary objects exists, is writable, and is not full.

CCN8757	Compiler cannot create the temporary file "%1$s". The file system may be full or not writable.

Explanation

The intermediate code files could not be created. Please verify that the target file system exists, is writable, and is not full.

In the message text:

"%1$s" is a file name.

User response

Ensure that the designated location for temporary objects exists, is writable, and is not full.

CCN8758	"pragma %1$s" is already specified for this option. The entire pragma clause is ignored.

Explanation

The entire pragma clause is ignored because one or more options it contains have already been specified.

In the message text:

"%1$s" is the name of the pragma that is ignored.

User response

Remove the pragma specification.

CCN8800	The omp threadprivate variable "%1$s" is not allowed to appear in the "%2$s" clause.

Explanation

Threadprivate variables are allowed in copyin, schedule and if clauses.

In the message text:

"%1$s" is the variable name, "%2$s" is the incorrect clause.

User response

Remove the clause or change the variable from being threadprivate.

CCN8801	The expression "%1$s" is not allowed to appear in the "%2$s"

clause, where a variable is expected.

Explanation

The expression "%1$s" is not a variable or name expression.

In the message text:

"%1$s" is an expression. "%2$s" is an omp clause.

User response

Only use variable names in the clause.

CCN8802	The "%1$s" qualifier in argument "%2$s" is ignored in the linkage signature for function "%3$s".

Explanation

Top-level cv-qualifiers in function arguments are not part of the function parameter types, and are not included in the function linkage signature.

In the message text:

"%1$s" is a cv-qualifier. "%2$s" is a function argument. "%3$s" is the function.

User response

Remove the cv-qualifier in question from the function argument in the specified function declaration.

CCN8803	Build with the "%1$s" compiler option to include cv-qualifiers of function arguments in function linkage signatures.

Explanation

Informational message about the option for including cv-qualifiers of function arguments in function linkage signatures.

In the message text:

"%1$s" is the compiler option that includes cv-qualifiers of function arguments in the function linkage signature.

User response

See the primary message.

CCN8804	The linkage specifier %1$s is invalid in "%2$s" mode.

Explanation

This linkage specifier has no meaning unless the object is built with the opposite XPLINK mode.

In the message text:

%1$s is a linkage specifier (i.e. OS_DOWNSTACK) "%2$s" is the current XPLINK mode.

User response

Turn on or off the XPLINK option.

CCN8805	"%1$s" has an invalid return type for the OS linkage specifier.

Explanation

Only functions with a return type of int or void may be used with the OS linkage specifier.

In the message text:

"%1$s" is the function.

User response

Check the return type or remove the OS linkage specifier.

CCN8806	The linkage specifier %1$s is not supported in the current context.

Explanation

This linkage specifier is not supported in the current context and its usage may result in incorrect output.

In the message text:

%1$s is a linkage specifier (i.e. OS_DOWNSTACK)

User response

Refer to the compiler documentation for correct usage in the current context.

CCN8807	The return type "%1$s" must not be used for a function that is declared to be extern "FORTRAN".

Explanation

A function that has extern "FORTRAN" language linkage can only return void, integral or double on certain platforms.

In the message text:

"%1$s" is the invalid function return type.

User response

Change the return type of the function to be void, integral or double.

CCN8808   **The return type "%1$s" must not be used for a function that is declared to be extern "COBOL".**

Explanation

A function that has extern "COBOL" language linkage can only return void on certain platforms.

In the message text:

"%1$s" is the invalid function return type.

User response

Change the return type of the function to be void.

CCN8809   **The function "%1$s" must not be declared __cdecl because the overridden function "%2$s" is not declared __cdecl.**

Explanation

A virtual function can only be declared __cdecl when the function in the base class is also __cdecl.

In the message text:

"%1$s" is the invalid function, "%2$s" is the overridden function.

User response

Remove the __cdecl qualifier from the derived class's overriding function or add the __cdecl qualifier to the base class's overridden function.

CCN8810   **The function "%1$s" must be declared __cdecl because the overridden function "%2$s" is declared __cdecl.**

Explanation

A virtual function must be declared __cdecl when the function in the base class is declared __cdecl.

In the message text:

"%1$s" is the invalid function, "%2$s" is the overridden function.

User response

Add a __cdecl qualifier to the derived class's overriding function or remove the __cdecl qualifier from the base class's overridden function.

CCN8811   **The linkage of the virtual function "%1$s" does not match the linkage of the overridden function "%2$s".**

Explanation

Virtual functions must have compatible linkage since there are several different ways of calling the function and they must all have the same linkage.

In the message text:

"%1$s" is the invalid function, "%2$s" is the overridden function.

User response

Ensure that the linkages match.

CCN8812   **The argument of a 'num_threads' clause must be a positive integer expression.**

Explanation

The number of threads in a team must be positive.

User response

Change the argument in the 'num_threads' clause to a positive integer expression.

CCN8813   **The argument of a 'num_threads' clause must be an integer expression.**

Explanation

The type of the 'num_threads' argument must be integer.

User response

Change the argument in the 'num_threads' clause to an expression of integer type.

CCN8814   **The threadprivate variable '%1$s' must be a file scope or namespace scope variable or static block scope variable.**

Explanation

The variable specified in the 'threadprivate' directive must be the outermost scope variable or local static variable.

In the message text:

"%1$s" is the threadprivate variable.

User response

Declare the variable static or move its declaration to the outermost scope.

CCN8815	The iteration variable must not be volatile.

Explanation

The iteration variable must not change in the loop body, therefore it must not be volatile.

User response

Remove the 'volatile' qualifier for the loop variable, or use another loop variable.

CCN8816	The threadprivate variable '%1$s' must not be referenced prior to the #pragma omp threadprivate directive.

Explanation

The variable '%1$s' must be listed in #pragma omp threadprivate directive before its first usage.

In the message text:

"%1$s" is the threadprivate variable.

User response

Move #pragma omp threadprivate directive before the first usage of variables listed in the directive.

CCN8817	The threadprivate directive for variable '%1$s' must not appear within a nested scope of that variable.

Explanation

The threadprivate directive for variable '%1$s' must appear within the scope of the variable declaration.

In the message text:

"%1$s" is the threadprivate variable.

User response

Move #pragma omp threadprivate directive so that it appears within the scope of the variable declaration and not in a nested scope.

CCN8819	Format string contains unknown conversion type character '%1$s' in conversion %2$s.

Explanation

An incorrect character has been specified in the format string syntax.

In the message text:

"%1$s" is the incorrect format character and %2$s is the conversion specification information.

User response

Remove the character in question.

CCN8820	The number of arguments is less than required by the format string.

Explanation

Not enough arguments have been specified for the format string.

User response

Add the argument required for the format string specification.

CCN8821	The number of arguments is greater than required by the format string.

Explanation

More arguments have been specified than required by the format string.

User response

Remove the extra arguments that are not required by the format string.

CCN8822	Format string is null.

Explanation

The specified format string is a null pointer.

User response

Specify a format string which is not null.

CCN8823   **The format string is empty.**

Explanation

The specified format string is an empty string.

User response

Specify a format string which contains at least one character.

CCN8824   **The format string contains '\\0'.**

Explanation

The specified format string contains an embedded '\\0' character.

User response

Remove the embedded '\\0' character from format string.

CCN8825   **The format string contains an illegitimate trailing '%%'.**

Explanation

The specified format string contains a dangling '%%' character.

User response

Either specify a conversion specification with the '%%' character or specify two '%%' characters for a percent character.

CCN8826   **The format string is not a string literal and format arguments are not given.**

Explanation

The specified format string is not a string literal and it may contain conversion specifications for which arguments are not specified.

User response

Make sure that enough arguments are specified for the format string.

CCN8827   **The format string is not a string literal and argument types are unchecked.**

Explanation

The specified format string is not string literal and its argument types cannot be checked.

User response

Make sure that correct argument types are specified for the format string.

CCN8828   **A wide character string is not permitted as a format string.**

Explanation

The specified format string contains wide characters.

User response

Remove the wide characters from the format string.

CCN8829   **The format string contains an operand number out of range.**

Explanation

%n$ operand number is out of range.

User response

Specify an operand number which matches the number of the argument for the format string.

CCN8830   **The format is missing a $ operand number.**

Explanation

%n$ operand number must be specified for all conversion specifications in the format string.

User response

Specify an operand number for conversion specifications which are missing operand numbers.

CCN8831   **Unused format argument (arg %1$s) precedes the used argument (arg %2$s) in the $-style format.**

Explanation

%n$ operand numbers in format string skip over unused arguments.

In the message text:

%1$s and %2$s are argument numbers.

User response

Specify operand numbers in the format string which do not skip over unused arguments.

CCN8832   **Not all given arguments are used by $-style format.**

Explanation

Extra unreferenced arguments appear in %n$ operand number format.

User response

Specify operand numbers which utilize all specified arguments.

CCN8833   **The format is taking no arguments and given an operand number.**

Explanation

The operand number is specified for a conversion taking no arguments.

User response

Remove the operand number in question.

CCN8834   **%%n$ operand number formats are unsupported by ISO C++ 98.**

Explanation

Operand number formats are an extension to ISO C++ 98.

User response

Do not use operand number formats in ISO C++ 98 mode.

CCN8835   **Invalid use of '%1$s' flag with '%2$s' %3$s format.**

Explanation

The specified flag name and format conversion combination is unsupported.

In the message text:

%1$s is a flag name, %2$s is a conversion name and %3$s is a function-style name.

User response

Remove the flag in question from the specified conversion.

CCN8836   **'%1$s' flag is disregarded when combined with the '%2$s' flag in a %3$s format.**

Explanation

The specified flag names conflict with each other.

In the message text:

%1$s and %2$s are flag names and %3$s is a function-style name.

User response

Remove one of the conflicting flags.

CCN8837   **'%1$s' flag is disregarded when combined with precision and '%2$s' printf format.**

Explanation

The specified flag is in conflict with the given precision and conversion.

In the message text:

%1$s is the flag name and %2$s is the printf conversion.

User response

Remove conflicting flag.

CCN8838   **'%1$s' flag is found repeating in %2$s format.**

Explanation

The flag has been specified multiple times.

In the message text:

%1$s is the flag name and %2$s is a function-style name.

User response

Remove the duplicate specifications of the same flag.

CCN8839   **The platform %1$s the use of a non-portable extension character '%2$s' in the format.**

Explanation

A non-portable extension character has been specified in a format string.

In the message text:

%1$s is 'supports' or 'does not support' and %2$s is an extension character in the format string.

User response

This extension character is not supported across platforms.

CCN8840	'%1$s' flag is unsupported by ISO C++ 98 in %2$s format.

Explanation

The specified flag is an extension to ISO C++ 98.

In the message text:

%1$s is a flag name and %2$s is a function-style name.

User response

Do not use the specified flag in ISO C++ 98 mode.

CCN8841	Invalid use of field width in '%1$s' %2$s format.

Explanation

The specified field width and format conversion combination is unsupported.

In the message text:

%1$s is a conversion and 2s is a function-style name.

User response

Remove the field width for the specified conversion.

CCN8842	Invalid use of precision in '%1$s' printf format.

Explanation

The specified precision and format conversion combination is unsupported.

In the message text:

%1$s is a conversion.

User response

Remove the precision for the specified conversion.

CCN8843	Argument '%1$s' is not an integer type: required for field %2$s.

Explanation

Argument must be int type.

