IGZ0002S

Contents

IGZ0002S *debugging-information*

Explanation: A SYNAD error has occurred on a QSAM file. The text was supplied by the system SYNADAF routine. Since the debugging information supplied in this message is system specific, the message format differs between CMS and MVS environments. The message issued under MVS consists of the following:

```
IGZ0002S job name, step name, unit address, device
         type, ddname, operation attempted, error
         description, actual track address and
         block number, access method.
```

The message issued under CMS is as follows:

```
IGZ0002S 120S operation type ERROR nnn ON ddname,
```

Definitions requiring further explanation for the above message formats are:

120S	is the CMS message number for SYNAD errors
operation type	INPUT or OUTPUT
device type	UR for unit record device
TA	Magnetic tape device
DA	Direct access device
nnn	is the associated error code
ddname	is the DDNAME of the related file
operation attempted	actual operation

System action: The application was terminated.

Programmer response: For more information regarding the CMS message number 120S and related error codes, see *z/VM CMS and REXX/VM Messages and Codes* or *z/VM CP Messages and Codes*. For information on the MVS text of this SYNADAF message, see *z/OS DFSMS Macro Instructions for Data Sets*, and *z/OS DFSMS Using Data Sets*.

Symbolic Feedback Code: IGZ002

IGZ0003W **A logic error occurred for file** *file-name* **in program** *program-name* **at relative location** *relative-location***.**

Explanation: This error is usually caused by an I/O operation request that is not valid for the file—for example, a WRITE into a file opened for INPUT, or a START to a VSAM ESDS.

A file status clause was specified or an error declarative statement was active for the file.

System action: No system action was taken.

Programmer response: Check the operation request and modify the program.

Symbolic Feedback Code: IGZ003

IGZ0005S **OS/VS COBOL programs in the application were found in multiple enclaves.**

Explanation: OS/VS COBOL programs are restricted to one enclave within an application.

System action: The application was terminated.

Programmer response: Modify the application so that the OS/VS COBOL programs appear in one enclave only.

Symbolic Feedback Code: IGZ005

IGZ0006S **The reference to table** *table-name* **by verb number** *verb-number* **on line** *line-number* **addressed an area outside the region of the table.**

Explanation: When the SSRANGE option is in effect, this message is issued to indicate that a fixed-length table has been subscripted in a way that exceeds the defined size of the table, or, for variable-length tables, the maximum size of the table.

The range check was performed on the composite of the subscripts and resulted in an address outside the region of the table. For variable-length tables, the address is outside the region of the table defined when all OCCURS DEPENDING ON objects are at their maximum values; the ODO object's current value is not considered. The check was not performed on individual subscripts.

System action: The application was terminated.

Programmer response: Ensure that the value of literal subscripts and/or the value of variable subscripts as evaluated at run-time do not exceed the subscripted dimensions for subscripted data in the failing statement.

Symbolic Feedback Code: IGZ006

IGZ0007S **The size of variable length group** *group-name* **exceeded the maximum defined length of the group at the time of reference by verb number** *verb-number* **on line** *line-number***.**

Explanation: When the SSRANGE option is in effect, this message is issued to indicate that a variable-length group generated by OCCURS DEPENDING ON has a length that is less than zero, or is greater than the limits defined in the OCCURS DEPENDING ON clauses. The range check was performed on the composite length of the group, and not on the individual OCCURS DEPENDING ON objects.

System action: The application was terminated.

Programmer response: Ensure that OCCURS DEPENDING ON objects as evaluated at run-time do not exceed the maximum number of occurrences of the dimension for tables within the referenced group item.

Symbolic Feedback Code: IGZ007

IGZ0009C **A delete of module** *module-name* **was unsuccessful.**

Explanation: An attempt to delete a module failed.

System action: The application was terminated.

Programmer response: See your IBM service representative.

Symbolic Feedback Code: IGZ009

IGZ0011C *module-name* **was not a proper module for this system environment.**

Explanation: A library subroutine that is system sensitive is inappropriate for the current system environment. For example, an OS environment specific module has been loaded under CICS. The likely causes are:

- Improper concatenation sequence of partitioned data sets that contain the subroutine library, either during run-time or during link-edit of the COBPAC.
- An attempt to use a function unsupported on the current system (for example, ACCEPT on CICS).

System action: The application was terminated.

Programmer response: Check for the conditions stated above, and modify the environment or the application as needed.

Symbolic Feedback Code: IGZ00B

IGZ0012S **There was an invalid attempt to end a sort or merge.**

Explanation: A sort or merge initiated by a COBOL program was in progress and one of the following was attempted:

1. A STOP RUN was issued.
2. A GOBACK or an EXIT PROGRAM was issued within the input procedure or the output procedure of the COBOL program that initiated the sort or merge. Note that the GOBACK and EXIT PROGRAM statements are allowed in a program called by an input procedure or an output procedure.
3. A user handler associated with the program that initiated the sort or merge moved the condition handler resume cursor and resumed the application.

System action: The application was terminated.

Programmer response: Change the application so that it does not use one of the above methods to end the sort or merge.

Symbolic Feedback Code: IGZ00C

IGZ0013S **An error return code** *return-code* **came from a CICS command** *CICS-command* **issued by library subroutine** *library-subroutine***.**

Explanation: An error was encountered when a run-time routine issued a CICS command. The error return code is from the field EIBRESP in the CICS EIB. For more information about the values for the field EIBRESP, see the *CICS/ESA Application Programmer's Reference* , SC33-0676.

System action: The application was terminated.

Programmer response: Modify your application as required.

Symbolic Feedback Code: IGZ00D

IGZ0014W *module-name* **is no longer supported. Its content was ignored.**

Explanation: This message is issued when the run-time detects that IGZETUN or IGZEOPT is linked with the application. IGZETUN and IGZEOPT are ignored when running with Language Environment. CEEUOPT may be used in place of IGZETUN and IGZEOPT.

System action: No system action was taken.

Programmer response: Remove the explicit INCLUDE of IGZEOPT or IGZETUN during the link-edit step.

Symbolic Feedback Code: IGZ00E

IGZ0015S **A recursive call was attempted to a program that was already active. The program name is** *program-name***.**

Explanation: An illegal recursive entry to an active program is detected. For example, Program A has CALLed Program B, and Program B is CALLing Program A.

System action: The application was terminated.

Programmer response: Remove the recursive call to *program-name* or specify the IS RECURSIVE phrase on the PROGRAM-ID statement for the recursively CALLed program. Additionally, if the recursive program is called dynamically, link-edit it with REUS.

Symbolic Feedback Code: IGZ00F

IGZ0016W **Program** *program-name* **could not be deactivated by non-return exit of a routine. Subsequent reentry is not supported.**

Explanation: A COBOL program cannot normally be recursively entered. When non-return style procedure collapse processing is being performed for a COBOL program, an attempt is made to reset the program to a state where it can be recursively entered. This is not supported for certain combinations of function used within the program. After this message is issued, any attempt to reenter the program will result in message IGZ0015S and termination of the enclave.

System action: No system action was taken.

Programmer response: Do not reenter the program or modify the program to allow it to be successfully reset.

Symbolic Feedback Code: IGZ00G

IGZ0017S **The open of DISPLAY or ACCEPT file with environment name** *environment-name* **was unsuccessful.**

Explanation: An error occurred while opening the DISPLAY/ACCEPT file.

System action: The application was terminated.

Programmer response: Check to make sure a ddname has been defined for the file.

Symbolic Feedback Code: IGZ00H

IGZ0018S **On CICS, an attempt was made to run a COBOL program which is not reentrant. The program name is** *program-name***.**

Explanation: COBOL programs running on CICS must be reentrant.

System action: The application was terminated.

Programmer response: In order to make a COBOL program reentrant, compile the COBOL program with the RENT compile-time option.

Symbolic Feedback Code: IGZ00I

IGZ0019W **A FUNCTION result used as a DELIMITED BY operand is larger than the UNSTRING object in program** *program-name* **at displacement** *displacement***. The DELIMITED BY phrase is ignored.**

Explanation: A FUNCTION used as a DELIMITED BY operand was larger than the UNSTRING object.

System action: No system action was taken.

Programmer response: Check the FUNCTION arguments to ensure that they are not larger than the UNSTRING object.

Symbolic Feedback Code: IGZ00J

IGZ0020S **A logic error occurred. Neither FILE STATUS nor a declarative was specified for file** *file-name* **in program** *program-name* **at relative location** *relative-location***. The status code was** *status-code***.**

Explanation: This error is an I/O error, usually caused by an operation request that is not valid for the file, for example, a WRITE into a file opened for INPUT, or a START to a VSAM ESDS. No file status clause was specified, and no error declarative was in effect.

System action: The application was terminated.

Programmer response: Check operation request for the file.

Symbolic Feedback Code: IGZ00K

IGZ0021C *macro-name* **was unsuccessful for file** *file-name***.**

Explanation: The execution of an ENDREQ, GENCB, MODCB, SHOWCB, or TESTCB macro failed. This is the result of system or VSAM problems.

System action: The application was terminated.

Programmer response: See your IBM service representative.

Symbolic Feedback Code: IGZ00L

IGZ0022W **File** *file-name* **in program** *program-name* **will return the maximum record length when read.**

Explanation: A VSAM RRDS with a varying record length has been opened for input. The maximum record length will be returned.

System action: No system action was taken.

Programmer response: None

Symbolic Feedback Code: IGZ00M

IGZ0023S **The dynamic allocation of file** *file-name* **was unsuccessful. The return code was** *return-code***. The reason code was** *reason-code***.**

Explanation: An attempt to dynamically allocate a file using DYNALLOC failed, resulting in the indicated return and reason codes.

System action: The application was terminated.

Programmer response: Review the job stream or filedef to see if any DDNAMES are missing or misspelled. If you can not find any errors, resubmit the job with the CBLQDA(OFF) run-time option and check for any access method messages.

Symbolic Feedback Code: IGZ00N

IGZ0024S An invalid separate sign character was detected in *program-name* **at displacement** *displacement***.**

Explanation: An operation was attempted on data defined with a separate sign. The value in the sign position was not a plus (+) or a minus (-).

System action: The application was terminated.

Programmer response: This error might have occurred because of a REDEFINES clause involving the sign position or a group move involving the sign position, or the position was never initialized. Check for these cases. The compiler formatting option TEST(), or equivalent, along with the ABTERMENC() run-time option, can be used to generate a formatted dump of the user data. This dump can then be used to identify the unacceptable data item contents.

Symbolic Feedback Code: IGZ00O

IGZ0026W The SORT-RETURN special register was never referenced, but the current content indicated the sort or merge operation in program *program-name* **on line number** *line-number* **was unsuccessful.**

Explanation: The COBOL source does not contain any references to the sort-return register. The compiler generates a test after each sort or merge verb. A nonzero return code has been passed back to the program by Sort/Merge.

System action: No system action was taken.

Programmer response: Determine why the Sort/Merge was unsuccessful and fix the problem. Possible reasons why the Sort/Merge was unsuccessful include:

- There was an error detected by DFSORT. See the DFSORT messages for the reason for the error.
- The SORT-RETURN special register was set to a non-zero value by the application program while in an input procedure or an output procedure.

Symbolic Feedback Code: IGZ00Q

IGZ0027W The sort control file could not be opened.

Explanation: An attempt to open the sort control file has failed. Possible reasons for the open failure include:

- A ddname for the sort control file was not provided.
- The IGZSRTCD ddname was provided, but the file associated with the ddname could not be found.

When the sort control file cannot be opened, user-supplied sort control cards will not be passed to Sort/Merge.