In the message text:

%1$s is an argument number, %2$s is a field number.

User response

Specify an argument which is int type.

CCN8844	The use of the $ operand with '*' %1$s in a printf format may result in undefined behavior.

Explanation

The operand number conflicts with the variable field width or precision.

In the message text:

%1$s is width or precision.

User response

Do not specify an operand number with varible field width or precision.

CCN8845	Invalid %1$s format for %2$s argument type in argument %3$s.

Explanation

An invalid argument type has been specified for the given conversion.

In the message text:

%1$s is conversion, %2$s is argument type and %3$s is argument number.

User response

Specify an argument type that matches the given conversion type.

CCN8846	'%1$s' type character is incompatible with '%2$s' length modifer.

Explanation

An invalid type modifier has been specified for the given conversion.

In the message text:

%1$s is a conversion and %2$s is a type modifier.

User response

Change the type modifier for the given conversion.

CCN8847	Argument %1$s is expected to be a pointer type.

Explanation

The given conversion requires a pointer type.

In the message text:

%1$s is an argument number.

User response

A pointer argument type must be specified for the given conversion.

CCN8848	**Argument %1$s is %2$s through a null pointer.**

Explanation

The given conversion was given a const null pointer argument.

In the message text:

%1$s is an argument number and %2$s is either "reading" or "writing".

User response

Specify an argument which is not a null pointer.

CCN8849	**Argument %1$s is writing into a constant object.**

Explanation

The argument for a given conversion points to a constant object.

In the message text:

%1$s is an argument number.

User response

Specify an argument which does not point to a constant object.

CCN8850	**'%1$s' %2$s format is unsupported by ISO C++ 98.**

Explanation

The given conversion is an extension to ISO C++ 98.

In the message text:

%1$s is a conversion and %2$s is a function-style name.

User response

Do not use this conversion in ISO C++ 98 mode.

CCN8851	**'%1$s' %2$s length modifier unsupported by ISO C++ 98.**

Explanation

The given type modifier is an extension to ISO C++ 98.

In the message text:

%1$s is a length modifier and %2$s is a function-style name.

User response

Do not use this type modifier in ISO C++ 98 mode.

CCN8852	**Invalid %1$s format for %2$s argument type in argument %3$s.**

Explanation

An invalid argument type has been specified for the given conversion.

In the message text:

%1$s is a conversion, %2$s is an argument type and %3$s is an argument number.

User response

Specify an argument type that matches the given conversion type.

CCN8853	**Argument %1$s is expected to have type pointer to void.**

Explanation

The argument type for the given conversion is not pointer to void.

In the message text:

%1$s is the argument number.

User response

Change argument type for given conversion to pointer to void.

CCN8854	**Assignment suppression flag does not take an operand number.**

Explanation

An operand number was specified for the conversion along with a flag which suppresses argument assignment to that conversion.

User response

Remove the operand number.

CCN8855	**Invalid use of '*' flag with a length modifier in scanf format.**

Explanation

A '*' flag was specified for a conversion which has a length modifier.

User response

Remove the conflicting '*' flag for the given conversion.

CCN8856	Zero width cannot be specified for an input conversion.

Explanation

Zero width was specified in a format string.

User response

Specify a positive format width.

CCN8857	Format string contains out of range integer literal in conversion specification %1$s.

Explanation

The integer literal is not valid.

In the message text:

"%1$d" is the number of the conversion specification in format string.

User response

Change the integer literal.

CCN8858	Argument %1$s is expected to be of type pointer to pointer type.

Explanation

The given conversion expects pointer to pointer type.

In the message text:

%1$s is the argument number.

User response

Specify an argument of type pointer to pointer for the given conversion.

CCN8859	Argument %1$s is writing through a null pointer.

Explanation

The argument for the given conversion is a const null pointer.

In the message text:

%1$s is the argument number.

User response

Specify an argument for the given conversion which is not a const null pointer.

CCN8860	'%%[' format is missing closing ']'.

Explanation

For '%%[' the closing ']' was not specified.

User response

Specify the closing ']'.

CCN8861	Invalid use of '%1$s' flag with '%2$s' flag in %3$s format.

Explanation

Conflicting flags have been specified for the given format conversion.

In the message text:

%1$s and %2$s are flag names and %3$s is a function-style name.

User response

Remove one of the conflicting flags.

CCN8862	Field width unsupported in strict ISO C++ 98 mode.

Explanation

The field width for the given format is an extension to ISO C++ 98.

User response

Do not use field width in ISO C++ 98 mode.

CCN8863	Only the last two digits of the year are given by the '%1$s' conversion.

Explanation

The given conversion yields a 2-digit year.

In the message text:

%1$s is conversion name.

User response

Find an alternative conversion which yields a 4-year digit.

CCN8864   **Only the last two digits of year are given by '%1$s' conversion in some locales.**

Explanation

The given conversion yields a 2-digit year in some locales.

In the message text:

%1$s is the conversion name.

User response

Find an alternative conversion which yields a 4-digit year.

CCN8865   **Invalid use of '%1$s' modifier with '%2$s' strftime format.**

Explanation

An invalid combination of a modifier and a conversion was specified.

In the message text:

%1$s is a modifier name and %2$s is a conversion.

User response

Remove the conflicting modifier.

CCN8866   **'%1$s' modifier is found repeating in strftime format.**

Explanation

The given modifier has been specified multiple times.

In the message text:

%1$s is a modifier name.

User response

Specify modifier only one time for the given conversion.

CCN8867   **Invalid use of 'E' modifier with 'O' modifier in strftime format.**

Explanation

The E and O modifiers conflict with each other for the given format conversion.

User response

Remove one of the conflicting modifiers.

CCN8868   **The '%1$s' modifier is unsupported by ISO C++ 98 in the strftime format.**

Explanation

Modifiers are an extension to ISO C++ 98.

In the message text:

%1$s is modifier name.

User response

Do not use modifiers in ISO C++ 98 mode.

CCN8869   **The %1$s precision in strfmon format is empty.**

Explanation

An empty precision has been specified for strfmon format.

In the message text:

%1$s is Left or Right.

User response

Specify a number for the precision.

CCN8870   **Invalid multibyte character was found in the format string.**

Explanation

The multibyte character in the format string is invalid.

User response

Change the multibyte character.

CCN8871   **Format string argument must be a string type.**

Explanation

The format string argument number specified in __attribute__((format)) or __attribute__((format_arg)) must be a string type.

User response

Change the numeric value in the attribute.

CCN8872   **'...' is required for arguments to be formatted.**

Explanation

The arguments to be formatted in __attribute__((format)) must be an ellipsis.

User response

Change the numeric value in the attribute.

CCN8873	User function must return a string type.

Explanation

The return type specified in the declaration with __attribute__((format_arg)) must be a string type.

User response

Change the return type.

CCN8874	The '%1$s' modifier with '%2$s' format is unsupported by ISO C++ 98 in strftime format.

Explanation

The given modifier and format specification is an extension to ISO C++ 98.

In the message text:

%1$s is modifier name and %2$s is format name.

User response

Do not use the given modifier and format in ISO C++ 98 mode.

CCN8875	The '%1$s' attribute can only be applied to the definition of a non-static filescope variable.

Explanation

The attribute has no effect on filescope static or auto function scoped variables.

In the message text:

%1$s is the attribute name.

User response

Remove the attribute.

CCN8876	Attribute "aligned" cannot be used to decrease the alignment of "%1$s" and is ignored.

Explanation

Do not use the attribute specifier "aligned" to reduce the alignment of a variable or an aggregate.

In the message text:

"%1$s" is the variable name

User response

Remove the use of the attribute specifier "aligned", or increase the value.

CCN8877	The built-in function "%1$s" is not valid for this target system.

Explanation

The built-in function makes use of features not available on this target system.

In the message text:

"%1$s" is the built-in name

User response

Remove the builtin or move the source to a valid target system.

CCN8878	The built-in function "%1$s" is not valid for this architecture.

Explanation

The built-in function makes use of features not available with this architecture.

In the message text:

"%1$s" is the built-in name

User response

Remove the built-in or move the source to a valid architecture.

CCN8879	The built-in function "%1$s" requires option "%2$s".

Explanation

The built-in function depends on the option being set.

In the message text:

"%1$s" is the built-in name, "%2$s" is the required option.

User response

Set the required option or remove the built-in.

CCN8880   **The built-in function "%1$s" takes "%2$s" arguments.**

Explanation

The wrong number of arguments have been supplied to the built-in function.

In the message text:

"%1$s" is the built-in name, "%2$s" is the number of arguments.

User response

Correct the arguments to the built-in function call.

CCN8881   **The built-in function "%1$s"'s argument "%2$s" must be "%3$s".**

Explanation

A wrong argument type has been supplied to the built-in function.

In the message text:

"%1$s" is the built-in name, "%2$s" is the parameter number, and "%3$s" is the required type.

User response

Correct the type of the argument on the built-in function call.

CCN8882   **The built-in function "%1$s"'s argument "%2$s" must be in the range "%3$s".**

Explanation

An argument to the built-in function is out of the allowed range.

In the message text:

"%1$s" is the built-in name, "%2$s" is the parameter number, and "%3$s" is the valid range.

User response

Correct the value of the built-in argument to be in the allowable range.

CCN8883   **Inline function "%1$s" given attribute noinline.**

Explanation

The function is given noinline attribute because noinline has higher precedence.

In the message text:

"%1$s" is the function name.

User response

Remove one of the conflicting attributes.

CCN8884   **A temporary object reachable during exception unwinding may not have been constructed.**

Explanation

The logical operation may skip the temporary object construction, which may be destructed later if an exception is thrown from the same expression.

User response

Use -qeh=v6 option.

CCN8885   **The alignment of "%1$s" exceeds the maximum supported value of "%2$s". The alignment has been limited to "%2$s".**

Explanation

An alignment value exceeded the maximum supported value. The alignment may be ignored.

In the message text:

"%1$s" is the variable name. "%2$s" is the maximum supported value for alignment.

User response

Use an alignment less than or equal to the maximum.

CCN8887   **A flexible array member is not permitted in this scope.**

Explanation

A flexible array member is only allowed in the member scope of a class or struct.

User response

Remove the flexible array member.

CCN8888   **An array of aggregates that contain a flexible array member is not allowed.**

Explanation

A flexible array is not permitted when the enclosing struct is used as an array element type.

User response

Remove the flexible array member or remove the array declaration.

CCN8889	The pragma is in an invalid source location within another statement.

Explanation

This pragma causes a pragma statement to be generated but is located within another statement.

User response

Move the pragma before the parent statement or to within a set of braces "{" "}" following the parent statement to clarify its location.

CCN8890	The argument "%1$s" of the built-in function "%2$s" is invalid.

Explanation

The built-in function accepts only one of the predefined strings. See the documentation for this built-in function for valid arguments.

In the message text:

"%1$s" is the argument passed to the built-in function, "%2$s" is the built-in function name.

User response

Change the argument to one of the predefined strings.

CCN8891	The array member "%1$s" may only be followed by members of consistent type.

Explanation

A zero extent array or flexible array member may only be followed by members of the same type within a struct nest.

In the message text:

"%1$s" is the array member.

User response

Remove "%1$s" or modify the members that follow it so they are of the same type as "%1$s".

CCN8892	A variably modified type may not be thrown.

Explanation

Exception handling for variable length arrays is not defined.

User response

Change the thrown object type.

CCN8893	A catch handler may not catch a variably modified type.

Explanation

Exception handling for variable length arrays is not defined.

User response

Catch an appropriate pointer type instead.

CCN8894	The use of C99 variable length arrays in C++ is not portable.

Explanation

Variable length arrays are not defined in the C++ language.

User response

Use dynamically allocated arrays through library routines such as alloca or malloc.

CCN8895	A sizeof operator cannot be applied to a variable length array of unknown size.

Explanation

The result of a sizeof operator on a variable length array of unknown size is not known.

User response

Remove the call to sizeof.

CCN8896	A variable length array may not have linkage.

Explanation

A variable length array must be an automatic variable.

User response

Remove the static or extern storage class specifier.

CCN8897	A typeid expression of a variably modified type is not permitted.

Explanation

A typeid expression of variably modified type is not defined in the C++ language.

User response

Remove the typeid operation or change the expression.

CCN8898	Functions with parameters of variably modified type must have extern "C" linkage.

Explanation

Function overload resolution with functions having variable length array parameters is not defined in C++.

User response

Add the extern "C" linkage specification to the function.

CCN8899	A string literal is required for the format string.

Explanation

The specified format string is not a string literal.

User response

Make sure that correct argument types are specified for the format string.

CCN8900	Section "%1$s" is already specified as a "%2$s" section.

Explanation

The user section has already been specified as being another type of section. Data sections and text sections must have distinct names.

In the message text:

"%1$s" is the section name and "%2$s" is "text" or "data".

User response

Remove one of the declarations for the section.

CCN8901	A missing break statement allows fall-through to this case.

Explanation

A potential fall-through to this case exists as a result of a missing break statement.

User response

Make sure that the fall-through is intentional or add a break statement.

CCN8902	The function "%1$s" is declared using a type with no linkage.

Explanation

A function may not be declared in terms of something that has no scope linkage.

In the message text:

"%1$s" is the parameter name.

User response

Correct the offending function parameter or return type so that it has linkage or remove it from the function declaration.

CCN8903	A variable length array may not be specified in an OpenMP "%1$s" clause.

Explanation

Variable length arrays are not currently supported with OpenMP.

In the message text:

"%1$s" is an OpenMP clause.

User response

Remove the variable length array from the OpenMP clause variable list.

CCN8904	Non-static initialization of a flexible array member is not permitted.

Explanation

A flexible array member may not be initialized in this scope.

User response

Remove the initializers for the flexible array member.

CCN8905	The asm statement is not portable.

Explanation

The meaning of an asm statement is implementation-defined.

User response

Remove the asm statement.

CCN8906   **The __align specifier cannot be used to reduce the alignment of an aggregate or a variable.**

Explanation

One cannot use __align to restrict the alignment of a variable or an aggregate more than its natural alignment.

User response

Remove the __align specifier or change the specified value.

CCN8907   **The subscript %1$s is out of range. The valid range is 0 to %2$s.**

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

"%1$s" is the index into the array "%2$s" is the max index

User response

Change the index so it falls within the bounds of the array or increase the size of the array. This message is usually generated when the user tries to index the array with the size of the array and forgets to subtract one.

CCN8908   **The subscript %1$s is less than zero. The subscript of an array should be greater than or equal to zero.**

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

"%1$s" is the index into the array

User response

Change the index so it falls within the bounds of the array.

CCN8909   **The subscript %1$s is out of range. The only valid subscript is 0.**

Explanation

The user attempted to index an array with a value that is not within the bounds of the array.

In the message text:

"%1$s" is the index into the array

User response

Change the index so it falls within the bounds of the array or increase the size of the array. This message is usually generated when the user tries to index the array with the size of the array and forgets to subtract one.

CCN8910   **The template "%1$s" uses a file organization for tempinc, but tempinc is not being used.**

Explanation

The compiler determined that the implementation of the template is contained in a separate file. The compiler can handle this automatic instantiation if tempinc is enabled. An alternative is to use template registry. Please consult the documentation on tempinc and template registry for the best solution.

In the message text:

"%1$s" is the name of the template

User response

Enable the tempinc option or organize the source files to use template registry.

CCN8911   **Variable "%1$s" must be private in the enclosing context.**

Explanation

Variable listed in the copyprivate clause must be private in the enclosing context.

In the message text:

"%1$s" is the variable name.

User response

Check the scope of the variable in the copyprivate clause.

CCN8913   **Template specialization "%1$s" must match the formal definition.**

Explanation

The definition of a specialization does not match the formal template.

In the message text:

"%1$s" is the name of the specialization.

User response

Ensure that the declaration of the specialization matches the formal template.

CCN8914	A decimal floating-point type may not be thrown.

Explanation

Exception handling for decimal floating-point types is not supported.

User response

Change the thrown object type to a supported type.

CCN8915	A catch handler may not catch a decimal floating-point type.

Explanation

Exception handling for a decimal floating-point type is not supported.

User response

Change the catch object type to a supported type.

CCN8916	"pragma %1$s" conflicts with option "%2$s". The pragma is ignored.

Explanation

The specified pragma cannot be used under the specified option. The pragma is ignored.

In the message text:

"%1$s" is the pragma name. "%2$s" is the option value.

User response

Verify the option values, and either remove or modify it.

CCN8917	Specified register name "%1$s" can only be used to declare a variable with register storage class.

Explanation

A declaration may not be mapped to a register name.

In the message text:

"%1$s" is the mapped name.

User response

Use a different mapped name.

CCN8918	An OpenMP 'for' construct may only have one binding 'ordered' construct.

Explanation

An iteration of a loop with a 'for' construct must not execute more than one 'ordered' directive.

User response

Remove the extra 'ordered' construct.

CCN8922	The argument of a "%1$s" clause must be a positive constant integer expression.

Explanation

The value of the expression must be a constant value greater than zero.

In the message text:

"%1$s" is the omp clause name.

User response

Change the argument of the clause to a positive constant integer expression.

CCN8923	The value of the expression in the "collapse" clause must not be greater than the number of nested "for" loops enclosed by the "parallel for" directive.

Explanation

The value of the expression must not exceed the number of nested "for" loops enclosed by the "parallel for" directive.

User response

Change the argument of the clause to a positive constant integer expression that does not exceed the number of nested "for" loops enclosed by the "parallel for" directive.

CCN8924 **Cannot pass an argument of non-POD class type "%1$s" through ellipsis.**

Explanation

The value of the expression must be a constant value greater than zero.

In the message text:

"%1$s" is a class type.

User response

Check to see if the non-POD class was the intended argument and if so use a parameter to pass the argument with non-POD class type.

CCN8925 **The inner loop bounds and stride must not depend on the outer loop control variable "%1$s" within a construct with a "collapse" clause.**

Explanation

The loop boundaries must be free from dependencies on the outer loop control variables in order for the loop to be collapsed.

In the message text:

"%1$s" is the declaration with the dependency.

User response

Change the structure of the for loops to a rectangular form, which is free from dependencies.

CCN8926 **Branching out of a collapsed loop is not allowed.**

Explanation

The collapsed loop must not have any loop control flow statements.

User response

Remove the loop control flow statements.

CCN8927 **The "unroll" pragma cannot be used with a construct containing the "collapse" clause.**

Explanation

The pragma unroll and collapse clause are incompatible.

User response

Remove either the pragma unroll or the collapse clause.

CCN8928 **Integral constant "%1$s" has implied type unsigned long int under the non-C++0x language levels. It has implied type long long int under C++0x.**

Explanation

Note that the type of constant may affect the evaluation of the expression.

In the message text:

%1$s is an integer constant.

User response

Add explicit suffix to the constant if necessary.

CCN8931 **The scheduling type "%1$s" is not supported. "static" is assumed.**

Explanation

An unsupported scheduling has been specified with the OMP schedule directive.

In the message text:

"%1$s" is the user specified schedule type.

User response

Specify one of the supported scheduling types.

CCN8932 **"%1$s" has no parameter packs that can be expanded.**

Explanation

A pack expansion is not permitted.

In the message text:

"%1$s" is name of the invalid pack expansion.

User response

Remove the ellipsis.

CCN8933 **A template parameter pack must be the last template parameter in this context.**

Explanation

Only argument deduction is able to instantiate a parmameter pack that is not in the last position.

User response

Move the parameter pack to the end of the list of parameters.

CCN8934	Template parameter pack "%1$s" is not expanded. All template parameter packs must be expanded in this context.

Explanation

Parameter pack "%1$s" must be expanded.

In the message text:

"%1$s" is the name of the unexpanded pack.

User response

Expand the template parameter pack using an ellipsis.

CCN8935	A C++0x feature is being used. Use the extended0x langlvl or langlvl "%1$s" to enable this feature.

Explanation

The language level option must allow either C++0x or the specific C++0x feature.

In the message text:

"%1$s" is the language option to enable this C++0x feature.

User response

Use the extended0x language level or the specific language level for this C++0x feature.

CCN8936	Function "%1$s" should have static linkage.

Explanation

When the weakref function attribute is attached to the function "%1$s", the function should have static linkage.

In the message text:

"%1$s" is the function's name.

User response

Add static keyword to the function.

CCN8937	Mismatched pack lengths while expanding pack for "%1$s".

Explanation

The pack expansion failed because the packs have different lengths.

In the message text:

"%1$s" is the name of a pack expansion.

User response

In this pack expansion, packs must be the same length.

CCN8938	The operand of sizeof... must name a parameter pack.

Explanation

Sizeof... is only valid when called with an expression that names a parameter pack.

User response

Specify an expression that names a parameter pack.

CCN8939	A pack expansion is not allowed in this context.

Explanation

Template and function parameter packs can only be expanded in certain contexts.

User response

Avoid the expansion of a parameter pack in this context.

CCN8940	A template parameter pack cannot be a template template parameter, this has not been implemented.

Explanation

Variadic template template parameters have not been implemented in this release.

User response

Avoid using template template parameter packs.

CCN8941	A template parameter pack must not have a default argument.

Explanation

Default arguments cannot be specified for template parameter packs.

User response

Remove the default argument from the template parameter pack.

CCN8942   **A template parameter pack or expansion is not permitted in this context.**

Explanation

A variadic template parameter cannot be used here.

User response

Remove the parameter pack.

CCN8943   **Integral constant "%1$s" has implied type unsigned long long or is not allowed with "%2$s" under C++0x. Its implied type is not unsigned long long under non-C++0x language levels.**

Explanation

Note that the type of constant may affect the evaluation of the expression.

In the message text:

%1$s is an integer constant. %2$s is an langlvl suboption.

User response

Add explicit suffix to the constant if necessary.

CCN8947   **The function "%1$s" has a previous declaration with different top-level cv-qualifiers, which may cause link errors when used in another compilation unit under the option "%2$s".**

Explanation

Differences in top-level cv-qualifiers in function arguments are significant under older name mangling schemes.

In the message text:

"%1$s" is a the function declaration. "%2$s" is a compiler option that includes top-level cv-qualifiers of function arguments in the function linkage signature.

User response

Adjust the top-level cv-qualifiers of arguments in the previous function declaration to match the current function declaration.

CCN8952   **The attribute "%1$s" requires "%2$s" on function "%3$s"; the attribute is ignored.**

Explanation

The attribute "%1$s" is invalid without "%2$s".

In the message text:

"%1$s" is an attribute name, "%2$s" is a keyword, "%3$s" is a function name.