The sort control file is optional. On MVS, if you did not provide a ddname for the sort control file (the sort control file name is IGZSRTCD unless it is overridden by changing the value of the SORT-CONTROL special register) you will also get this message: IEC130I 'IGZSRTCD DD STATEMENT MISSING'. This message is informational only.

System action: No system action was taken.

Programmer response: If you want to pass in sort control cards from the sort control file, verify that the ddname is specified and the file is available.

Symbolic Feedback Code: IGZ00R

IGZ0028S An I/O error occurred in sort control file *file-name***.**

Explanation: An I/O error was encountered while trying to read the sort control file. Some or all of the user-supplied sort control cards will not be passed to Sort/Merge.

System action: The application was terminated.

Programmer response: For more information, look at the previous system message you received relating to this I/O error.

Symbolic Feedback Code: IGZ00S

IGZ0029S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than zero.**

Explanation: An illegal value for argument-1 was used.

System action: The application was terminated.

Programmer response: Ensure that argument-1 is greater than or equal to zero.

Symbolic Feedback Code: IGZ00T

IGZ0030S **Argument-2 for function** *function-name* **in program** *program* **at line** *line-number* **was not a positive integer.**

Explanation: An illegal value for argument-2 was used.

System action: The application was terminated.

Programmer response: Ensure that argument-2 is a positive integer.

Symbolic Feedback Code: IGZ00U

IGZ0031S **A restart was not possible since the checkpoint record** *record-name* **was taken while a sort or merge was in progress.**

Explanation: An attempt was made to use the restart facility of checkpoint/restart to resume execution of a job from a checkpoint taken by a COBOL program because of a rerun clause during a Sort/Merge operation. Only checkpoints taken by the sort product can be used to restart from a point within the Sort/Merge operation. The checkpoint record cannot be used for restart.

System action: The application was terminated.

Programmer response: Use a different checkpoint record. If no other checkpoint records exist, the job cannot be restarted.

Symbolic Feedback Code: IGZ00V

IGZ0032S **A CANCEL was attempted on active program** *program-name***.**

Explanation: An attempt was made to cancel an active program. For example, program A called program B; program B is trying to cancel program A.

System action: The application was terminated.

Programmer response: Remove the failing CANCEL statement. In order to locate the failing CANCEL statement, rerun the application with TERMTHDACT(TRACE) or (ABEND). Review the traceback information to identify the program that issued the CANCEL.

Symbolic Feedback Code: IGZ010

IGZ0033S **An attempt was made to pass a parameter address above 16 megabytes to AMODE(24) program** *program-name***.**

Explanation: An attempt was made to pass a parameter located above the 16-megabyte storage line to a program in AMODE(24). The called program will not be able to address the parameter.

System action: The application was terminated.

Programmer response: If the calling program is compiled with the RENT option, the DATA(24) option may be used in the calling program to make sure that its data is located in storage accessible to an AMODE(24) program. If the calling program is compiled with the NORENT option, the RMODE(24) option may be used in the calling program to make sure that its data is located in storage accessible to an AMODE(24) program. Verify that no linkedit, binder or genmod overrides are responsible for this error.

Symbolic Feedback Code: IGZ011

IGZ0034W The file with system-name *system-name* could not be extended. Secondary extents were not specified or were not available. The last WRITE was at offset *offset* in program *program-name*.

Explanation: There is insufficient space available for an output file. There is no invalid key clause, file status, or user error declarative. This corresponds to the MVS X37 ABEND.

System action: No system action was taken.

Programmer response: Check the file attributes and if necessary, reallocate the file. Also check data set allocations.

Symbolic Feedback Code: IGZ012

IGZ0035S There was an unsuccessful OPEN or CLOSE of file *file-name* in program *program-name* at relative location *location*. Neither FILE STATUS nor an ERROR declarative were specified. The status code was *status-code*.

Explanation: An error has occurred while opening or closing the named file. No file status or user error declarative was specified.

System action: The application was terminated.

Programmer response: Check to make sure there is a ddname defined for the indicated file.

Symbolic Feedback Code: IGZ013

IGZ0036W Truncation of high order digit positions occurred in program *program-name* on line number *line-number*.

Explanation: The generated code has truncated an intermediate result (that is, temporary storage used during an arithmetic calculation) to 30 digits; some of the truncated digits were not 0.

System action: No system action was taken.

Programmer response: See *COBOL for OS/390 & VM Programming Guide* or *COBOL for MVS & VM Programming Guide* for a description of intermediate results.

Symbolic Feedback Code: IGZ014

IGZ0037S The flow of control in program *program-name* proceeded beyond the last line of the program.

Explanation: The program did not have a terminator (STOP, GOBACK, or EXIT), and control fell through the last instruction.

System action: The application was terminated.

Programmer response: Check the logic of the program. Sometimes this error occurs because of one of the following logic errors:

- The last paragraph in the program was only supposed to receive control as the result of a PERFORM statement, but due to a logic error it was branched to by a GO TO statement.
- The last paragraph in the program was executed as the result of a "fall-through" path, and there was no statement at the end of the paragraph to end the program.

Symbolic Feedback Code: IGZ015

IGZ0038S A reference modification length value of *reference-modification-value* on line *line-number* which was not equal to 1 was found in a reference to data item *data-item* which was passed by value.

Explanation: The length value in a reference modification specification was not equal to 1. The length value must be equal to 1.

System action: The application was terminated.

Programmer response: Check the indicated line number in the program to ensure that any reference modified length values are (or will resolve to) 1.

Symbolic Feedback Code: IGZ016

IGZ0039S An invalid overpunched sign was detected in program *program-name* **on line** *line-number***.**

Explanation: An operation was attempted on data defined with an overpunched sign. The value in the sign was not valid.

System action: The application was terminated.

Programmer response: This error might have occurred because of a REDEFINES clause involving the sign position or a group move involving the sign position, or the position was never initialized. Check for the above cases.

Symbolic Feedback Code: IGZ017

IGZ0040S An invalid separate sign was detected in program *program-name* **on line** *line-number***.**

Explanation: An operation was attempted on data defined with a separate sign. The value in the sign position was not a plus (+) or a minus (-).

System action: The application was terminated.

Programmer response: This error might have occurred because of a REDEFINES clause involving the sign position or a group move involving the sign position, or the position was never initialized. Check for the above cases.

Symbolic Feedback Code: IGZ018

IGZ0041W The warning message limit was exceeded. Further warning messages were suppressed.

Explanation: The limit on warning messages is 256. This constraint on the number of warning messages prevents a looping program from flooding the system buffers.

System action: No system action was taken.

Programmer response: Correct the situations causing the warning messages or correct the looping problem.

Symbolic Feedback Code: IGZ019

IGZ0042C There was an attempt to use the IGZBRDGE macro, but the calling program was not COBOL.

Explanation: A non-COBOL program attempted to call a COBOL program using the IGZBRDGE interface. COBOL/370 could not find a COBOL environment.

System action: The application was terminated.

Programmer response: Do not call an entry point specified via the IGZBRDGE macro from a non-COBOL program.

Symbolic Feedback Code: IGZ01A

IGZ0044S There was an attempt to call the COBOL main program *program-name* **that was not in initial state.**

Explanation: You will receive this message if you attempt to enter a NONREENTRANT COBOL/370, VS COBOL II, COBOL for MVS & VM, or COBOL for OS/390 & VM main program more than once. This is a nonstandard entry attempt.

System action: The application was terminated.

Programmer response: Modify the application so that the non-reentrant COBOL main program won't be called more than once.

Symbolic Feedback Code: IGZ01C

IGZ0045S Unable to invoke method *method-name* **on line number** *line number* **in COBOL program** *program-name***.**

Explanation: The specific method is not supported for the class of the current object reference.

System action: The application was terminated.

Programmer response: Check the indicated line number in the program to ensure that the class of the current object reference supports the method being invoked.

Symbolic Feedback Code: IGZ01D

IGZ0046W **The value specified in the program for the** *special-register* **special register was overridden by the corresponding value in the sort control file.**

Explanation: A nondefault value for the SORT special register specified in the message was used in a program, but a value in the SORT control file which corresponds to that SORT special register was found. The value in the SORT control file was used, and the value in the SORT special register was ignored.

System action: No system action was taken.

Programmer response: See *COBOL for OS/390 & VM Programming Guide* or *COBOL for MVS & VM Programming Guide* for a description of SORT special registers and the SORT control file.

Symbolic Feedback Code: IGZ01E

IGZ0047S **Unable to invoke method** *method-name* **on line number** *line number* **in COBOL class** *class-name***.**

Explanation: The specific method is not supported for the class of the current object reference.

System action: The application was terminated.

Programmer response: Check the indicated line number in the class to ensure that the class of the current object reference supports the method being invoked.

Symbolic Feedback Code: IGZ01F

IGZ0048W **A negative base was raised to a fractional power in an exponentiation expression in program** *program-name* **at displacement** *displacement***. The absolute value of the base was used.**

Explanation: A negative number raised to a fractional power occurred in a library routine. The value of a negative number raised to a fractional power is undefined in COBOL. If a SIZE ERROR clause had appeared on the statement in question, the SIZE ERROR imperative would have been used. However, no SIZE ERROR clause was present, so the absolute value of the base was used in the exponentiation.

System action: No system action was taken.

Programmer response: Ensure that the program variables in the failing statement have been set correctly.

Symbolic Feedback Code: IGZ01G

IGZ0049W **A zero base was raised to a zero power in an exponentiation expression in program** *program-name* **at displacement** *displacement***. The result was set to one.**

Explanation: The value of zero raised to the power zero occurred in a library routine. The value of zero raised to the power zero is undefined in COBOL. If a SIZE ERROR clause had appeared on the statement in question, the SIZE ERROR imperative would have been used. However, no SIZE ERROR clause was present, so the value returned was one.

System action: No system action was taken.

Programmer response: Ensure that the program variables in the failing statement have been set correctly.

Symbolic Feedback Code: IGZ01H

IGZ0050S **A zero base was raised to a negative power in an exponentiation expression in program** *program-name* **at displacement** *displacement***.**

Explanation: The value of zero raised to a negative power occurred in a library routine. The value of zero raised to a negative number is not defined. If a SIZE ERROR clause had appeared on the statement in question, the SIZE ERROR imperative would have been used. However, no SIZE ERROR clause was present.

System action: The application was terminated.

Programmer response: Ensure that the program variables in the failing statement have been set correctly.

Symbolic Feedback Code: IGZ01I

IGZ0051S An invalid EBCDIC digit string was detected on conversion to floating point in *program-name* **at displacement** *displacement***.**

Explanation: The input to the conversion routine contained invalid EBCDIC data.

System action: The application was terminated.

Programmer response: Ensure that the program variables in the failing statement have been set correctly.

Symbolic Feedback Code: IGZ01J

IGZ0052C An internal error or invalid parameters were detected in the floating point conversion routine called from *program-name* **at displacement** *displacement***.**

Explanation: None

System action: The application was terminated.

Programmer response: See your IBM service representative.

Symbolic Feedback Code: IGZ01K

IGZ0053S An overflow occurred on conversion to floating point in *program-name* **at displacement** *displacement***.**

Explanation: A number was generated in the program that is too large to be represented in floating point.

System action: The application was terminated.

Programmer response: You need to modify the program appropriately to avoid an overflow.

Symbolic Feedback Code: IGZ01L

IGZ0054W An overflow occurred on conversion from floating point to fixed point in *program-name* **at displacement** *displacement***. The result was truncated.**

Explanation: The result of a conversion to fixed point from floating point contains more digits than will fit in the fixed point receiver. The high order digits were truncated.

System action: No system action was taken.

Programmer response: No action is necessary, although you may want to modify the program to avoid an overflow.