User response

Add "%2$s" to allow attribute "%1$s" or remove attribute "%1$s".

CCN8953   **"%3$s" redeclared "%2$s" without "%1$s" attribute.**

Explanation

The attribute "%1$s" is required when using "%2$s" if there are any other declaration that uses both.

In the message text:

"%1$s" is an attribute name, "%2$s" is a keyword, "%3$s" is a function name.

User response

Add attribute "%1$s" to allow "%2$s" or remove "%2$s".

CCN8954   **The attribute "malloc" is not valid on function "%1$s" since it has a return type other than a pointer or C++ reference. The attribute is ignored.**

Explanation

Functions declared with the "malloc" attribute need to have a pointer or reference return type.

In the message text:

"%1$s" is a function name.

User response

Change the return type or remove the "malloc" attribute.

CCN8955 Compilation may require C++0x Feature. Try using the option "%1$s".

Explanation

The language level option must allow the specific C++0x feature.

In the message text:

"%1$s" is a fine grain C++0x feature option to enable the C++0x feature.

User response

Use the specific C++0x feature option.

CCN8958 C++0x will reserve "%1$s" as a keyword whose C++0x feature can be enabled by "%2$s".

Explanation

The user should avoid using C++0x specific keyword as normal identifiers in non-C++0x mode.

In the message text:

"%1$s" is a C++0x keyword that is not reserved in non-C++0x level, %2$s is the corresponding fine grain C++0x feature option to enable the C++0x feature.

User response

Use non-C++0x keyword or specify -qnokeyword option.

CCN8959 The right side "%1$s" of the assignment expression of the OMP Atomic Read construct should be a lvalue.

Explanation

To perform an atomic read, the source should be a lvalue expression, which represent a valid memory location where the data to be read resides.

In the message text:

"%1$s" is the right side of the assignment expression under the OMP Atomic Read construct.

User response

Change the right side to a scalar lvalue expression, such as a simple variable, array element, reference, etc.

CCN8960 Expected two statements in the structured block of the OMP Atomic Capture construct.

Explanation

Expected two statements in the structured block of the OMP Atomic Capture construct.

User response

Create appropriate number and type of statements in the structured block of the OMP Atomic Capture construct.

CCN8961 The function "%1$s" cannot be defined in the current scope.

Explanation

The current scope is not one in which the function can be defined.

In the message text:

"%1$s" is the function name.

User response

Move the function definition to the correct scope.

CCN8962 The variable "%1$s" cannot be defined in the current scope.

Explanation

The current scope is not one in which the variable can be defined.

In the message text:

"%1$s" is the variable name.

User response

Move the variable definition to the correct scope.

CCN8963 The member "%1$s" does not match any member declared in its containing class.

Explanation

The member that is being defined outside of a class is not declared in the class.

In the message text:

"%1$s" is the member.

User response

Declare the variable or function as a member of the class.

CCN8964   **Candidate member in the containing class: "%1$s".**

Explanation

The member that is being defined outside of a class does not match any candidate member in the class.

In the message text:

"%1$s" is the candidate member.

User response

The member defined must match the declaration in the corresponding class.

CCN8966   **The name mangling schema for expression of decltype used as function return type is not implemented on the current product.**

Explanation

Avoid using expression of decltype used as function return type.

User response

See the primary message.

CCN8967   **The two occurrences of expression "x" are different: "%1$s" vs "%2$s".**

Explanation

The two occurrences of expression "x" are different.

In the message text:

The two occurrences are "%1$s" and "%2$s".

User response

Check the statement block and use same expression for both occurrences, or two expressions that evaluate to the same memory location.

CCN8968   **The parameter type "%1$s" for constexpr function or constructor "%2$s" is not a literal type**

Explanation

Each parameter type for a constexpr function or constructor shall be a literal types

In the message text:

"%1$s" is the name of the parameter type in error, "%2$s" is the name of the function or constructor

User response

Remove constexpr specifier from the function or constructor declaration

---

CCN8969   **The return type "%1$s" for constexpr function "%2$s" is not a literal type**

Explanation

Return type for a constexpr function shall be a literal types

In the message text:

"%1$s" is the name of the return type in error, "%2$s" is the name of the function

User response

Remove constexpr specifier from the function declaration

CCN8970   **A constexpr function "%1$s" shall not be declared virtual**

Explanation

A virtual function cannot be declared constexpr

In the message text:

"%1$s" is the name of the function in error

User response

Remove virtual specifier from the function declaration

CCN8971   **Invalid statement is found in the body of constexpr function or constructor "%1$s"**

Explanation

Not all statements are allowed in the body of a constexpr function or constructor

In the message text:

"%1$s" is the name of the function in error

User response

Check the function body and remove the statement not allowed

CCN8972	Function try block can not be used for a constexpr constructor "%1$s"

Explanation

The Body for constexpr constructor cannot be a function try block

In the message text:

"%1$s" is the name of the constructor in error

User response

Make the body a simple compound statement

CCN8973	The function body for constexpr constructor "%1$s" is not empty

Explanation

A constexpr constructor shall have empty function body

In the message text:

"%1$s" is the name of the constructor in error

User response

Make the function body empty

CCN8974	Type "%1$s" for constexpr variable or static data member "%2$s" is not a literal type

Explanation

A constexpr variable or static data member shall have literal type

In the message text:

"%1$s" is the type name in error, "%2$s" is the name of the variable or static data member

User response

Remove the constexpr specifier

CCN8975	Class type "%1$s" contains constexpr functions but is not a literal class

Explanation

A class type of which constexpr function is a member shall be a literal class

In the message text:

"%1$s" is the class type in error

User response

Remove constexpr member function declaration or make the class type a literal class

CCN8976	The "constexpr" specifier cannot be applied to non-static data member "%1$s"

Explanation

Specifier constexpr can only be used for variable or static data member

In the message text:

"%1$s" is the non-static data member in error

User response

Remove the constexpr specifier

CCN8977	The non-static data member "%1$s" is not initialized by constexpr constructor "%2$s"

Explanation

A constexpr constructor shall initialize every non-static data member

In the message text:

"%1$s" is the name of the data member, "%2$s" is the name of the constructor in error

User response

Add the data member to the constexpr constructor's initialization list

CCN8978	The variable "%1$s" should not be referenced by the expression "%2$s".

Explanation

The expression in the OMP atomic construct should not reference "%1$s".

In the message text:

"%1$s" is the atomic variable that is being manipulated, "%2$s" is a scalar expression that references the atomic variable.

User response

Fix the expression so that it does not reference or access "%1$s".

CCN8979   **Constructor "%1$s" called from the member initializer list of constexpr contructor "%2$s" is not constexpr**

Explanation

Every constructor invovled in initializing non-static data member and base class sub-object shall be a constexpr constructor

In the message text:

"%1$s" is the name of the constructor called, "%2$s" is the name of the constexpr constructor in error

User response

Use a different constructor or make the called constructor constexpr

CCN8980   **Sub-object of base class "%1$s" is not initialized by constexpr constructor "%2$s"**

Explanation

A constexpr constructor shall initialize every base class sub-object

In the message text:

"%1$s" is the name of the base class, "%2$s" is the name of the constructor in error

User response

Add explicit initializer for the base class sub-object

CCN8981   **The "omp for" pragma must be followed with one or more associated for-loops.**

Explanation

The "omp for" pragma must contain at least one for-loop.

User response

Add for-loop(s) after the pragma statement.

CCN8984   **Cannot take the address of a built-in function.**

Explanation

The built-in function "%1$s" cannot have its address taken.

In the message text:

"%1$s" is the name of the built-in function being address taken.

User response

Remove the address operator, or override the built-in function with your own.

CCN8985   **cv-qualifiers must not be added to function type.**

Explanation

The const and volatile qualifiers cannot added by the compiler when forming a function type.

User response

Remove the const or volatile specifiers.

CCN8986   **The use of decimal floating-point is invalid for architecture level "%1$s".**

Explanation

The effective architecture value specified on either #pragma arch_section or a compiler option does not support decimal floating-point.

In the message text:

"%1$s" is the architecture level specified on #pragma arch_section or a target architecture option.

User response

Use architecture value 7 and above, or remove the use of decimal floating-point.

CCN8987   **%1$s is disabled under option %2$s. Use %3$s instead.**

Explanation

%1$s is disabled under option %2$s. Use %3$s instead.

User response

Use the suggested option value instead.

CCN8988	The "%1$s" must not return NULL unless it is declared 'throw()' or %2$s is in effect.

Explanation

The throwing version of operator new or operator new[] should throw 'std::bad_alloc' exception rather than returning the null pointer.

In the message text:

%1$s is throwing version of operator new or operator new[] and %2$s is a compiler option.

User response

Specify 'throw()' exception specification or compile with %2$s.

Note

The following error messages may be produced by the compiler if the message file is itself invalid.

SEVERE ERROR EDC0090: Unable to open message file *&1*.
SEVERE ERROR EDC0091: Invalid offset table in message file *&1*.
SEVERE ERROR EDC0092: Message component *&1s* not found.
SEVERE ERROR EDC0093: Message file *&1* corrupted.
SEVERE ERROR EDC0094: Integrity check failure on msg *&1*
SEVERE ERROR EDC0095: Bad substitution number in message *&1*
SEVERE ERROR EDC0096: Virtual storage exceeded
ERROR: Failed to open message file. Reason *&1*.
ERROR: Unable to read message file. Reason *&1*.
ERROR: Invalid offset table in message file *&1*.
ERROR: Message component *&1s* not found.
ERROR: Message file *&1* corrupted.
ERROR: Integrity check failure on msg *&1* — retrieved *&2*.
ERROR: Message retrieval disabled. Cannot retrieve *&1*.
INTERNAL ERROR: Bad substitution number in message *&1*.

The previous messages are only generated in English.

Chapter 3. Utility messages

This topic contains information about the DSECT, CXXFILT, and CDAHLASM utility messages, and should not be used as programming interface information. For the localedef, iconv, and genxlt utility messages, refer to the *z/OS Language Environment Debugging Guide*. For the as and dbgld utility messages, refer to the *z/OS UNIX System Services Messages and Codes*.

Other return codes and messages

See the *z/OS Language Environment Debugging Guide* for messages and return codes for the following:

- prelinker and object library utility
- runtime messages and return codes
- localedef utility
- genxlt utility
- iconv utility
- System Programmer C (SPC)

DSECT utility messages

The following section describes return codes and messages that are issued by the DSECT utility.

Return codes

The DSECT utility issues the following return codes:

Table 4. Return codes from the DSECT utility	
Return Code	**Meaning**
0	Successful completion.
4	Successful completion, warnings issued.
8	DSECT utility failed, error messages issued.
12	DSECT utility failed, severe error messages issued.
16	DSECT utility failed, insufficient storage to continue processing.

Messages

The messages that the DSECT utility issues have the following format:

EDCnnnns text *<&s>* where:

nnnn
    error message number

s
    error severity

00
    informational message

10
    warning message

30
    error message

40
    severe error message

&s
    substitution variable

The DSECT utility issues the following messages:

EDC5500 10	**Option %s is not valid and is ignored.**

Explanation

The option specified in the message is not a valid DSECT Utility option or a valid option has been specified with an invalid value. The specified option is ignored.

User response

Rerun the DSECT Utility with the correct option.

EDC5501 30	**No DSECT or CSECT names were found in the SYSADATA file.**

Explanation

The SECT option was not specified or SECT(ALL) was specified. The SYSADATA was searched for all DSECTs and CSECTs but no DSECTs or CSECTs were found.

User response

Rerun the DSECT Utility with a SYSADATA file that contains the required DSECT or CSECT definition.

EDC5502 30	**Sub option %s for option %s is too long.**

Explanation

The sub option specified for the option was too long and is ignored.

EDC5503 30	**Section name %s was not found in SYSADATA File.**

Explanation

The section name specified with the SECT option was not found in the External Symbol records in the SYSADATA file. The C structure is not produced.

User response

Rerun the DSECT Utility with a SYSADATA file that contains the required DSECT or CSECT definition.

EDC5504 30	**Section name %s is not a DSECT or CSECT.**

Explanation

The section name specified with the SECT option is not a DSECT or CSECT. Only a DSECT or CSECT names may be specified. The C structure is not produced.

EDC5505 00	**No fields were found for section %s, structure is not produced.**

Explanation

No field records were found in the SYSADATA file that matched the ESDID of the specified section name. The C structure is not produced.

EDC5506 30	**Record length for file "%s" is too small for the SEQUENCE option, option ignored.**

Explanation

The record length for the output file specified is too small to enable the SEQUENCE option to generate the sequence number in columns 73 to 80. The available record length must be greater than or equal to 80 characters. The SEQUENCE option is ignored.

EDC5507 40	**Insufficient storage to continue processing.**

Explanation

No further storage was available to continue processing.

User response

Rerun the DSECT Utility with a larger region (MVS™).

EDC5508 30	**Open failed for file "%s": %s**

Explanation

This message is issued if the open fails for any file required by the DSECT Utility. The file name passed to fopen() and the error message returned by strerror(errno) is included in the message.

User response

The message text indicates the cause of the error. If the file name was specified incorrectly on the OUTPUT option, rerun the DSECT Utility with the correct file name.

EDC5509 40   %s failed for file "%s": %s

Explanation

This message is issued if any error occurs reading, writing or positioning on any file by the DSECT Utility. The name of the function that failed (Read, Write, fgetpos, fsetpos), file name and text from strerror(errno) is included in the message.

User response

This message may be issued if an error occurs reading or writing to a file. This may be caused by an error within the file, such as an I/O error or insufficient disk space. Correct the error and rerun the DSECT Utility.

EDC5510 40   Internal Logic error in function %s

Explanation

The DSECT Utility has detected that an error has occurred while generating the C structure. Processing is terminated and the C structure is not produced.

User response

This may be caused by an error in the DSECT Utility or by incorrect input in the SYSADATA file. Contact your system administrator.

EDC5511 10   No matching right parenthesis for %s option.

Explanation

The option specified had a sub option beginning with a left parenthesis but no right parenthesis was present.

User response

Rerun the DSECT Utility with the parenthesis for the option correctly paired.

EDC5512 10   No matching quote for %s option.

Explanation

The OUTPUT option has a sub option beginning with a single quote but no matching quote was found.

User response

Rerun the DSECT Utility with the quotes for the option correctly paired.

EDC5513 10   Record length too small for file "%s".

Explanation

The record length for the Output file specified is less than 10 characters in length. The minimum available record length must be at least 10 characters.

User response

Rerun the DSECT Utility with an output file with a available record length of at least 10 characters.

EDC5514 30   Too many suboptions were specified for option %s.

Explanation

More than the maximum number of suboptions were specified for the particular option. The extra suboptions are ignored.

User response

Check the syntax of the DSECT utility option in the C/C++ User's Guide, and remove the extra suboption(s).

EDC5515 00   HDRSKIP option value greater than length for section %s, structure is not produced.

Explanation

The value specified for the HDRSKIP option was greater than the length of the section. A structure was not produced for the specified section.

User response

Rerun the DSECT Utility with a smaller value for the HDRSKIP option.

EDC5516 10   SECT and OPTFILE options are mutually exclusive, OPTFILE option is ignored

Explanation

Both the SECT and OPTFILE options were specified, but the options are mutually exclusive.

User response

Rerun the DSECT Utility with either the SECT or OPTFILE option.

EDC5517 10	Line %i from "%s" does not begin with SECT option

Explanation

The line from the file specified on the OPTFILE option did not begin with the SECT option. The line was ignored.

User response

Rerun the DSECT Utility without OPTFILE option, or correct the line in the input file.

EDC5518 10	setlocale() failed for locale name "%s".

Explanation

The setlocale() function failed with the locale name specified on the LOCALE option. The LOCALE option was ignored.

User response

Rerun the DSECT Utility without LOCALE option, or correct the locale name specified with the LOCALE option.

EDC5519 10	Long names were detected and truncated. Check output.

Explanation

The dsect utility detected at least one name whose length exceeds the maximum allowed, and has truncated the name, and appended "..." to the end of the name to signify the condition. If the input name is within limits, and the UNIQUE option is specified, the mapping of national characters in the input name could have extended the name length beyond the maximum allowed.

User response

Check the dsect utility output. Long names are truncated and this is indicated by "..." at the end of the

name. Modify the UNIQUE option field if applicable, or modify the input name so that it does not exceed the maximum length when expanded.

EDC5520 40	Architecture Level %i of SYSADATA is not supported. The latest supported level is %d

Explanation

The SYSADATA file has probably been produced by a recent HLASM release which is not yet supported by the DSECT utility.

User response

Contact your IBM representative.

EDC5521 40	Architecture Level %i of SYSADATA is not supported. The earliest supported level is %d

Explanation

The SYSADATA file has probably been produced by an obsolete HLASM release.

User response

Use a supported HLASM release to produce the SYSADATA file.

EDC5522 10	Edition %d, SYSADATA level %d of record type X"%04x" - %s - is not supported. Edition %d is assumed.

Explanation

The likely reason is that HLASM maintenance has introduced an updated layout of this record type. This should not cause a problem unless the offsets of fixed fields processed by the DSECT utility have changed. The message can be ignored unless the produced output is incorrect.

User response

If the DSECT utility is producing incorrect output, then please contact your IBM representative.

CXXFILT utility messages

The following section describes return codes and messages that are issued by the CXXFILT utility.

Return codes

The CXXFILT utility issues the following return codes:

Table 5. Return codes from the CXXFILT utility	
Return Code	**Meaning**
0	Processing successful: CXXFILT processing completed successfully.
4	A warning was issued and a result was generated.
8	CXXFILT utility failed, possibly due to a read error.
16	CXXFILT utility failed.

Messages

The CXXFILT utility issues the following messages:

CCN9500   **Cannot open the following file: @1 -- ignored.**

Explanation

The specified file cannot be opened for reading or does not exist.

User response

Ensure that the file exists and is readable.

CCN9501   **Cannot continue reading input.**

Explanation

A read error occurred while reading the input stream.

User response

Ensure that the input stream is still available and try again.

CCN9502   **No options specified after (.**

Explanation

A (indicating start of options was encountered but no options followed.

User response

Ensure that the input stream is still available and try again.

CCN9503   **An invalid option (@1) was specified -- ignored.**

Explanation

An invalid option was specified.

User response

Refer to the z/OS or OS/390® C/C++ User's Guide under cxxfilt for valid options.

CCN9504   **Option (@1) was specified with too few suboptions. @2 suboption(s) required -- ignored.**

Explanation

Not all the required suboptions were supplied.

User response

Refer to the z/OS or OS/390 C/C++ User's Guide under cxxfilt for the number of required suboptions.

CCN9505   **Option (@1) was specified with too many suboptions. @2 suboption(s) required -- ignored.**

Explanation

More suboptions were supplied than what is allowed by this option.

User response

Refer to the z/OS or OS/390 C/C++ User's Guide under cxxfilt for the number of required suboptions.

CCN9506   **Option (@1) requires a positive suboption -- ignored.**

Explanation

This error occurred because the specified suboptions for this option are invalid. Only positive suboptions are allowed.

User response

Refer to the z/OS or OS/390 C/C++ User's Guide under cxxfilt for the allowed suboptions.

CCN9507　　　**Internal Error. Contact your Service Representative.**

Explanation

The cxxfilt utility has malfunctioned.

User response

Please report this problem.

CCN9508　　　**No negative form for option @1 -- ignored.**

Explanation

The specified option does not have a negative form.

User response

Refer to the z/OS or OS/390 C/C++ User's Guide under cxxfilt for valid options.

CCN9509　　　**An incomplete option (@1) has been specified. -- ignored**

Explanation

The specified option is incomplete.

User response

Refer to the z/OS or OS/390 C/C++ User's Guide under cxxfilt for valid options.

CDAHLASM utility messages

The following section describes return codes and messages that are issued by the CDAHLASM utility.

Return codes

The CDAHLASM utility issues the following return codes:

Table 6. Return codes from the CDAHLASM utility	
Return Code	**Meaning**
0	Assembled successfully.
2	Assembled with a notice.
4	Assembled with a warning.
16	Error assembling or CDAHLASM error.

Messages

The CDAHLASM utility issues the following messages:

CDA3401　　　**The assemble step ended with rc = number.**

Explanation

Assemble step completed with a non-zero return code.

In the message text:

number is the return code from the assemble step.

User response

This does not necessarily mean that you need to take action. If necessary, correct the error indicated by the preceding messages, and rerun the CDAHLASM utility.

CDA3402　　　**Exactly one source file must be specified.**

Explanation

The CDAHLASM utility requires that you specify exactly one source file. The source file must be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

User response

Rerun the CDAHLASM utility and specify exactly one source file.

CDA3403   **The PDS** *string* **cannot be assembled. Specify a PDS member instead.**

Explanation

The indicated input file is a PDS. The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of the input source file.

User response

Rerun the CDAHLASM utility and specify a PDS member.

CDA3404   **The PDSE** *string* **cannot be assembled. Specify a PDSE member instead.**

Explanation

The indicated input file is a PDSE. The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of the input source file.

User response

Rerun the CDAHLASM utility and specify a PDSE member.

CDA3405   **The VSAM file** *string* **cannot be assembled.**

Explanation

The indicated input file is a VSAM file. The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of the input source file.

User response

Rerun the CDAHLASM utility and specify a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

CDA3406   **The block special file** *string* **cannot be assembled.**

Explanation

The indicated input file is a block special file. The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of the input source file.

User response

Rerun the CDAHLASM utility and specify a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

CDA3407   **The character special file** *string* **cannot be assembled.**

Explanation

The indicated input file is a character special file. The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of the input source file.

User response

Rerun the CDAHLASM utility and specify a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

CDA3408   **The directory** *string* **cannot be assembled. Specify a UNIX System Services file instead.**

Explanation

The indicated input file is a directory. The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of the input source file.

User response

Rerun the CDAHLASM utility and specify a UNIX System Services file.

CDA3409	The socket file *string* cannot be assembled.

Explanation

The indicated input file is a socket file. The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of the input source file.

User response

Rerun the CDAHLASM utility and specify a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

CDA3412	The file *string1* cannot be opened with attributes: *string2*. errno: *number*, strerror: *string3*.

Explanation

The CDAHLASM utility requires permission to open the indicated file with the indicated permission.

In the message text:

string1 is the processing file name. *string2* contains the file attributes passed into fopen(). *number* is the errno set by fopen(). *string3* is the error message associated with errno.

User response

If the indicated file already exists on your system, rename the file and rerun the CDAHLASM utility. Review the fopen() information in the XL C/C++ Run-Time Library Reference and use the errno to determine the cause of the error.