Symbolic Feedback Code: IGZ01M

IGZ0055W An underflow occurred on conversion to floating point in *program-name* **at displacement** *displacement***. The result was set to zero.**

Explanation: On conversion to floating point, the negative exponent exceeded the limit of the hardware. The floating point value was set to zero.

System action: No system action was taken.

Programmer response: No action is necessary, although you may want to modify the program to avoid an underflow.

Symbolic Feedback Code: IGZ01N

IGZ0056W One or more files were not closed by program *program-name* **before program termination.**

Explanation: The specified program has finished but has not closed all of the files it opened. COBOL attempts to clean up storage and closes any open files.

System action: No system action was taken.

Programmer response: Check that all files are closed before the program terminates.

Symbolic Feedback Code: IGZ01O

IGZ0057S There was an attempt to initialize a reusable environment through ILBOSTP0, but either the enclave was not the first enclave or COBOL was not the main program of the already established enclave.

Explanation: A request to establish a reusable environment through ILBOSTP0 can only occur at the beginning of the application. Examples when this error can occur:

- PL/I program calls ASSEMBLE program which calls ILBOSTP0.
- Language Environment enabled ASSEMBLER program calls ILBOSTP0.

System action: The application was terminated.

Programmer response: Only invoke ILBOSTP0 before calling any program within the application that brings up Language Environment.

Symbolic Feedback Code: IGZ01P

IGZ0058S Exponent overflow occurred in program *program-name* **at displacement** *displacement***.**

Explanation: Floating point exponent overflow occurred in a library routine.

System action: The application was terminated.

Programmer response: Ensure that the program variables in the failing statement have been set correctly.

Symbolic Feedback Code: IGZ01Q

IGZ0059W An exponent with more than nine digits was truncated in program *program-name* **at displacement** *displacement***.**

Explanation: Exponents in fixed point exponentiations may not contain more than nine digits. The exponent was truncated back to nine digits; some of the truncated digits were not 0.

System action: No system action was taken.

Programmer response: No action is necessary, although you may want to adjust the exponent in the failing statement.

Symbolic Feedback Code: IGZ01R

IGZ0060W Truncation of high order digit positions occurred in program *program-name* **at displacement** *displacement***.**

Explanation: Code in a library routine has truncated an intermediate result (that is, temporary storage used during an arithmetic calculation) back to 30 digits; some of the truncated digits were not 0.

System action: No system action was taken.

Programmer response: See *COBOL for OS/390 & VM Programming Guide* or *COBOL for MVS & VM Programming Guide* for a description of intermediate results.

Symbolic Feedback Code: IGZ01S

IGZ0061S Division by zero occurred in program *program-name* **at displacement** *displacement***.**

Explanation: Division by zero occurred in a library routine. Division by zero is not defined. If a SIZE ERROR clause had appeared on the statement in question, the SIZE ERROR imperative would have been used. However, no SIZE ERROR clause was present.

System action: The application was terminated.

Programmer response: Ensure that the program variables in the failing statement have been set correctly.

Symbolic Feedback Code: IGZ01T

IGZ0063S An invalid sign was detected in a numeric edited sending field in ***program-name*** **on line number** ***line-number*****.**

Explanation: An attempt has been made to move a signed numeric edited field to a signed numeric or numeric edited receiving field in a MOVE statement. However, the sign position in the sending field contained a character that was not a valid sign character for the corresponding PICTURE.

System action: The application was terminated.

Programmer response: Ensure that the program variables in the failing statement have been set correctly.

Symbolic Feedback Code: IGZ01V

IGZ0064S A recursive call to active program ***program-name*** **in compilation unit** ***compilation-unit*** **was attempted.**

Explanation: COBOL does not allow reinvocation of an internal program which has begun execution, but has not yet terminated. For example, if internal programs A and B are siblings of a containing program, and A calls B and B calls A, this message will be issued.

System action: The application was terminated.

Programmer response: Examine your program to eliminate calls to active internal programs.

Symbolic Feedback Code: IGZ020

IGZ0065S A CANCEL of active program ***program-name*** **in compilation unit** ***compilation-unit*** **was attempted.**

Explanation: An attempt was made to cancel an active internal program. For example, if internal programs A and B are siblings in a containing program and A calls B and B cancels A, this message will be issued.

System action: The application was terminated.

Programmer response: Examine your program to eliminate cancellation of active internal programs.

Symbolic Feedback Code: IGZ021

IGZ0066S The length of external data record ***data-record*** **in program** ***program-name*** **did not match the existing length of the record.**

Explanation: While processing External data records during program initialization, it was determined that an External data record was previously defined in another program in the run-unit, and the length of the record as specified in the current program was not the same as the previously defined length.

System action: The application was terminated.

Programmer response: Examine the current file and ensure the External data records are specified correctly.

Symbolic Feedback Code: IGZ022

IGZ0067S The NOEQUALS keyword in the sort control file ***file-name*** **conflicted with the specifications of the DUPLICATES phrase on the SORT statement.**

Explanation: A sort control file with an OPTION card specifying the NOEQUALS keyword was used for a SORT which had the DUPLICATES IN ORDER phrase specified. The NOEQUALS keyword and the DUPLICATES phrase conflict.

System action: The application was terminated.

Programmer response: Either remove the NOEQUALS keyword from the sort control file or remove the DUPLICATES IN ORDER phrase from the SORT statement.

Symbolic Feedback Code: IGZ023

IGZ0068W **Duplicate characters were ignored in an INSPECT CONVERTING statement in program** *program-name* **at displacement** *displacement***.**

Explanation: The same character appeared more than once in the identifier that contained the characters to be converted in an INSPECT CONVERTING statement. The first occurrence of the character, and the corresponding character in the replacement field, are used, and subsequent occurrences are not used.

System action: No system action was taken.

Programmer response: Duplicate characters in the indicated INSPECT statement may be deleted; programmer action is not required.

Symbolic Feedback Code: IGZ024

IGZ0069S **On VM, file** *file-name* **in program** *program-name* **attempted to use VSAM in XA or ESA mode. Using VSAM while in XA or ESA mode is not supported under the installed level of VM. The program was terminated.**

Explanation: VSAM can only operate in S/370 mode virtual machines on VM/SP XA and VM/ESA Release 1 ESA feature. The job was cancelled. Only on VM/ESA Release 1.1 (CMS8), and higher releases, can VSAM and VS COBOL II be used in XA-mode and XC-mode virtual machines.

System action: The application was terminated.

Programmer response: See your systems programmer for assistance.

Symbolic Feedback Code: IGZ025

IGZ0070S **The FILEDEF command "FILEDEF** *ddname* **DISK FILE** *ddname* **A4" was unsuccessful.**

Explanation: An attempt at dynamic allocation for CMS file *ddname* using the FILEDEF command has failed.

System action: The application was terminated.

Programmer response: See your systems programmer for assistance.

Symbolic Feedback Code: IGZ026

IGZ0071S **ALL subscripted table reference to table** *table-name* **by verb number** *verb-number* **on line** *line-number* **had an ALL subscript specified for an OCCURS DEPENDING ON dimension, and the object was less than or equal to 0.**

Explanation: When the SSRANGE option is in effect, this message is issued to indicate that there are 0 occurrences of dimension subscripted by ALL.

The check is performed against the current value of the OCCURS DEPENDING ON OBJECT.

System action: The application was terminated.

Programmer response: Ensure that ODO object(s) of ALL-subscripted dimensions of any subscripted items in the indicated statement are positive.

Symbolic Feedback Code: IGZ027

IGZ0072S **A reference modification start position value of** *reference-modification-value* **on line** *line-number* **referenced an area outside the region of data item** *data-item***.**

Explanation: The value of the starting position in a reference modification specification was less than 1, or was greater than the current length of the data item that was being reference modified. The starting position value must be a positive integer less than or equal to the number of characters in the reference modified data item.

System action: The application was terminated.

Programmer response: Check the value of the starting position in the reference modification specification.

Symbolic Feedback Code: IGZ028

IGZ0073S **A non-positive reference modification length value of** *reference-modification-value* **on line** *line-number* **was found in a reference to data item** *data-item***.**

Explanation: The length value in a reference modification specification was less than or equal to 0. The length value must be a positive integer.

System action: The application was terminated.

Programmer response: Check the indicated line number in the program to ensure that any reference modified length values are (or will resolve to) positive integers.

Symbolic Feedback Code: IGZ029

IGZ0074S **A reference modification start position value of** *reference-modification-value* **and length value of** *length* **on line** *line-number* **caused reference to be made beyond the rightmost character of data item** *data-item***.**

Explanation: The starting position and length value in a reference modification specification combine to address an area beyond the end of the reference modified data item. The sum of the starting position and length value minus one must be less than or equal to the number of characters in the reference modified data item.

System action: The application was terminated.

Programmer response: Check the indicated line number in the program to ensure that any reference modified start and length values are set such that a reference is not made beyond the rightmost character of the data item.

Symbolic Feedback Code: IGZ02A

IGZ0075S **Inconsistencies were found in EXTERNAL file** *file-name* **in program** *program-name***. The following file attributes did not match those of the established external file:** *attribute-1 attribute-2 attribute-3 attribute-4 attribute-5 attribute-6 attribute-7*

Explanation: One or more attributes of an external file did not match between two programs that defined it.

System action: The application was terminated.

Programmer response: Correct the external file. For a summary of file attributes which must match between definitions of the same external file, see *IBM COBOL Language Reference*

Symbolic Feedback Code: IGZ02B

IGZ0076W **The number of characters in the INSPECT REPLACING CHARACTERS BY data-name in program** *program-name* **at displacement** *displacement* **was not equal to one. The first character was used.**

Explanation: A data item which appears in a CHARACTERS phrase within a REPLACING phrase in an INSPECT statement must be defined as being one character in length. Because of a reference modification specification for this data item, the resultant length value was not equal to one. The length value is assumed to be one.

System action: No system action was taken.

Programmer response: You may correct the reference modification specifications in the failing INSPECT statement to ensure that the reference modification length is (or will resolve to) 1; programmer action is not required.

Symbolic Feedback Code: IGZ02C

IGZ0077W **The lengths of the** *data-item* **items in program** *program-name* **at displacement** *displacement* **were not equal. The shorter length was used.**

Explanation: The two data items which appear in a REPLACING or CONVERTING phrase in an INSPECT statement must have equal lengths, except when the second such item is a figurative constant. Because of the reference modification for one or both of these data items, the resultant length values were not equal. The shorter length value is applied to both items, and execution proceeds.

System action: No system action was taken.

Programmer response: You may adjust the operands of unequal length in the failing INSPECT statement; programmer action is not required.

Symbolic Feedback Code: IGZ02D

IGZ0078S **ALL subscripted table reference to table** *table-name* **by verb number** *verb-number* **on line** *line-number* **will exceed the upper bound of the table.**

Explanation: When the SSRANGE option is in effect, this message is issued to indicate that a multi-dimension table with ALL specified as one or more of the subscripts will result in a reference beyond the upper limit of the table.

The range check was performed on the composite of the subscripts and the maximum occurrences for the ALL subscripted dimensions. For variable-length tables the address is outside the region of the table defined when all OCCURS DEPENDING ON objects are at their maximum values; the ODO object's current value is not considered. The check was not performed on individual subscripts.

System action: The application was terminated.

Programmer response: Ensure that OCCURS DEPENDING ON objects as evaluated at run-time do not exceed the maximum number of occurrences of the dimension for table items referenced in the failing statement.

Symbolic Feedback Code: IGZ02E

IGZ0079S **On CICS,** *program-lang* **program** *program-name* **attempted to call OS/VS COBOL program** *program-name***.**

Explanation: On CICS, a COBOL/370, VS COBOL II, COBOL for MVS & VM, or COBOL for OS/390 & VM program attempted to call an OS/VS COBOL program with the CALL statement. Using the CALL statement to perform calls between the following are not not supported on CICS:

- COBOL for OS/390 & VM programs and OS/VS COBOL programs
- COBOL for MVS & VM programs and OS/VS COBOL programs
- COBOL/370 programs and OS/VS COBOL programs
- VS COBOL II programs and OS/VS COBOL programs

System action: The application was terminated.

Programmer response: If you need to invoke an OS/VS COBOL program from a COBOL/370, VS COBOL II, COBOL for MVS & VM, or COBOL for OS/390 & VM program use EXEC CICS LINK.

Symbolic Feedback Code: IGZ02F

IGZ0080S **A dynamic call to** *module-name* **failed because the program entry name** *program-name* **does not match.**

Explanation: If a program compiled with the PGMNAME(LONGUPPER) or the PGMNAME(LONGMIXED) option is dynamically called, the program name must be identical to the name of the module that contains it. If an alternate entry name is called, the entry name must be identical to the ALIAS name representing that entry point. Note that the program entry name can not exceed 8 bytes and must be entirely upper-case.

System action: The application was terminated.

Programmer response: The name of the program failing the dynamic call must be modified to comply with the rules stated above. Otherwise, only static calls to the program are permitted.

Symbolic Feedback Code: IGZ02G

IGZ0096C **A load of module** *module-name* **was unsuccessful.**

Explanation: An attempt to load a module failed. The module was not available or a system load failure occurred.

System action: The application was terminated.

Programmer response: See your systems programmer for assistance.

Symbolic Feedback Code: IGZ030

IGZ0097S **Argument-1 for function** *function-name* **in program** *program-name* **at displacement** *displacement* **contained no digits.**

Explanation: Argument-1 for the indicated function must contain at least 1 digit.

System action: The application was terminated.

Programmer response: Adjust the number of digits in Argument-1 in the failing statement.

Symbolic Feedback Code: IGZ031

IGZ0098C **The message text for message** *message-number* **was inaccessible to IGZCWTO.**

Explanation: The message text module used by IGZCWTO did not contain message text for the indicated message number.

System action: The application was terminated.

Programmer response: See your IBM service representative.

Symbolic Feedback Code: IGZ032

IGZ0099C **Internal error** *error-code* **was detected in module** *module-name***.**

Explanation: An unrecoverable error was detected in run-time module *module-name*.

When the module name in the message is IGZCXCC, the error-code indicates the error as described below:

Error-code	**Description**
1	The COBOL environment is not initialized. The COBOL environment must be initialized before calling IGZCXCC.
2	An invalid function code was passed to IGZCXCC.
3	An invalid name length was passed to IGZCXCC.
4	IGZCXCC detected that a nested enclave should be created.
5	IGZCXCC cannot be called when running on CICS.

When the module name in the message is IGZCLNC, IGZCLNK, or IGZCFCC, the error-code indicates the error as described below:

Error-code	**Description**
9	IGZCXCC is being used and an invalid cancel was attempted.

When the module name in the message is IGZEINI, the error-code indicates the error as described below:

Error-code	**Description**
101	There was an attempt to initialize a VS COBOL II or OS/VS COBOL program as a subprogram before the main program has run.
102	An OS/VS COBOL program is being initialized but the TGT address was not passed.

When the module name in the message is IGZCII1, the error-code indicates the error as described below:

Error-code	**Description**
NOTMAIN1	Subprogram initialization occurred when main program initialization was expected.
MAINCLAS	COBOL class initialization occurred when main program initialization was expected.

INVSTRC1
Invalid threading status.

INVST001
Invalid program initialization state.

INVST002
Invalid program initialization state.

INVST003
Invalid program initialization state.

PGMIIP01
Program initialization occurred when another program was in the process of being initialized.

When the module name in the message is IGZCII2, the error-code indicates the error as described below: :

Error-code
Description

INVSIG01
During class initialization, the initialization in-progress count is negative.

INVSIG02
During program initialization, the initialization in-progress count is not one.

INVSIG03
During program initialization, the initialization in-progress count is not zero.

INVST001
Invalid program initialization state.

System action: The application was terminated.

Programmer response: See your IBM service representative.

Symbolic Feedback Code: IGZ033

IGZ0100S **Argument-1 for function** *function* **in program** *program* **at displacement** *displacement* **was less than or equal to -1.**

Explanation: An illegal value was used for Argument-1.

System action: The application was terminated.

Programmer response: Ensure that argument-1 is greater than -1.

Symbolic Feedback Code: IGZ034

IGZ0108S **The cancel of program** *program-name* **failed because the module load point address was not provided when the program was loaded.**

Explanation: In a Language Environment/370 preinitialized environment users may specify their own load service routine. If this routine fails to provide the module load point address as an output parameter when loading a COBOL program, that program can not be cancelled using COBOL'S CANCEL statement.

System action: The application was terminated.

Programmer response: Modify the user load service to provide the module load point address.

Symbolic Feedback Code: IGZ03C

IGZ0151S **Argument-1 for function** *function-name* **in program** *program-name* **at displacement** *displacement* **contained more than 18 digits.**

Explanation: The total number of digits in argument-1 of the indicated function exceeded 18 digits.

System action: The application was terminated.

Programmer response: Adjust the number of digits in argument-1 in the failing statement.

Symbolic Feedback Code: IGZ04N

IGZ0152S **Invalid character** *character* **was found in column** *column-number* **in argument-1 for function** *function-name* **in program** *program-name* **at displacement** *program-displacement***.**

Explanation: A non-digit character other than a decimal point, comma, space or sign (+,-,CR,DB) was found in argument-1 for NUMVAL/NUMVAL-C function.

System action: The application was terminated.

Programmer response: Correct argument-1 for NUMVAL or NUMVAL-C in the indicated statement.

Symbolic Feedback Code: IGZ04O

IGZ0153S **Program** *program-name* **was compiled with a level of the compiler that requires service to be installed on Language Environment.**

Explanation: An attempt was made to run a program that was compiled with a release or service level of the compiler that requires maintenance to be installed on Language Environment, but that maintenance has not been installed.

System action: The application was terminated.

Programmer response: Contact your system programmer to ensure that the corresponding Language Environment maintenance was installed as was indicated in the recent compiler version or maintenance that was installed.

Symbolic Feedback Code: IGZ04P

IGZ0154S **A procedure pointer was set to nested program** *nested-program-name* **in program** *program-name* **at displacement** *displacement***.**

Explanation: Procedure pointers can not be set to a nested program.

System action: The application was terminated.

Programmer response: Make sure that the procedure program is set to an external program.

Symbolic Feedback Code: IGZ04Q

IGZ0155S **Invalid character** *character* **was found in column** *column-number* **in argument-2 for function** *function-name* **in program** *program-name* **at displacement** *program-displacement***.**

Explanation: Illegal character was found in argument-2 for NUMVAL-C function.

System action: The application was terminated.

Programmer response: Check that the function argument does follow the syntax rules.

Symbolic Feedback Code: IGZ04R

IGZ0156S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than zero or greater than 28.**

Explanation: Input argument to function FACTORIAL is greater than 28 or less than 0.

System action: The application was terminated.

Programmer response: Check that the function argument is only one byte long.

Symbolic Feedback Code: IGZ04S

IGZ0157S **The length of Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was not equal to 1.**

Explanation: The length of input argument to ORD function is not 1.

System action: The application was terminated.

Programmer response: Check that the function argument is only one byte long.

Symbolic Feedback Code: IGZ04T

IGZ0158S **The length of Argument-1 for function** *function-name* **in program** *program-name* **at displacement** *displacement* **was zero.**

Explanation: The length of the argument of the REVERSE, the UPPER-CASE or the LOWER-CASE function is zero.

System action: The application was terminated.

Programmer response: Make sure that the length of the argument is greater than zero.

Symbolic Feedback Code: IGZ04U

IGZ0159S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than 1 or greater than 3067671.**

Explanation: The input argument to DATE-OF-INTEGER or DAY-OF-INTEGER function is less than 1 or greater than 3067671.

System action: The application was terminated.

Programmer response: Check that the function argument is in the valid range.

Symbolic Feedback Code: IGZ04V

IGZ0160S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than 16010101 or greater than 99991231.**

Explanation: The input argument to function INTEGER-OF-DATE is less than 16010101 or greater than 99991231.

System action: The application was terminated.

Programmer response: Check that the function argument is in the valid range.

Symbolic Feedback Code: IGZ050

IGZ0161S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than 1601001 or greater than 9999365.**

Explanation: The input argument to function INTEGER-OF-DAY is less than 1601001 or greater than 9999365.

System action: The application was terminated.

Programmer response: Check that the function argument is in the valid range.

Symbolic Feedback Code: IGZ051

IGZ0162S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than 1 or greater than the number of positions in the program collating sequence.**

Explanation: The input argument to function CHAR is less than 1 or greater than the highest ordinal position in the program collating sequence.

System action: The application was terminated.

Programmer response: Check that the function argument is in the valid range.

Symbolic Feedback Code: IGZ052

IGZ0163S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than zero.**

Explanation: The input argument to function RANDOM is less than 0.

System action: The application was terminated.

Programmer response: Correct the argument for function RANDOM in the failing statement.

Symbolic Feedback Code: IGZ053

IGZ0164C *module-name* **was unable to get HEAP storage.**

Explanation: The request made to obtain heap storage failed.

System action: The application was terminated.

Programmer response: See your IBM service representative.

Symbolic Feedback Code: IGZ054

IGZ0165S **A reference modification start position value of** *start-position-value* **on line** *line* **referenced an area outside the region of the function result of** *function-result***.**

Explanation: The value of the starting position in a reference modification specification was less than 1, or was greater than the current length of the function result that was being reference modified. The starting position value must be a positive integer less than or equal to the number of characters in the reference modified function result.

System action: The application was terminated.

Programmer response: Check the value of the starting position in the reference modification specification and the length of the actual function result.

Symbolic Feedback Code: IGZ055

IGZ0166S **A non-positive reference modification length value of** *length* **on line** *line-number* **was found in a reference to the function result of** *function-result***.**

Explanation: The length value in a reference modification specification for a function result was less than or equal to 0. The length value must be a positive integer.

System action: The application was terminated.

Programmer response: Check the length value and make appropriate correction.

Symbolic Feedback Code: IGZ056

IGZ0167S **A reference modification start position value of** *start-position* **and length value of** *length* **on line** *line-number* **caused reference to be made beyond the rightmost character of the function result of** *function-result***.**

Explanation: The starting position and length value in a reference modification specification combine to address an area beyond the end of the reference modified function result. The sum of the starting position and length value minus one must be less than or equal to the number of characters in the reference modified function result.

System action: The application was terminated.

Programmer response: Check the length of the reference modification specification against the actual length of the function result and make appropriate corrections.

Symbolic Feedback Code: IGZ057

IGZ0168S **The creation of a second enclave within a reusable environment was attempted. The first program of the second enclave was** *program-name***.**

Explanation: Reusable environment support is limited to a single enclave. The enclave must be the first enclave.

System action: The application was terminated.

Programmer response: Modify the application so that it can run within a single enclave with the COBOL reusable environment. If the program name printed is "????????" then the first program of the second enclave is not COBOL.

Symbolic Feedback Code: IGZ058

IGZ0169W External data *data-record* was allocated within the 31-bit address range. The called program *program-name* contained a definition for this external data, and it was compiled with the DATA(24) option.

Explanation: External data was allocated ANYWHERE within the 31-bit addressing range by a program. But a subsequently called program containing a definition for that same external data was compiled with the DATA(24) option. This was discovered while processing external data records during program initialization.