CDA3416	DD names passed to assembler:

Explanation

This is the message header for displaying the data definition names that are being passed to the assembler. A list of data definition names will follow

this message. The message is issued when the VERBOSE option is specified for the CDAHLASM utility.

User response

Rerun the CDAHLASM utility without the VERBOSE option.

CDA3417	The DLL *string* is not found.

Explanation

The indicated Common Debug Architecture run-time library cannot be found.

In the message text:

string is the name of the Common Debug Architecture run-time library.

User response

The indicated Common Debug Architecture run-time library should be installed in the SCEERUN2 data set. Verify that the run-time library is installed properly.

CDA3418	The data set does not exist.

Explanation

This message provides the cause of the failure in data definition name allocation. Message CDA3421 provides the name of the data set.

User response

Make sure the data set is properly allocated.

CDA3419	A Ddpi error has been encountered: *number*.

Explanation

An error has occured while generating DWARF 4.0 debug information.

In the message text:

number is the error number generated by the libddpi APIs.

User response

If you do not require debug information, rerun the CDAHLASM utility with the NODEBUG option. Otherwise, provide the indicated error number to the IBM service representative responsible for your installation.

CDA3420	An error has been encountered in *string*.

Explanation

The indicated message text contains the phase where the error has occured. The known phases are:

- ESI (extraction of symbol information)
- A2D (ADATA to DWARF conversion)
- POM (production of object map)

In the message text:

string is the phase where the error has occured.

User response

This is an internal error. Provide the indicated error text to the IBM service representative responsible for your installation.

CDA3421	An error has occurred when establishing the DD name for *string1*. *string2*

Explanation

An error has occured while the CDAHLASM utility uses the SVC99 service to allocate the data definition name.

In the message text:

string1 is the file name. *string2* is the SVC99 message.

User response

The cause of the error is indicated in the second part of the message. If the cause of the error is still unclear, rerun the CDAHLASM utility with the VERBOSE option. Provide this information to the IBM service representative responsible for your installation.

CDA3422	An error has occurred when reading *string1*: errno: *number*, strerror: *string2*.

Explanation

Unable to read from the indicated file.

In the message text:

string1 is the file being fread(). *number* is the errno set by fread(). *string2* is the error messsage associated with errno.

User response

This is an internal error. Rerun the CDAHLASM utility with the VERBOSE option. Provide this information to the IBM service representative responsible for your installation.

CDA3423	An error has occurred when writing *string1*: errno: *number*, strerror: *string2*.

Explanation

Unable to write to the indicated file.

In the message text:

string1 is the file being fwrite(). *number* is the errno set by fwrite(). *string2* is the error messsage associated with errno.

User response

This is an internal error. Rerun the CDAHLASM utility with the VERBOSE option. Provide this information to the IBM service representative responsible for your installation.

CDA3424	An error has occurred: *string*.

Explanation

This is a generic error message. Refer to the error message text for a description of the error.

In the message text:

string contains the cause of the error and the method of recovery.

User response

Refer to the error messsage text for information on how to recover from the error.

CDA3425	The file *string* does not exist.

Explanation

The CDAHLASM utility cannot find the indicated file.

In the message text:

string is a file name.

User response

Verify that the file name specified is correct and the proper permissions are set.

CDA3426	An incompatible DLL has been detected. 0x*hexnum1* is the LIBDDPI_DLL_VERSION with which *string* was compiled.

0x*hexnum2* **is the
LIBDDPI_DLL_VERSION of the
DLL.**

Explanation

The Common Debug Architecture run-time version is
outdated.

In the message text:

hexnum1 is the Common Debug Architecture run time
that is compiled with the CDAHLASM utility. *string* is
one of the source file names in the CDAHLASM utility.
hexnum2 is the Common Debug Architecture run time
that is being used to run the CDAHLASM utility.

User response

Contact the IBM service representative responsible for
your installation and verify that you have the latest
Common Debug Architecture run time installed.

CDA3427	An incompatible DLL has been detected. 0x*hexnum1* is the LIBELF_DLL_VERSION with which *string* was compiled. 0x*hexnum2* is the LIBELF_DLL_VERSION of the DLL.

Explanation

The Common Debug Architecture run-time version is
outdated.

In the message text:

hexnum1 is the Common Debug Architecture run time
that is compiled with the CDAHLASM utility. *string* is
one of the source file names in the CDAHLASM utility.
hexnum2 is the Common Debug Architecture run time
that is being used to run the CDAHLASM utility.

User response

Contact the IBM service representative responsible for
your installation and verify that you have the latest
Common Debug Architecture run time installed.

CDA3428	The listing file name is too long. It must not be more than *number* characters.

Explanation

The specified listing file name is too long.

In the message text:

number is the maximum character length for the file
name.

User response

Provide a listing file name with less than the indicated
number of characters.

CDA3429	The object file name is too long. It must not be more than *number* characters.

Explanation

The specified object file name is too long.

In the message text:

number is the maximum character length for the file
name.

User response

Provide an object file name with less than the
indicated number of characters.

CDA3432	Out of memory allocating *number* bytes for *string*.

Explanation

The CDAHLASM utility ran out of memory trying to
assemble the file. This sometimes happens with large
input files.

In the message text:

number is the number of bytes CDAHLASM tries to
allocate. *string* contains the name of the function trying
to allocate the memory.

User response

Shut down any large processes that are running.
You may need to specify the runtime option
HEAP(,,,FREE,,) to prevent the CDAHLASM utility from
running out of memory.

CDA3433	Out of memory allocating *number* bytes.

Explanation

The CDAHLASM utility ran out of memory trying to
assemble the file. This sometimes happens with large
input files.

In the message text:

number is the number of bytes CDAHLASM tries to
allocate.

User response

Shut down any large processes that are running. You may need to specify the runtime option HEAP(,,,FREE,,) to prevent the CDAHLASM utility from running out of memory.

CDA3434   **SVC99 error code 0x*hexnum1*, info code 0x*hexnum2*.**

Explanation

This message provides the cause of the failure in data definition name allocation. Message CDA3421 provides the name of the data set.

In the message text:

hexnum1 is the SVC99 error code. *hexnum2* is the SVC99 information code.

User response

Make sure the data set is properly allocated. If the error still persists, rerun the CDAHLASM utility with the VERBOSE option. Provide this information to the IBM service representative responsible for your installation.

CDA3437   **Unable to allocate *string1* data set concatenation. The first data set in error is *string2*.**

Explanation

The CDAHLASM utility is unable to process the indicated data set.

In the message text:

string1 is a list of concatenated data set names. *string2* is the data set that is not allocated.

User response

Verify that the data set exists and is properly allocated.

CDA3438   **Unable to load the assembler compiler.**

Explanation

The CDAHLASM utility cannot load the assembler compiler.

User response

Rerun the CDAHLASM utility. If the problem persists, contact the IBM service representative responsible for your installation.

CDA3439   **Unable to obtain the file information for *string*.**

Explanation

The CDAHLASM utility requires the input source file to be a sequential data set, a PDS member, a PDSE member or a UNIX System Services file.

In the message text:

string is the name of input source file.

User response

Rerun the CDAHLASM utility and specify a valid file name.

CDA3440   **Unable to open the debug file *string*.**

Explanation

The indicated file cannot be opened for writing.

In the message text:

string is the name of the debug file.

User response

If a file with the same name already exists, give the existing file another name before you rerun the CDAHLASM utility.

CDA3441   **Unable to open the listing file *string*.**

Explanation

The indicated file cannot be opened for writing.

In the message text:

string is the name of the listing file.

User response

If a file with the same name already exists, make sure it is renamed to another name before you rerun the CDAHLASM utility.

CDA3442   **Unable to open the object file *string*.**

Explanation

The indicated file cannot be opened for writing.

In the message text:

string is the name of the object file.

User response

If a file with the same name already exists, make sure it is renamed to another name before you rerun the CDAHLASM utility.

CDA3443	Unable to open the source file *string* for read.

Explanation

The indicated file cannot be opened for reading.

In the message text:

string is the name of the input source file.

User response

Make sure the source file exists and has the correct read permission.

CDA3491	An invalid option *string* is specified.

Explanation

The indicated option is not a valid CDAHLASM option.

In the message text:

string is an invalid option.

User response

Rerun the CDAHLASM utility and specify a valid option.

CDA3495	The debug file name is too long. It must not be more than *number* characters.

Explanation

The specified debug file name is too long.

In the message text:

number is the maximum character length for the file name.

User response

Provide a debug file name with less than the indicated number of characters.

CDA3500	Error reading the option file

string.

Explanation

The option file specified must be either a sequential data set, a PDS member, a PDSE member or a UNIX System Services file. It must also have read permission.

In the message text:

string is the name of the option file.

User response

Rerun the CDAHLASM utility and specify a sequential data set, a PDS member, a PDSE member or a UNIX System Services file with read permission.

CDA3505	A Ddpi error *number1* has been encountered in file *string* at line *number2*.

Explanation

An error has occurred while generating DWARF debug information.

In the message text:

number1 is the error number generated by the libddpi APIs. *string* is the libddpi file name where the error occurs. *number2* is the line number where the error occurs.

User response

If you do not require debug information, rerun the CDAHLASM utility with the NODEBUG option. Otherwise, provide the indicated error number, file name, and line number to the IBM service representative responsible for your installation.

CDA3506	Unable to update the assembler source file *string1*.

Explanation

The user provided debug side name is different from the one in the assembler source file. The CDAHLASM utility requires write permission to update the assembler source file with the user provided debug side file name.

In the message text:

string1 is the assembler file name.

User response

Make sure you have permission to write to the assembler source file.

CDADBGLD utility messages

The following section describes return codes and messages that are issued by the CDADBGLD utility.

Return codes

The CDADBGLD utility issues the following return codes:

Table 7. Return codes from the CDADBGLD utility	
Return Code	**Meaning**
0	Successful completion.
4	Warning.
8	Error.
12	Severe Error.

Messages

The CDADBGLD utility issues the following messages:

CDA1002   **An invalid option *string* is specified.**

Explanation

The indicated option is not a valid CDADBGLD option.

In the message text:

string is an invalid option.

User response

Rerun the CDADBGLD utility and specify a valid option.

CDA1003   **BINDER API failed. retcode:*number1*, rsncode:0x*number2***

Explanation

CDADBGLD is unable to retrieve information from the input module. The BINDER API has failed.

In the message text:

number1 is the return code from BINDER API.
number2 is the reason code from BINDER API.

User response

This is an internal error. Provide the indicated error text to the IBM service representative responsible for your installation.

CDA1004   **The file *string* cannot be opened.**

Explanation

Either the input file does not exist, or the permissions for the input file or the directory containing the input file do not have read or search permission set.

In the message text:

string is the input file name.

User response

Ensure the input file exists and that the input file and the directories containing the input file have read and search permissions set.

CDA1005   **No debug information was found in *string*.**

Explanation

CDADBGLD is unable to locate any debug information within the input file. No output file will be generated.

In the message text:

string is the input file name.

User response

Compile at least one compilation unit with the debug compiler option.

CDA1006   **Out of memory allocating *number* bytes for *string*.**

Explanation

The CDADBGLD utility ran out of memory processing the input file. This may happen with a large input file.

In the message text:

number is the number of bytes CDADBGLD tries to allocate. *string* contains the name of the function trying to allocate the memory.