System action: No system action was taken.

Programmer response: Re-compile program with the DATA(31) option if appropriate. If the external data needs to be allocated below 16M, then the FIRST program in the rununit that contains a definition of the external data must be compiled with the DATA(24) option.

Symbolic Feedback Code: IGZ059

IGZ0170S One or more files were not closed by NORENT program *program-name* and the program cannot be found in storage.

Explanation: The specified NORENT program has not closed all of the files it opened and the program cannot be found in storage. COBOL is unable to close the files because the required control blocks which reside in the program are no longer available. Unpredictable results will occur when the system attempts to close the files. This error can occur if the application has an assembler program that loads and deletes the specified NORENT program.

System action: The application was terminated.

Programmer response: Ensure that all files are closed by the NORENT program.

Symbolic Feedback Code: IGZ05A

IGZ0171S An attempt was made by program *program-name* to open VSAM variable-length relative record file *file-name* but the file was not defined to VSAM as RRDS. The COBOL variable-length relative record data set simulation provided by SIMVRD is no longer supported.

Explanation: The SIMVRD run-time option can no longer be used to provide simulation of VSAM variable-length RRDS in programs compiled with Enterprise COBOL V4R1 and later.

System action: The application was terminated.

Programmer response: Change the file to a VSAM variable-length RRDS file and use native VSAM support for variable-length RRDS data sets.

Symbolic Feedback Code: IGZ05B

IGZ0172W RTEREUS was specified, but ignored. A reusable run-time environment was not established because the first program in the application was not COBOL.

Explanation: A reusable environment can be established only when the main program of the first enclave is COBOL.

System action: No system action is taken.

Programmer response: Ensure that RTEREUS is off. The performance benefits of using RTEREUS are available without the run-time option when the application is running under Language Environment.

Symbolic Feedback Code: IGZ05C

IGZ0173S There was an invalid attempt to start a sort or merge.

Explanation: A sort or merge initiated by a COBOL program was already in progress when another sort or merge was attempted by another COBOL program. Only one sort or merge can be active at a time.

System action: The application was terminated.

Programmer response: Change the application so that it does not initiate another sort or merge from within the COBOL sort exists.

Symbolic Feedback Code: IGZ05D

IGZ0174S A dynamic call to *module-name* failed because the load module is a DLL.

Explanation: A COBOL dynamic call cannot be made to a load module that is a DLL. A load module that is a DLL contains one or more of the following:

- A COBOL for OS/390 & VM program compiled with the DLL option and the EXPORTALL option.
- A C routine compiled with the DLL option that exports functions or variables.
- A C++ routine that exports functions or variables.

System action: The application was terminated.

Programmer response: Change the dynamically called load module so that it does not contain routines that export functions or variables. If the load module contains COBOL for OS/390 & VM programs compiled with the DLL and the EXPORTALL options, recompile the programs with NOEXPORTALL.

Symbolic Feedback Code: IGZ05E

IGZ0175S A dynamic call to *module-name* failed because the entry point is a COBOL program compiled with the DLL compiler option.

Explanation: A COBOL dynamic call cannot be made to a COBOL for OS/390 & VM program that is compiled with the DLL compiler option.

System action: The application was terminated.

Programmer response: Compile the COBOL for OS/390 & VM program with the NODLL compiler option.

Symbolic Feedback Code: IGZ05F

IGZ0176S A call from a COBOL program compiled with the DLL compiler option failed because the program *program-name* was previously dynamically called by a COBOL program compiled without the DLL compiler option.

Explanation: When dynamically calling a COBOL program, insure that the DLL compiler option is consistent between calling and called programs.

System action: The application was terminated.

Programmer response: Compile both the calling and called COBOL for OS/390 & VM programs with either the DLL or the NODLL compiler option.

Symbolic Feedback Code: IGZ05G

IGZ0177S A CANCEL of DLL *program-name* is not allowed.

Explanation: The program was called with a CALL identifier statement from a COBOL program compiled with the DLL option. This caused the called program to be identified as a DLL. A DLL cannot be cancelled.

System action: The application was terminated.

Programmer response: Do not request that a DLL be cancelled.

Symbolic Feedback Code: IGZ05H

IGZ0178S An attempt to find program *program-name* in DLL *module-name* was unsuccessful.

Explanation: An error during the load of a DLL or during a query DLL function request prevented an entry point address from being returned.

System action: The application was terminated.

Programmer response: See the corresponding CEEnnnnI message for additional information and the details of the problem. If the CEEnnnn message is not found in the MSGFILE insure that the runtime option INFOMSGFILTER is OFF.

Symbolic Feedback Code: IGZ05I

IGZ0179S A dynamic call to *module-name* failed because the load module contains one or more routines with XPLINK linkage.

Explanation: A COBOL dynamic call cannot be made to a load module that contains routines with XPLINK linkage.

System action: The application was terminated.

Programmer response: Change the dynamically called load module so that it does not contain routines that use XPLINK linkage.

Symbolic Feedback Code: IGZ05J

IGZ0180S An attempt was made to run a VS COBOL II or OS/VS COBOL program in a OS/390 UNIX process. The program name is *program-name*.

Explanation: VS COBOL II and OS/VS COBOL programs cannot be run in a z/OS UNIX process.

System action: The application was terminated.

Programmer response: Compile the program with COBOL for MVS & VM or COBOL for OS/390 & VM.

Symbolic Feedback Code: IGZ05K

IGZ0181S An attempt was made to run a COBOL program that is not reentrant in a OS/390 UNIX process. The program name is *program-name*.

Explanation: COBOL programs running in a z/OS UNIX process must be reentrant.

System action: The application was terminated.

Programmer response: In order to make a COBOL program reentrant, compile the COBOL program with the RENT compile-time option.

Symbolic Feedback Code: IGZ05L

IGZ0182W A fork() is not allowed when a COBOL reusable environment is active.

Explanation: A COBOL reusable environment is active and the fork() function was called. A COBOL reusable environment is established by doing one of the following:

- Using the RTEREUS run-time option
- Calling ILBOSTP0
- Calling IGZERRE

System action: The fork() function is not performed.

Programmer response: Change the application so that a COBOL reusable environment is not used.

Symbolic Feedback Code: IGZ05M

IGZ0183W A fork() is not allowed when an OS/VS COBOL program or a VS COBOL II program is in the environment.

Explanation: At least one OS/VS COBOL program or VS COBOL II program is in the environment and the fork() function was called.

System action: The fork() function is not performed.

Programmer response: Compile all OS/VS COBOL programs and the VS COBOL II programs with COBOL for MVS & VM or COBOL for OS/390 & VM.

Symbolic Feedback Code: IGZ05N

IGZ0184W A fork() is not allowed when a sort or merge is in progress.

Explanation: A SORT or MERGE statement is in progress and the fork() function was called.

System action: The fork() function is not performed.

Programmer response: Change the application to call fork() when sort or merge is not active.

Symbolic Feedback Code: IGZ05O

IGZ0185W A fork() is not allowed when a declarative in a COBOL program is active.

Explanation: A declarative in a COBOL program is active and the fork() function was called.

System action: The fork() function is not performed.

Programmer response: Change the application to call fork() when a declarative is not active.

Symbolic Feedback Code: IGZ05P

IGZ0186S An attempt was made to run a VS COBOL II program with the run-time option XPLINK(ON). The program name is *program-name*.

Explanation: Run-time option XPLINK(OFF) must be specified to run VS COBOL II programs.

System action: The application was terminated.

Programmer response: Set the XPLINK run-time option to OFF and remove any load modules from the application that use XPLINK linkage, or compile the COBOL program with COBOL for MVS & VM or COBOL for OS/390 & VM.

Symbolic Feedback Code: IGZ05Q

IGZ0187S There was an attempt to establish a COBOL reusable environment with the run-time option XPLINK(ON).

Explanation: A COBOL reusable environment cannot be established when the XPLINK(ON) run-time option is specified. A COBOL reusable environment is established by doing one of the following:

- Using the RTEREUS run-time option
- Calling ILBOSTP0
- Calling IGZERRE

System action: The application was terminated.

Programmer response: Set the XPLINK run-time option to OFF and remove any load modules from the application that use XPLINK linkage, or do not use a COBOL reusable environment.

Symbolic Feedback Code: IGZ05R

IGZ0188S Value *string* is invalid for environment variable _IGZ_SYSOUT.

Explanation: Allowable values for environment variable _IGZ_SYSOUT are "stdout" or "stderr". Value can be any combination of upper and lower case and must not contain leading or trailing spaces.

System action: The application was terminated.

Programmer response: Change value to be either "stdout" or "stderr".

Symbolic Feedback Code: IGZ05S

IGZ0189S Program *program-name* cannot be run in this operating system environment.

Explanation: The program contains features that are not supported in this operating system environment. For example, when running on CMS, the following features are not supported:

- programs compiled with the DLL compiler option
- programs compiled with the ARITH(EXTEND) compiler option

- programs compiled with Enterprise COBOL for z/OS and OS/390 V3R1 and later

System action: The application was terminated.

Programmer response: Modify the program to use supported features for the environment or run the program in the appropriate environment.

Symbolic Feedback Code: IGZ05T

IGZ0190S **There was an attempt to parse an XML document with an ENCODING phrase that specified an invalid CCSID.**

Explanation: The CCSID value must be a positive integer between 1 and 65,535 inclusive.

System action: The application was terminated.

Programmer response: Correct the ENCODING phrase of the XML PARSE statement.

Symbolic Feedback Code: IGZ05U

IGZ0191S **There was an attempt to parse an XML document in a national data item but with an ENCODING phrase that specified a CCSID other than 1200.**

Explanation: The CCSID value for parsing a national document must be 1200.

System action: The application was terminated.

Programmer response: Correct the ENCODING phrase of the XML PARSE statement.

Symbolic Feedback Code: IGZ05V

IGZ0192S **There was an attempt to parse an XML document in an alphanumeric data item but with an ENCODING phrase that specified CCSID 1200.**

Explanation: The CCSID value for parsing an alphanumeric document must be 1208 (Unicode UTF-8), one of the supported EBCDIC CCSIDs, or, if the RETURNING NATIONAL phrase was specified, any CCSID that can be converted to CCSID 1200 (national).

System action: The application was terminated.

Programmer response: Correct the ENCODING phrase of the XML PARSE statement.

Symbolic Feedback Code: IGZ060

IGZ0193W **Search argument** *argument-number* **in the WHEN phrase of the SEARCH ALL statement in program** *program-name* **on line number** *line-number* **was a signed item with a negative value. The corresponding table key was an unsigned item, and so the argument could never match the key in any of the table entries.**

Explanation: The SEARCH ALL statement specified a search argument that was a signed item with a negative sign. The table key was an unsigned numeric item, and so the comparison of the argument and the keys would always be unequal. Hence the SEARCH ALL statement could never succeed in locating a matching table entry.

System action: No system action was taken.

Programmer response: Ensure that the search argument is correctly initialized before issuing the SEARCH ALL statement. For an unsigned table key, a numeric argument must either be unsigned or have a positive sign for the search statement to have a possibility of successfully finding a matching table entry.

Symbolic Feedback Code: IGZ061

IGZ0194W **Search argument** *argument-number* **in the WHEN phrase of the SEARCH ALL statement in program** *program-name* **on line number** *line-number* **was longer than the corresponding key. The excess digit or character positions of the argument were not zeros or spaces respectively, and so the argument could never match the key in any of the table entries.**

Explanation: The SEARCH ALL statement specified a search argument that was longer than the table key, and since the excess digits or characters were not zeros or spaces respectively, the comparison of the argument and the keys would always be unequal. Hence the SEARCH ALL statement could never succeed in locating a matching table entry.

System action: No system action was taken.

Programmer response: Initialize the excess argument positions to zeros or blanks as appropriate before issuing the SEARCH ALL statement. Alternatively, use a MOVE statement to move the argument to a shorter temporary variable to truncate the excess argument positions, then use that temporary variable as the SEARCH ALL argument.

Symbolic Feedback Code: IGZ062

IGZ0195S A SORT or MERGE statement was attempted when running in an OS/390 UNIX process.

Explanation: SORT and MERGE statements are not supported when running in a z/OS UNIX process.

System action: The application was terminated.

Programmer response: Remove the SORT or MERGE statements from the application or run the program in a z/OS environment.

Symbolic Feedback Code: IGZ063

IGZ0196S Argument-1 for function *function-name* **in program** *program-name* **at displacement** *displacement* **contained more than 31 digits.**

Explanation: The total number of digits in argument-1 of the indicated function exceeded 31 digits.

System action: The application was terminated.

Programmer response: Adjust the number of digits in argument-1 in the failing statement.

Symbolic Feedback Code: IGZ064

IGZ0197S There was an unsuccessful READ or WRITE of file *file-name* **in program** *program-name* **at relative location** *location* **Neither FILE STATUS nor an ERROR declarative were specified. The file status code was** *status-code***. The BPX return code was** *return-code***. The BPX reason code was** *reason-code***.**

Explanation: An error has occurred while reading or writing the named file. No file status or user error declarative was specified.

System action: The application was terminated.

Programmer response: For additional information on the return-code and reason-code, see *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SC23-3020, topics for read (BPX1RED) and write (BPX1WRT).

Symbolic Feedback Code: IGZ065

IGZ0198W File *file-name* **in program** *program-name* **had a block size of** *block-size* **which exceeds the maximum supported block size.**

Explanation: The program file description specified a block size that exceeds the maximum supported block size. The OPEN statement failed.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Ensure that the block size specified in the BLOCK CONTAINS clause is supported for the file, for the device that the dataset resides on and for the operating system level being used.

Symbolic Feedback Code: IGZ066

IGZ0199S An attempt was made to run a COBOL program that was compiled with the SEPARATE suboption of the TEST compiler option. This is not supported with this level of Language Environment or this level of Debug Tool. The program name is *program-name*.

Explanation: COBOL programs running with TEST(,,SEPARATE) must run on levels of Language Environment and Debug Tool that support it. This error can occur with any of the following:

- running with a level of Language Environment that does not support the SEPARATE suboption
- running with a level of Language Environment that could support the SEPARATE suboption but does not have current maintenance applied
- running with a level of Debug Tool that could support the SEPARATE suboption but does not have current maintenance applied

System action: The application was terminated.

Programmer response: Run the program under levels of Language Environment and Debug Tool that support programs compiled with TEST(,,SEPARATE) or recompile the COBOL program without the SEPARATE suboption of the TEST compiler option.

Symbolic Feedback Code: IGZ067

IGZ0200W A file attribute mismatch was detected. File *file-name* in program *program-name* was defined as a physical sequential file and the file specified in the ASSIGN clause was a VSAM data set.

Explanation: The program file description specified that the file was a physical sequential file and the data set associated with the ASSIGN clause was found to be a VSAM file. The OPEN statement failed.

System action: If a file status was specified, no system action is performed. If a file status field was not specified, the program is terminated and message IGZ0035S is generated.

Programmer response: Check that the file description and the DD parameter associated with the ASSIGN clause are for the correct data set.

Symbolic Feedback Code: IGZ068

IGZ0201W A file attribute mismatch was detected. File *file-name* in program *program-name* had a record length of *record-length-1* and the file specified in the ASSIGN clause had a record length of *record-length-2*.

Explanation: The program file description specified a record length that did not match the record length of the data set associated with the ASSIGN clause. The OPEN statement failed.

System action: If a file status was specified, no system action is performed. If a file status field was not specified, the program is terminated and message IGZ0035S is generated.

Programmer response: For Format-V and Format-S files the maximum record length specified in your program must be exactly 4 bytes smaller than the length attribute of the data set. For Format-F files, the record length specified in your program must exactly match the length attribute of the data set. For Format-U files, the maximum record length specified in your program must exactly match the length attribute of the data set. If your file is a printer file, the compiler may add one byte to the file description for carriage control character, depending on the ADV compiler option and the COBOL statements used in your program. In which case, the added byte must be included in the data set length attribute. For VSAM files, the record length must not be greater than the maximum length attribute of the data set. For VSAM simulated RRDS (SIMVRD run-time option) the record length specified in the ASSIGN clause is incremented by 4 bytes before comparison with the length attribute of the data set.

Symbolic Feedback Code: IGZ069

IGZ0202W A file attribute mismatch was detected. File *file-name* in program *program-name* specified ASCII data and the file specified in the ASSIGN clause did not contain the ASCII data attribute.

Explanation: The CODE-SET clause was specified in the program file description and the data set associated with the ASSIGN clause did not contain ASCII data. The OPEN statement failed.

System action: If a file status was specified, no system action is performed. If a file status field was not specified, the program is terminated and message IGZ0035S is generated.

Programmer response: Check that the data set associated with the ASSIGN clause is the correct one, and if it is, check the data set for the ASCII attribute.

Symbolic Feedback Code: IGZ06A

IGZ0203W A file attribute mismatch was detected. File *file-name* in program *program-name* specified non-ASCII data and the file specified in the ASSIGN clause contained the ASCII data attribute.

Explanation: The data set associated with the ASSIGN clause contained ASCII type data and the file description in the program did not contain ASCII data. The OPEN statement failed.

System action: If a file status was specified, no system action is performed. If a file status field was not specified, the program is terminated and message IGZ0035S is generated.

Programmer response: Check that the data set associated with the ASSIGN clause is the correct one, and if it is, check the data set for the ASCII attribute.

Symbolic Feedback Code: IGZ06B

IGZ0204W A file attribute mismatch was detected. File *file-name* in program *program-name* was defined as RECORDING MODE *recording-mode* and the file specified in the ASSIGN clause did not contain the same attribute.

Explanation: The RECORDING MODE specified in the program file description did not match the data control block fields of the data set associated with the ASSIGN clause. The OPEN statement failed.

System action: If a file status was specified, no system action is performed. If a file status field was not specified, the program is terminated and message IGZ0035S is generated.

Programmer response: Check the data control block fields of the actual data set to verify that the RECORDING MODE matches. The most common cause of this error is conflicting fixed and variable record length data set attributes.

Symbolic Feedback Code: IGZ06C

IGZ0205W An OPEN failure occurred for file *file-name* in program *program-name* because the SMSVSAM server was not available. The file was closed.

Explanation: COBOL encountered a SMSVSAM server not available error return while performing OPEN, I/O, or control block testing of a VSAM data set in RLS mode. For this error condition VSAM requires that the file be closed, opened, and positioned before resubmitting requests. Look for possible VSAM error messages in the job log.

System action: No system action is performed.

Programmer response: COBOL only performs a close of the file. Resolve the SMSVSAM server not available condition and resubmit the run or remove the RLS keyword specification from the DD statement.

Symbolic Feedback Code: IGZ06D

IGZ0206W The AIXBLD run-time option was invalid for file *file-name* in program *program-name* because the file was opened in RLS mode. The file was closed.

Explanation: The AIXBLD option is only supported for VSAM data sets opened without RLS mode. VSAM data sets opened in RLS mode can be empty, but upgrades to empty paths are not supported. The alternate index path must be built before using RLS mode. The alternate index was not built and the file was closed.

System action: No system action is performed.

Programmer response: If AIXBLD option is required, remove the RLS keyword specification from the DD statement for this file and resubmit the run.

Symbolic Feedback Code: IGZ06E

IGZ0207W The SIMVRD run-time option was invalid for file *file-name* in program *program-name* because the file was opened in RLS mode. The file was closed.

Explanation: The SIMVRD option is not supported for VSAM data sets in RLS mode. The file was closed.

System action: No system action is performed.

Programmer response: If SIMVRD option is required, remove the RLS keyword specification from the DD statement for this file and resubmit the run.

Symbolic Feedback Code: IGZ06F

IGZ0210S There was an attempt to run an OS/VS COBOL program *program-name* in a non-initial thread.

Explanation: OS/VS COBOL programs can only run in the initial thread. For example, OS/VS COBOL programs can not run in a subtask created by a PL/I CALL statement with the TASK, EVENT, or PRIORITY option.

System action: The application is terminated.

Programmer response: Compile the COBOL program with the COBOL for MVS & VM or COBOL for OS/390 & VM compiler.

Symbolic Feedback Code: IGZ06I

IGZ0215S Argument —1 for function *function-name* in program *program-name* at line *line-number* was less than 0 or greater than 99.

Explanation: An illegal value was used for Argument-1.

System action: The application is terminated.

Programmer response: Ensure that argument-1 is greater than, or equal to 0, and less than 100.

Symbolic Feedback Code: IGZ06N

IGZ0216S Argument —1 for function *function-name* in program *program-name* at line *line-number* was less than 0 or greater than 99366.

Explanation: An illegal value was used for Argument-1.

System action: The application is terminated.

Programmer response: Ensure that argument-1 is greater than, or equal to 0, and less than 99367.

Symbolic Feedback Code: IGZ06O

IGZ0217S Argument —1 for function *function-name* in program *program-name* at line *line-number* was less than 0 or greater than 991231.

Explanation: An illegal value was used for Argument-1.

System action: The application is terminated.

Programmer response: Ensure that argument-1 is greater than, or equal to 0, and less than 991231.

Symbolic Feedback Code: IGZ06P

IGZ0218S The sum of the year at the time of execution and the value of argument —2 was less than 1700 or greater than 10000 for function *function-name* in program *program-name* at line *line-number*.

Explanation: An illegal value was used for Argument-2.

System action: The application is terminated.

Programmer response: Ensure that the sum of the year at the time of execution and the value of argument-2 is less than 1700 or greater than 10000.

Symbolic Feedback Code: IGZ06Q

IGZ0219S The base year for program *program-name* **was outside the valid range of current year minus 99 through 1999. The sliding window value** *window-value* **resulted in a base year of** *base-year***.**

Explanation: The current year was outside the 100-year fixed window specified by the YEARWINDOW compiler option value.

For example, if a COBOL program is compiled with YEARWINDOW(1920), the 100-year window for the program is 1920 and 2019. When the program is run in the year 2020, this error message would occur since the current year is not within the 100-year window.

System action: The application was terminated.

Programmer response: Examine the application design to determine if it will support a change to the YEARWINDOW option value. If the application can run with a change to the YEARWINDOW option value, then compile the program with an appropriate YEARWINDOW option value. If the application cannot run with a change to the YEARWINDOW option value, then convert all date fields to expanded dates and compile the program with NODATEPROC.

Symbolic Feedback Code: IGZ06R

IGZ0220S The current year was outside the 100-year window, *year-start* **through** *year-end* **for program** *program***.**

Explanation: The current year was outside the 100-year fixed window specified by the YEARWINDOW compiler option value.

For example, if a COBOL program is compiled with YEARWINDOW(1920), the 100-year window for the program is 1920 through 2019. When the program is run in the year 2020, this error message would occur since the current year is not within the 100 year window.

System action: The application was terminated.

Programmer response: Examine the application design to determine if it will support a change to the YEARWINDOW option value. If the application can run with a change to the YEARWINDOW option value, then compile the program with an appropriate YEARWINDOW option value. If the application cannot run with a change to the YEARWINDOW option value, then convert all date fields to expanded dates and compile the program with NODATEPROC.