User response

Shut down any large processes that are running. You may need to specify the runtime option HEAP(,,,FREE,,) to prevent the CDADBGLD utility from running out of memory.

CDA1007	INTERNAL UTILITY ERROR: Procedure *string*:*number*.

Explanation

An internal utility error occurred.

In the message text:

string is the procedure where the error has occured. *number* is the line number where the error has occurred.

User response

This is an internal error. Provide the indicated error text to the IBM service representative responsible for your installation.

CDA1008	The DLL *string* is not found.

Explanation

The indicated Common Debug Architecture runtime library cannot be found.

In the message text:

string is the name of the Common Debug Architecture runtime library.

User response

The indicated Common Debug Architecture runtime library should be installed in the SCEERUN2 data set.Verify that the runtime library is installed properly.

CDA1009	An incompatible DLL has been detected. The utility requires LIBDDPI_DLL_VERSION to be at least 0x*number1*. The version found in the system is 0x*number2*.

Explanation

The Common Debug Architecture runtime version is outdated.

In the message text:

number1 is the Common Debug Architecture run time that CDADBGLD is compiled with. *number2* is the Common Debug Architecture run time that is currently being used.

User response

Contact the IBM service representative responsible for your installation and verify that you have the latest Common Debug Architecture run time installed.

CDA1010	An incompatible DLL has been detected. The utility requires LIBELF_DLL_VERSION to be at least 0x*number1*. The version found in the system is 0x*number2*.

Explanation

The Common Debug Architecture runtime version is outdated.

In the message text:

number1 is the Common Debug Architecture run time that CDADBGLD is compiled with. *number2* is the Common Debug Architecture run time that is currently being used.

User response

Contact the IBM service representative responsible for your installation and verify that you have the latest Common Debug Architecture run time installed.

CDA1011	Cannot find the function *string1* in DLL *string2*.

Explanation

The indicated function cannot be found in the Common Debug Architecture runtime library.

In the message text:

string1 is the name of the function in the Common Debug Architecture runtime library. *string2* is the name of the Common Debug Architecture runtime library.

User response

Contact the IBM service representative responsible for your installation and verify that you have the latest Common Debug Architecture run time installed.

CDA1012	The debug file *string* cannot be opened for reading.

Explanation

The file permissions for the debug file do not have read permissions set.

In the message text:

string is the debug file name.

User response

Ensure the debug file exists and that it has read permissions set.

CDA1013	The module map file *string* cannot be opened for writing.

Explanation

The file or directory permissions for the module map file do not have write permissions set.

In the message text:

string is the module map file.

User response

Ensure the CDADBGLD utility is being run from a directory with write permission, and the file has write permission if it already exists.

CDA1014	The input file *string* cannot be processed due to the EDIT=NO attribute.

Explanation

CDADBGLD cannot process modules that have been bounded with the EDIT=NO option.

In the message text:

string is the file name of the input module.

User response

Specify binder option EDIT=YES when binding the module.

CDA1015	The input file *string* cannot be processed.

Explanation

The BINDER API has failed while processing the input module.

In the message text:

string is the file name of the input module.

User response

Refer to the Program Management documentation for information about the BINDER API return code and reason code.

CDA1016	The object file produced from *string* contains ISD debug information.

Explanation

The CDADBGLD utility is converting ISD debug information into DWARF debug information. This process may degrade the performance of the CDADBGLD utility.

In the message text:

string is the source file name.

User response

To increase the performance of the CDADBGLD utility, please recompile the specified source file with the -g compiler option, rebind your application, and then rerun the CDADBGLD utility.

CDA1017	The debug side file *string* is outdated.

Explanation

The MD5 signature within the specified debug side file does not match the MD5 signature within the input module.

In the message text:

string is the debug side file name.

User response

Recompile the corresponding source file, rebind your application, and then rerun the CDADBGLD utility.

CDA1018	An error has occurred while processing the ISD debug information for *string*.

Explanation

The CDADBGLD utility is unable to convert the ISD debug information into DWARF debug information.

In the message text:

string is the source file name.

User response

Recompile the specified source file with the -g compiler option, rebind your application, and then rerun the CDADBGLD utility. If this is not a viable option, please contact the IBM service representative responsible for your installation.

CDA1019　　　**An error has occurred while processing the DWARF debug information for *string*.**

Explanation

The CDADBGLD utility is unable to process the DWARF debug information in the specified debug side file.

In the message text:

string is the debug side file name.

User response

This is an internal error. Provide the indicated error text to the IBM service representative responsible for your installation.

CDA1020　　　**The source file *string* cannot be opened for reading.**

Explanation

Either the source file does not exist, or the file permissions for the source file do not have read permissions set. The contents of the source file will not be added to the module map.

In the message text:

string is the source file name.

User response

Ensure the source file exists and that it has read permissions set.

CDA1021　　　**Error writing to module map file *string*.**

Explanation

There may be insufficient disk space to write to the file.

In the message text:

string is the module map file.

User response

Ensure there is enough disk space available.

CDA1022　　　**Failed to link debug information.**

Explanation

Object was compiled with "DEBUG(NOFILE)" option.

User response

To get debug file compile the object with option "DEBUG(FILE)".

Chapter 4. z/OS XL C/C++ legacy class libraries messages

This topic contains information about the XL C/C++ legacy class libraries messages that are included with the current release and should not be used as programming interface information.

The following information shows the format of these messages:

Message format: **CLBnnnn<&n>** text where:

nnnn
 error message number

&n
 error severity

text
 message which appears on the screen

CLB9900 **An attempt to allocate memory has failed.**

Explanation

The attempt to obtain memory in order to satisfy the current library request has failed. It cannot be performed on a collection because the collection is not empty.

System action

The requested function will fail.

User response

Run the program in a larger region or use the HEAP(,,FREE) runtime option instead of the HEAP(,,KEEP) option.

CLB9901 **IOStreams do not support Record Mode I/O.**

Explanation

The application is attempting to initialize an IOStreams object to perform Record Mode I/O. IOStream objects do not support Record Mode input and output.

System action

The attempt to initialize the object failed. The program continues to execute.

User response

Remove the "type=record" specification from the constructor or open() function call.

CLB9902 **Too many characters.**

Explanation

The application called the form() function with a format specifier string that caused form() to write past the end of the format buffer. form() is an obsolete interface provided in stream.h for compatibility with old code.

System action

Execution is stopped.

User response

Split the call to the form() function into two or more calls.

CLB9903 **There was a singularity; the application could not take the log of (0.0, 0.0).**

Explanation

The application is attempting to take the log of (0.0, 0.0).

System action

Execution is stopped.

User response

Correct the value passed to the log() function and resubmit.

CLB9904 **The attempt to release the mutex handle failed.**

Explanation

There was an internal error: pthread_mutex_destroy() failed.

System action

Execution is stopped.

User response

Note the return code and error number to identify the cause of the problem and inform IBM C++ Service and Support.

CLB9905	The attempt to lock the mutex handle failed.

Explanation

There was an internal error: pthread_mutex_lock() failed.

System action

Execution is stopped.

User response

Note the return code and error number to identify the cause of the problem and inform IBM C++ Service and Support.

CLB9906	The attempt to unlock the mutex handle failed.

Explanation

Internal error: pthread_mutex_unlock() failed.

System action

Execution is stopped.

User response

Note the return code and error number to identify the cause of the problem and inform IBM C++ Service and Support.

Appendix A. Accessibility

Accessible publications for this product are offered through IBM Documentation (www.ibm.com/docs/en/zos).

If you experience difficulty with the accessibility of any z/OS information, send a detailed message to the Contact the z/OS team web page (www.ibm.com/systems/campaignmail/z/zos/contact_z) or use the following mailing address.

> IBM Corporation
> Attention: MHVRCFS Reader Comments
> Department H6MA, Building 707
> 2455 South Road
> Poughkeepsie, NY 12601-5400
> United States

Accessibility features

Accessibility features help users who have physical disabilities such as restricted mobility or limited vision use software products successfully. The accessibility features in z/OS can help users do the following tasks:

- Run assistive technology such as screen readers and screen magnifier software.
- Operate specific or equivalent features by using the keyboard.
- Customize display attributes such as color, contrast, and font size.

Consult assistive technologies

Assistive technology products such as screen readers function with the user interfaces found in z/OS. Consult the product information for the specific assistive technology product that is used to access z/OS interfaces.

Keyboard navigation of the user interface

You can access z/OS user interfaces with TSO/E or ISPF. The following information describes how to use TSO/E and ISPF, including the use of keyboard shortcuts and function keys (PF keys). Each guide includes the default settings for the PF keys.

- *z/OS TSO/E Primer*
- *z/OS TSO/E User's Guide*
- *z/OS ISPF User's Guide Vol I*

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users who access IBM Documentation with a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line because they are considered a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that the screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1)

are mutually exclusive alternatives. If you hear the lines `3.1 USERID` and `3.1 SYSTEMID`, your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol is placed next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format `3 \* FILE`. Format `3* FILE` indicates that syntax element FILE repeats. Format `3* \* FILE` indicates that syntax element `* FILE` repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol to provide information about the syntax elements. For example, the lines `5.1*, 5.1 LASTRUN, and 5.1 DELETE` mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, it indicates a reference that is defined elsewhere. The string that follows the % symbol is the name of a syntax fragment rather than a literal. For example, the line `2.1 %OP1` means that you must refer to separate syntax fragment OP1.

The following symbols are used next to the dotted decimal numbers.

? indicates an optional syntax element
The question mark (?) symbol indicates an optional syntax element. A dotted decimal number followed by the question mark symbol (?) indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example `5? NOTIFY`). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines `5 ?, 5 NOTIFY`, and `5 UPDATE`, you know that the syntax elements NOTIFY and UPDATE are optional. That is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

! indicates a default syntax element
The exclamation mark (!) symbol indicates a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the dotted decimal number can specify the ! symbol. For example, if you hear the lines `2? FILE, 2.1! (KEEP), and 2.1 (DELETE)`, you know that (KEEP) is the default option for the FILE keyword. In the example, if you include the FILE keyword, but do not specify an option, the default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, the default `FILE(KEEP)` is used. However, if you hear the lines `2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE)`, the default option KEEP applies only to the next higher dotted decimal number, `2.1` (which does not have an associated keyword), and does not apply to `2? FILE`. Nothing is used if the keyword FILE is omitted.

* indicates an optional syntax element that is repeatable
The asterisk or glyph (*) symbol indicates a syntax element that can be repeated zero or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line `5.1* data area`, you know that you can include one data area, more than one data area, or no data area. If you hear the lines `3*, 3 HOST, 3 STATE`, you know that you can include HOST, STATE, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you can write HOST STATE, but you cannot write HOST HOST.
3. The * symbol is equivalent to a loopback line in a railroad syntax diagram.

+ indicates a syntax element that must be included

The plus (+) symbol indicates a syntax element that must be included at least once. A dotted decimal number followed by the + symbol indicates that the syntax element must be included one or more times. That is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loopback line in a railroad syntax diagram.

Notices

This information was developed for products and services that are offered in the USA or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for IBM Documentation. Use of hyperlinks in other output formats of this information is at your own risk.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Site Counsel
2455 South Road

Terms and conditions for product documentation

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at ibm.com®/privacy and IBM's Online Privacy Statement at ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at ibm.com/software/info/product-privacy.

Policy for unsupported hardware

Various z/OS elements, such as DFSMSdfp, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those

products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: IBM Lifecycle Support for z/OS (www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

Programming interface information

This publication documents information that is NOT intended to be used as Programming Interfaces of z/OS XL C/C++.

This publication documents *intended* Programming Interfaces that allow the customer to write z/OS XL C/C++ programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and Trademark information (www.ibm.com/legal/copytrade.shtml).

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Standards

The following standards are supported in combination with the Language Environment element:

- The C language is consistent with *Programming languages - C (ISO/IEC 9899:1999)* and a subset of *Programming languages - C (ISO/IEC 9899:2011)*. For more information, see International Organization for Standardization (ISO) (www.iso.org).
- The C++ language is consistent with *Programming languages - C++ (ISO/IEC 14882:1998)*, *Programming languages - C++ (ISO/IEC 14882:2003(E))*, and a subset of *Programming languages - C++ (ISO/IEC 14882:2011)*.

The following standards are supported in combination with the Language Environment and z/OS UNIX System Services elements:

- A subset of *IEEE Std. 1003.1-2001 (Single UNIX Specification, Version 3).* For more information, see IEEE (www.ieee.org).
- *IEEE Std 1003.1—1990, IEEE Standard Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C language]*, copyright 1990 by the Institute of Electrical and Electronic Engineers, Inc.
- The core features of *IEEE P1003.1a Draft 6 July 1991, Draft Revision to Information Technology—Portable Operating System Interface (POSIX), Part 1: System Application Program Interface (API) [C Language]*, copyright 1992 by the Institute of Electrical and Electronic Engineers, Inc.

- *IEEE Std 1003.2—1992, IEEE Standard Information Technology—Portable Operating System Interface (POSIX)—Part 2: Shells and Utilities*, copyright 1990 by the Institute of Electrical and Electronic Engineers, Inc.
- The core features of *IEEE Std P1003.4a/D6—1992, IEEE Draft Standard Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)—Amendment 2: Threads Extension [C language]*, copyright 1990 by the Institute of Electrical and Electronic Engineers, Inc.
- The core features of *IEEE 754-1985 (R1990) IEEE Standard for Binary Floating-Point Arithmetic (ANSI)*, copyright 1985 by the Institute of Electrical and Electronic Engineers, Inc.
- *X/Open CAE Specification, System Interfaces and Headers, Issue 4 Version 2*, copyright 1994 by The Open Group
- *X/Open CAE Specification, Networking Services, Issue 4*, copyright 1994 by The Open Group
- *X/Open Specification Programming Languages, Issue 3, Common Usage C*, copyright 1988, 1989, and 1992 by The Open Group
- United States Government's *Federal Information Processing Standard (FIPS) publication for the programming language C, FIPS-160*, issued by National Institute of Standards and Technology, 1991

Bibliography

This bibliography lists the publications for IBM products that are related to z/OS XL C/C++. It includes publications covering the application programming task. The bibliography is not a comprehensive list of the publications for these products, however, it should be adequate for most z/OS XL C/C++ users. Refer to *z/OS Information Roadmap* for a complete list of publications belonging to the z/OS product.

z/OS

- *z/OS Introduction and Release Guide*
- *z/OS Planning for Installation*
- *z/OS Release Upgrade Reference Summary*
- *z/OS Information Roadmap*
- *z/OS Licensed Program Specifications*
- *z/OS Upgrade Workflow*
- *z/OS Program Directory*

z/OS XL C/C++

- *z/OS XL C/C++ Programming Guide*
- *z/OS XL C/C++ User's Guide*
- *z/OS XL C/C++ Language Reference*
- *z/OS XL C/C++ Messages*
- *z/OS XL C/C++ Runtime Library Reference*
- *z/OS C Curses*
- *z/OS XL C/C++ Compiler and Runtime Migration Guide for the Application Programmer*
- *Standard C++ Library Reference*

z/OS Metal C Runtime Library

- *z/OS Metal C Programming Guide and Reference*

z/OS Runtime Library Extensions

- *z/OS Common Debug Architecture User's Guide*
- *z/OS Common Debug Architecture Library Reference*
- *DWARF/ELF Extensions Library Reference*

Debug Tool

- Debug Tool documentation, which is available at Debug Tool Utilities and Advanced Functions (www.ibm.com/software/awdtools/debugtool).

z/OS Language Environment

- *z/OS Language Environment Concepts Guide*
- *z/OS Language Environment Customization*
- *z/OS Language Environment Debugging Guide*
- *z/OS Language Environment Programming Guide*

- *z/OS Language Environment Programming Reference*
- *z/OS Language Environment Runtime Application Migration Guide*
- *z/OS Language Environment Writing Interlanguage Communication Applications*
- *z/OS Language Environment Runtime Messages*

Assembler

Assembler documentation, which is available at High Level Assembler and Toolkit Feature in IBM Documentation (www.ibm.com/docs/en/hla-and-tf/1.6).

COBOL

- COBOL documentation, which is available at the Enterprise COBOL for z/OS documentation library (www.ibm.com/support/docview.wss?uid=swg27036733).

PL/I

- PL/I documentation, which is available at the IBM Enterprise PL/I for z/OS library (www.ibm.com/support/docview.wss?uid=swg27036735).

VS FORTRAN

- VS FORTRAN documentation, which is available at the VS FORTRAN Library (www.ibm.com/software/awdtools/fortran/vsfortran/library.html).

CICS Transaction Server for z/OS

- CICS Transaction Server for z/OS documentation, which is available at CICS Transaction Server for z/OS (www.ibm.com/docs/en/cics-ts)

DB2

- DB2 for z/OS documentation, which is available at Db2 for z/OS in IBM Documentation (www.ibm.com/docs/en/db2-for-zos).

IMS/ESA®

- IMS documentation, which is available at IMS in IBM Documentation (www.ibm.com/docs/en/ims).

MVS

- *z/OS MVS Program Management: User's Guide and Reference*
- *z/OS MVS Program Management: Advanced Facilities*

QMF

- QMF documentation, which is available at the DB2 Query Management Facility Library (www.ibm.com/support/docview.wss?uid=swg27021603).

DFSMS

- *z/OS DFSMS Introduction*
- *z/OS DFSMS Managing Catalogs*
- *z/OS DFSMS Using Data Sets*
- *z/OS DFSMS Macro Instructions for Data Sets*

- *z/OS DFSMS Access Method Services Commands*

Index

CCN3425 86
CCN3426 86
CCN3427 86
CCN3428 86
CCN3429 86
CCN3430 86
CCN3431 87
CCN3432 87
CCN3433 87
CCN3434 87
CCN3435 87
CCN3436 87
CCN3437 87
CCN3438 87
CCN3439 88
CCN3440 88
CCN3441 88
CCN3442 88
CCN3443 88
CCN3444 88
CCN3445 88
CCN3446 88
CCN3447 89
CCN3448 89
CCN3449 89
CCN3450 89
CCN3451 89
CCN3452 89
CCN3453 89
CCN3454 89
CCN3455 89
CCN3456 90
CCN3457 90
CCN3458 90
CCN3459 90
CCN3460 90
CCN3461 90
CCN3462 90
CCN3463 90
CCN3464 90
CCN3465 91
CCN3466 91
CCN3467 91
CCN3468 91
CCN3469 91
CCN3470 91
CCN3471 91
CCN3472 91
CCN3473 91
CCN3474 92
CCN3475 92
CCN3476 92
CCN3477 92
CCN3478 92
CCN3479 92
CCN3480 92
CCN3481 92
CCN3482 92
CCN3483 92
CCN3484 93
CCN3485 93
CCN3486 93
CCN3487 93
CCN3489 93

CCN3490 93
CCN3491 93
CCN3492 94
CCN3493 94
CCN3494 94
CCN3495 94
CCN3496 94
CCN3497 94
CCN3498 94
CCN3499 94
CCN3503 94
CCN3505 95
CCN3508 95
CCN3509 95
CCN3512 95
CCN3513 95
CCN3514 95
CCN3515 95
CCN3517 95
CCN3518 96
CCN3519 96
CCN3520 96
CCN3521 96
CCN3522 96
CCN3524 96
CCN3531 96
CCN3545 96
CCN3546 97
CCN3547 97
CCN3548 97
CCN3549 97
CCN3550 97
CCN3551 97
CCN3552 97
CCN3553 97
CCN3554 97
CCN3555 98
CCN3556 98
CCN3557 98
CCN3558 98
CCN3559 98
CCN3560 98
CCN3561 98
CCN3562 99
CCN3563 99
CCN3564 99
CCN3565 99
CCN3566 99
CCN3567 99
CCN3568 99
CCN3569 99
CCN3570 100
CCN3571 100
CCN3572 100
CCN3573 100
CCN3574 100
CCN3575 100
CCN3576 100
CCN3578 101
CCN3585 101
CCN3586 101
CCN3600 101
CCN3610 101
CCN3671 101

CCN6149 217
CCN6150 217
CCN6151 217
CCN6152 218
CCN6153 218
CCN6154 218
CCN6155 218
CCN6156 218
CCN6157 218
CCN6158 218
CCN6159 218
CCN6160 218
CCN6161 219
CCN6162 219
CCN6163 219
CCN6164 219
CCN6165 219
CCN6166 219
CCN6167 220
CCN6168 220
CCN6169 220
CCN6170 220
CCN6171 220
CCN6172 220
CCN6173 220
CCN6174 220
CCN6175 221
CCN6176 221
CCN6177 221
CCN6178 221
CCN6179 221
CCN6180 221
CCN6181 221
CCN6182 221
CCN6183 222
CCN6184 222
CCN6185 222
CCN6186 222
CCN6187 222
CCN6188 222
CCN6189 222
CCN6190 223
CCN6191 223
CCN6192 223
CCN6193 223
CCN6194 223
CCN6195 223
CCN6196 223
CCN6197 223
CCN6198 223
CCN6199 224
CCN6200 224
CCN6201 224
CCN6202 224
CCN6203 224
CCN6204 224
CCN6205 224
CCN6206 224
CCN6207 225
CCN6208 225
CCN6209 225
CCN6210 225
CCN6211 225
CCN6212 225

CCN6213 225
CCN6214 225
CCN6215 226
CCN6216 226
CCN6217 226
CCN6218 226
CCN6219 226
CCN6220 226
CCN6221 226
CCN6222 227
CCN6223 227
CCN6224 227
CCN6225 227
CCN6226 227
CCN6227 227
CCN6228 227
CCN6229 227
CCN6230 228
CCN6231 228
CCN6232 228
CCN6233 228
CCN6234 228
CCN6235 228
CCN6236 228
CCN6237 229
CCN6238 229
CCN6239 229
CCN6240 229
CCN6255 229
CCN6257 229
CCN6258 229
CCN6259 229
CCN6260 230
CCN6261 230
CCN6262 230
CCN6263 230
CCN6264 230
CCN6265 230
CCN6266 230
CCN6267 230
CCN6268 231
CCN6269 231
CCN6270 231
CCN6271 231
CCN6272 231
CCN6273 231
CCN6274 231
CCN6275 232
CCN6276 232
CCN6277 232
CCN6278 232
CCN6279 232
CCN6280 232
CCN6281 232
CCN6282 233
CCN6283 233
CCN6284 233
CCN6285 233
CCN6286 233
CCN6287 233
CCN6288 233
CCN6289 233
CCN6290 234
CCN6291 234

IBM.

Product Number: 5650-ZOS

GC14-7305-50