Symbolic Feedback Code: IGZ06S

IGZ0221W The Y2PAST= *y2past-value* **SORT option (from the YEARWINDOW compiler option) was overridden by the Y2PAST value in the sort control file.**

Explanation: A windowed date field was specified as a KEY in a SORT or MERGE, which resulted in the YEARWINDOW compiler option being converted into a SORT option Y2PAST value, but Y2PAST was also specified in the sort control file.

The value in the sort control file was used, and the Y2PAST value from the program was ignored.

System action: No system action was taken.

Programmer response: See *COBOL for OS/390 & VM Programming Guide* or *Enterprise COBOL for z/OS, V3R4, Programming Guide* for a description of using windowed date fields with SORT and MERGE.

Symbolic Feedback Code: IGZ06T

IGZ0222S No significant digits remain in a fixed-point exponentiation operation in program *program* **at displacement** *displacement* **due to excessive decimal positions specified in the operands or receivers.**

Explanation: A fixed-point exponentiation operation that specifies a negative exponent could not be completed because all significant digits were lost after the operands were scaled. This condition is caused by excessive decimal positions being specified in the operands or receivers of the expression.

System action: The application was terminated.

Programmer response: Simplify the arithmetic expression, specifying less decimal positions in the operands.

Do not use exponentiation of a base having 31 decimal positions, using a negative integral exponent. Rather, use an exponentiation specifying a positive exponent followed by an explicit division operation.

Alternatively, use a floating-point expression. To do this, specify at least one floating-point operand or receiver.

Symbolic Feedback Code: IGZ06U

IGZ0223S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was less than zero or greater than 29.**

Explanation: Input argument to function FACTORIAL is greater than 29 or less than 0.

System action: The application was terminated.

Programmer response: Check that the function argument is in the valid range.

Symbolic Feedback Code: IGZ06V

IGZ0224S **There was an attempt to run COBOL programs in more than one thread and all of the COBOL programs were not enabled for multithreading. The error was detected when attempting to run COBOL program** *program-name***.**

Explanation: There was an attempt to run COBOL programs that are not enabled for multithreading in more than one thread. In order to run COBOL programs in more than one thread, all of the COBOL programs in the application must be compiled with the Enterprise COBOL compiler using the THREAD compiler option.

COBOL programs compiled with the following compilers can only run in one thread at a time:
- Enterprise COBOL with the NOTHREAD compiler option
- COBOL for OS/390 & VM
- COBOL for MVS & VM
- COBOL/370
- VS COBOL II
- OS/VS COBOL

This condition can occur when PL/I multitasking is used or when POSIX(ON) is in effect.

If PL/I multitasking is used, here are examples that can cause this condition:
- If an Enterprise COBOL program compiled without the THREAD compiler option or a COBOL for OS/390 & VM program has been invoked in the main task, then any attempts to invoke a COBOL program in a subtask created by a PL/I statement with the TASK, EVENT or the PRIORITY option will cause this condition to be signaled.
- If an Enterprise COBOL program compiled without the THREAD compiler option or a COBOL for OS/390 & VM program has been invoked in a subtask, then any attempts to invoke a COBOL program in any other task will cause this condition to be signaled until the subtask is terminated.

If POSIX(ON) is in effect, here are examples that can cause this condition:
- If an Enterprise COBOL program compiled without the THREAD compiler option or a COBOL for OS/390 & VM program has been invoked in the initial thread, then any attempts to invoke a COBOL program in a non-initial thread will cause this condition to be signaled.
- If an Enterprise COBOL program compiled without the THREAD compiler option or a COBOL for OS/390 & VM program has been invoked in a non-initial thread, then any attempts to invoke a COBOL program in another thread will cause this condition to be signaled until the non-initial thread in which a COBOL program was invoked is terminated.

System action: The application is terminated.

Programmer response: Compile all the COBOL programs with the Enterprise COBOL using the THREAD compiler option.

Symbolic Feedback Code: IGZ070

IGZ0225S **There was an attempt to run the thread enabled COBOL program** *program-name* **in a COBOL reusable environment.**

Explanation: Enterprise COBOL programs compiled with the THREAD compiler option cannot run in a COBOL reusable environment.

A COBOL reusable environment is established by doing one of the following:

- Using the RTEREUS run-time option
- Calling ILBOSTP0
- Calling IGZERRE

System action: The application is terminated.

Programmer response: If you want to continue to run with the COBOL reusable environment, compile the COBOL program with the NOTHREAD compiler option. If you want to run COBOL programs compiled with the THREAD option in a preinitialized environment, use the Language Environment preinitialization support provided by CEEPIPI.

Symbolic Feedback Code: IGZ071

IGZ0226S On CICS, an attempt was made to run a COBOL program that contains object-oriented syntax. The program name is *program-name*.

Explanation: COBOL programs running on CICS cannot contain any object-oriented syntax.

System action: The application is terminated.

Programmer response: Change the COBOL program so that it does not contain object-oriented class definitions, INVOKE statements, or references to the JNIEnvPtr special register.

Symbolic Feedback Code: IGZ072

IGZ0227S There was an invalid attempt to end an XML PARSE statement.

Explanation: An XML PARSE statement initiated by a COBOL program was in progress and one of the following was attempted:

1. A GOBACK or an EXIT PROGRAM was issued within the COBOL program that initiated the XML PARSE.
2. A user handler associated with the program that initiated the XML PARSE moved the condition handler resume cursor and resumed the application.

System action: The application is terminated.

Programmer response: Change the application so that it does not use one of the above methods to end the XML PARSE statement.

Symbolic Feedback Code: IGZ073

IGZ0228S There was an invalid attempt to start an XML PARSE statement.

Explanation: An XML PARSE statement initiated by a COBOL program was already in progress when another XML PARSE statement was attempted by the same COBOL program. Only one XML PARSE statement can be active in a given invocation of a COBOL program.

System action: The application is terminated.

Programmer response: Change the application so that it does not initiate another XML PARSE statement from within the same COBOL program.

Symbolic Feedback Code: IGZ074

IGZ0229S Argument–2 for function *function-name* in program *program-name* at line *line-number* was less than 1 or greater than 65535.

Explanation: Argument-2, the CODEPAGE, for the indicated function was not within the range of 1 thru 65535.

System action: The application is terminated.

Programmer response: Ensure the argument-2 for the indicated function is within the valid range of 1 thru 65535.

Symbolic Feedback Code: IGC075

IGZ0230S **The processing procedure for an XML PARSE statement set XML-CODE to an unsupported value of** *xml-code-value* **for XML event** *xml-event***.**

Explanation: The XMLPARSE(XMLSS) compiler option was in effect. For the END-OF-INPUT XML event, XML-CODE can be set to a value of -1, zero or 1. For all other XML events, only -1 and zero are supported.

System action: The application was terminated.

Programmer response: Correct the statements that set the value of the XML-CODE special register.

Symbolic Feedback Code: IGZ076

IGZ0251W **An invalid keyword** *keyword* **was found at position** *position* **in environment variable** *env-var* **while processing file** *file-name* **in program** *program-name***.**

Explanation: While processing the allocation of the file, the specified keyword was encountered at the indicated position. The keyword was not allowed in the environment variable. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Remove the identified keyword from the environment variable. Also, make sure that all keywords are in uppercase. Additionally, for QSAM and VSAM files, make sure that the DSN or PATH keyword is specified first in the environment variable. For line sequential files, make sure that the PATH keyword is the only keyword specified in the environment variable.

Symbolic Feedback Code: IGZ07R

IGZ0252W **An invalid delimiter was found at position** *position* **in environment variable** *env-var* **while processing file** *file-name* **in program** *program-name***.**

Explanation: While processing the allocation of the file, the delimiter at the indicated position was invalid in the context used in the environment variable. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Correct the identified delimiter in the environment variable.

Symbolic Feedback Code: IGZ07S

IGZ0253W **Keyword** *keyword1* **was found at position** *position* **in environment variable** *env-var* **while processing file** *file-name* **in program** *program-name* **and is mutually exclusive with keyword** *keyword2***.**

Explanation: While processing the allocation of the file, *keyword1* was found at the indicated position in the environment variable but cannot be specified with *keyword2* which was found earlier. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Remove either *keyword1* or *keyword2* from the environment variable.

Symbolic Feedback Code: IGZ07T

IGZ0254W **Environment variable** *env-var* **is null or only contains blanks for file** *file-name* **in program** *program-name***.**

Explanation: The contents of the environment variable that was used for the allocation of the file were null or only contained blanks. The file cannot be allocated. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: For QSAM and VSAM files, make sure that the ddname is properly defined for the file by specifying a ddname for the file setting an environment variable with the same name as the ddname to identify the file. For line sequential files, make sure that the file is properly defined by setting an environment variable with the same name as the file name to identify the file using an absolute pathname.

Symbolic Feedback Code: IGZ07U

IGZ0255W **Dynamic allocation failed for ddname** *ddname* **while processing file** *file-name* **in program** *program-name***. The return code from the dynamic allocation was** *return-code***, error code** *error-code***, reason code** *reason-code***, and information code** *information-code***.**

Explanation: A error occurred issuing the dynamic allocation for the ddname. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: See the JOBLOG output for any additional messages from data management explaining the error. For additional information about the error codes, see the topic about interpreting DYNALLOC return codes in *z/OS MVS Programming: Authorized Assembler Services Guide*.

Symbolic Feedback Code: IGZ07V

IGZ0256W **Dynamic deallocation failed for ddname** *ddname* **while processing file** *file-name* **in program** *program-name***. The return code from the dynamic deallocation was** *return-code***, error code** *error-code***, reason code** *reason-code***, and information code** *information-code***.**

Explanation: An error occurred issuing the dynamic deallocation for the ddname. The CLOSE statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: See the JOBLOG output for any additional messages from data management explaining the error. For additional information about the error codes, see the topic about interpreting DYNALLOC return codes in *z/OS MVS Programming: Authorized Assembler Services Guide*.

Symbolic Feedback Code: IGZ080

IGZ0257W **The environment variable** *env-var* **for file** *file-name* **in program** *program-name* **contains an invalid dataset name value.**

Explanation: While processing the allocation of the file, the dataset name specified in the DSN keyword of the environment variable was invalid. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Correct the dataset name in the DSN keyword of the environment variable.

Symbolic Feedback Code: IGZ081

IGZ0258W **The environment variable** *env-var* **for file** *file-name* **in program** *program-name* **contains an invalid member name value.**

Explanation: While processing the allocation of the file, the member name specified in the DSN keyword of the environment variable was invalid. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Correct the member name in the DSN keyword of the environment variable.

Symbolic Feedback Code: IGZ082

IGZ0259W The environment variable *env-var* for file *file-name* in program *program-name* contains an invalid path name value.

Explanation: While processing the allocation of the file, the path name specified in the PATH keyword of the environment variable was invalid. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Correct the path name in the PATH keyword of the environment variable.

Symbolic Feedback Code: IGZ083

IGZ0260W Temporary dataset names cannot be used for dynamic allocation in environment variable *env-var* for file *file-name* in program *program-name*.

Explanation: The contents of the environment variable that was used for the allocation of the file specified a temporary dataset name in the DSN parameter. Temporary dataset names are not supported for dynamic allocation. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Change the DSN parameter in the specified environment variable to specify a permanent dataset name.

Symbolic Feedback Code: IGZ084

IGZ0261W An absolute pathname was not specified in environment variable *env-var* for file *file-name* in program *program-name*.

Explanation: The contents of the environment variable that was used for the allocation of the file did not specify an absolute pathname in the PATH parameter. Only absolute pathnames are supported. The OPEN statement failed with file status 98.

System action: If a file status or error declarative was specified for the file, no system action is performed. If neither a file status nor an error declarative was specified for the file, the program is terminated and message IGZ0035S is generated.

Programmer response: Change the PATH parameter in the specified environment variable to specify an absolute pathname.

Symbolic Feedback Code: IGZ085

IGZ0262S *C-Runtime-message*

Explanation: An error occurred while attempting to retrieve the optimized schema. The message provides details.

System action: The application was terminated.

Programmer response: Correct the indicated error. For more information regarding the message, see *z/OS Language Environment Runtime Messages*.

Symbolic Feedback Code: IGZ086

| **IGZ0263I The previous condition has also occurred in program *prograrm-name* on line(s): *line-numbers*.**

| **Explanation:** See the explanation for the previous message for details.

| **System action:** No system action was taken.

| **Programmer response:** See the user response for the previous message for details.

Symbolic Feedback Code: IGZ087

IGZ0264S **There was an attempt to run both OS/VS COBOL and Enterprise COBOL V5 programs in the same enclave.**

Explanation: OS/VS COBOL programs can not be run in the same enclave where Enterprise COBOL V5 programs are also running.

System action: The application is terminated.

Programmer response: Compile the OS/VS COBOL program with an Enterprise COBOL compiler.

Symbolic Feedback Code: IGZ088

IGZ0265S **Argument-1 for function** *function-name* **in program** *program-name* **at line** *line-number* **was not a valid UTF-8 string.**

Explanation: Argument-1 is not a valid UTF-8 string.

System action: The application was terminated.

Programmer response: Ensure that argument-1 is a valid UTF-8 string.

Symbolic Feedback Code: IGZ089

IGZ0266S **Argument-2 and argument-3 for function USUBSTR in program** *program-name* **at line** *line-number* **was not a valid combination for substring of a UTF string.**

Explanation: The values for argument-2 and argument-3 for function USUBSTR were invalid.

System action: The application was terminated.

Programmer response: Ensure that argument-2 and argument-3 are valid for UTF-8 substring. Argument-2 must be an integer greater than zero. Argument-3 must be an integer greater than or equal to zero. (Argument-2 + argument-3 - 1) must be less than or equal to ULENGTH(argument-1).

Symbolic Feedback Code: IGZ08A

IGZ0270S **In a multithreaded environment, COBOL program** *program-name* **was called during condition handling and the program cannot be initialized.**

Explanation: In a multithreaded environment, COBOL programs cannot be called as a condition handler for conditions raised while performing COBOL program initialization.

System action: The application is terminated.

Programmer response: Use a condition handler written in assembler or C to handle conditions raised during COBOL program initialization.

Symbolic Feedback Code: IGZ08E

IGZ0271S **There was an attempt to initialize COBOL in a nested enclave when the initial enclave is multithreaded or multitasking. The first program of the nested enclave was** *program-name***.**

Explanation: A COBOL program cannot run in a nested enclave when the initial enclave is multithreaded or multitasking.

System action: The application is terminated.

Programmer response: Change the application so that the COBOL program is not running in a nested enclave or do not use multithreading or multitasking when using COBOL programs in nested enclaves. If the program name printed is "????????", then the first program of the nested enclave is not COBOL.

Symbolic Feedback Code: IGZ08F

IGZ0272S **Data conversion from CCSID** *CCSID1* **to** *CCSID2* **was unsuccessful. The return code from the Unicode Conversion Service CUNLCNV was** *return-code* **and the reason code was** *reason-code***.**

Explanation: The data conversion for the CCSID pair failed as indicated by the return code and the reason code. The return code and the reason code values for conversion services are described in *z/OS Unicode Services User's Guide and Reference*.

System action: The application is terminated.

Programmer response: Follow the programmer responses indicated for specific return and reason codes in *z/OS Unicode Services User's Guide and Reference*.

Symbolic Feedback Code: IGZ08G

IGZ0273S **A GOBACK, EXIT PROGRAM, or STOP RUN was attempted while an EXCEPTION/ERROR declarative was in control for a QSAM ABEND. The declarative in control is in program** *program-name* **for file** *file-name***.**

Explanation: A GOBACK, EXIT PROGRAM, or STOP RUN statement cannot be used while an EXCEPTION/ ERROR declarative is in control due to a QSAM ABEND for a READ, REWRITE, or WRITE statement. When a QSAM abend occurs during a READ, WRITE, or REWRITE, the file status code can be file status 34 or file status 90.

System action: The application is terminated with ABEND 4043.

Programmer response: Change the program to not use a GOBACK, EXIT PROGRAM, or STOP RUN statement when the EXCEPTION/ERROR declarative is in control.

Symbolic Feedback Code: IGZ08H

IGZ0274S **A GOBACK, EXIT PROGRAM, or STOP RUN was attempted while a LABEL declarative was in control. The declarative in control is in program** *program-name* **for file** *file-name***.**

Explanation: A GOBACK, EXIT PROGRAM, or STOP RUN statement cannot be used while a LABEL declarative is in control.

System action: The application is terminated with ABEND 4043.

Programmer response: Change the program to not use a GOBACK, EXIT PROGRAM, or STOP RUN statement when the LABEL declarative is active.

Symbolic Feedback Code: IGZ08I

IGZ0280S **XML to data structure conversion could not complete in program** *PROGRAM-NAME* **because an error return code of** *RETURN-CODE* **was received from the XML PARSE statement. The error occurred at element** *ELEMENT-NAME* **with the character content** *CHARACTER-CONTENT***.**

Explanation: XML to language structure conversion uses the COBOL XML PARSE statement. When the XMLPARSE(COMPAT) compiler option is in effect, the return code is provided from the XML PARSE statement and is described in the Enterprise COBOL Programming Guide. When the XMLPARSE(XMLSS) compiler option is in effect, the return code is provided from the z/OS XML System Services Parser and is described in the XML System Services User's Guide and Reference.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Use the return code to determine the error and correct the input XML message.

Symbolic Feedback Code: IGZ08O

IGZ0281S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because the valid range of a repeating group or repeating data item was exceeded. The error occurred at element** *ELEMENT-NAME***.**

Explanation: XML to language structure conversion encounters errors while transforming an XML message to a language structure. These errors are distinct from XML PARSE errors.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the input XML message. Ensure that the message is valid according to the XML Schema upon which the XML to language structure converter is based.

Symbolic Feedback Code: IGZ08P

IGZ0282S XML to language structure conversion could not complete in program *PROGRAM-NAME* **because none of the elements in the input XML document matched a mapped element in the XML Schema upon which the XML converter is based.**

Explanation: XML to language structure conversion encounters errors while transforming an XML message to a language structure. These errors are distinct from XML PARSE errors.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the input XML message. Ensure that the message is valid according to the XML Schema upon which the XML to language structure converter is based.

Symbolic Feedback Code: IGZ08Q

IGZ0283S XML to language structure conversion could not complete in program *PROGRAM-NAME* **because the character content for element** *ELEMENT-NAME* **was longer than the element's maximum of LIMIT characters.**

Explanation: XML to language structure conversion encounters errors while transforming an XML message to a language structure. These errors are distinct from XML PARSE errors.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the input XML message. Ensure that the message is valid according to the XML Schema upon which the XML to language structure converter is based.

Symbolic Feedback Code: IGZ08R

IGZ0284S XML to language structure conversion could not complete in program *PROGRAM-NAME* **because conversion of the character content of an element that is mapped as numeric failed. The error occurred at element** *ELEMENT-NAME* **with the character content** *CHARACTER-CONTENT***.**

Explanation: XML to language structure conversion encounters errors while transforming an XML message to a language structure. These errors are distinct from XML PARSE errors.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the input XML message. Ensure that the message is valid according to the XML Schema upon which the XML to language structure converter is based.

Symbolic Feedback Code: IGZ08S

IGZ0285S XML to language structure conversion could not complete in program *PROGRAM-NAME* **because the length of the input XML document is INPUTLEN characters, which exceeds the maximum of** *MAXIUMUM* **characters for this converter.**

Explanation: XML to language structure conversion imposes a limit on the length of XML documents that can be converted into language structures. The maximum length in bytes of the input XML document is 16MB when version 3 of Enterprise COBOL for z/OS is used and 32MB when version 4 of Enterprise COBOL for z/OS is used.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Check that the input XML message conforms to the XML Schema upon which it is based. Check any whitespace outside of element content has been trimmed.

Symbolic Feedback Code: IGZ08T

IGZ0286S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because the content for element** *ELEMENT-NAME* **had a length greater than or equal to INPUTLEN characters which was too long for the converter to process.**

Explanation: XML to language structure conversion maintains a buffer for character content that has a maximum size equal to 10 times the size of the largest item in the target language structure. The content of an element was longer than expected.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the input XML message. Ensure that the message is valid according to the XML Schema upon which the XML to language structure converter is based.

Symbolic Feedback Code: IGZ08U

IGZ0287S **Language structure to XML conversion could not complete in program** *PROGRAM-NAME* **because the maximum output message length of LENGTH characters was exceeded while generating the XML document.**

Explanation: Language structure to XML conversion performs whitespace suppression and entity reference expansion while generating the XML document. Entity reference expansion increases the length of data which is included as element content causing it to grow up to 6 times the original length of the data (the longest predefined entity is 6 characters long).

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Ensure that the language structure has not changed since the Language structure to XML converter has been generated. Regenerate the XML converter if changes have been made to the language structure.

Symbolic Feedback Code: IGZ08V

IGZ0288S **XML conversion could not complete in program** *PROGRAM-NAME* **because a non-zero return code was received from the Unicode Conversion Service** *CUNLCNV* **while converting from CCSID SOURCE-CCSID to CCSID TARGET-CCSID.**

Explanation: XML Conversion converts XML and language data to and from Unicode during parsing and generation of XML, if XML is expected to be exchanged in Unicode.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Check that conversion services is properly installed, and ensure that the conversion attempted by the XML Converter is supported by the systems installation of conversion services.

Symbolic Feedback Code: IGZ090

IGZ0289S **Language structure to XML conversion could not complete in program** *PROGRAM-NAME* **because the content of non-numeric member** *MEMBER-NAME* **of language structure** *STRUCTURE-NAME* **contained characters that are not legal in an XML document.**

Explanation: Certain data structure members permit storage of characters that are not legal in an XML document. The permitted characters are defined by the XML specification at `http://www.w3c.org`.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Ensure that the language structure member is properly initialized and does not contain any characters that are illegal in XML before attempting conversion.

Symbolic Feedback Code: IGZ091

IGZ0290S **Language structure to XML conversion could not complete in program** *PROGRAM-NAME* **because the content of numeric member** *MEMBER-NAME* **of language structure** *STRUCTURE-NAME* **is invalid.**

Explanation: Language structure to XML conversion has determined that the contents of the storage occupied by a numeric language structure member are invalid for the type.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Ensure that the numeric language structure member is properly initialized according to COBOL semantics.

Symbolic Feedback Code: IGZ092

IGZ0291S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because the maximum XML element nesting depth was exceeded. The error occurred at element** *ELEMENT-NAME* **with character content** *CHARACTER-CONTENT***.**

Explanation: The language structure to XML converter maintains an internal stack which represents the full qualification of the current element being processed in the XML document. If extraneous XML elements that are not described in the XML schema to which the converter is bound are present in the XML document, they will exceed the maximum supported element depth.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Either supply XML documents to the converter that validate against the bound XML schema or remove the extraneous element that causes the failure from the input XML document.

Symbolic Feedback Code: IGZ093

IGZ0292S **XML conversion could not complete in program** *PROGRAM-NAME* **because an attempt to register an exception handler failed with Language Environment error** *LE-ERROR***.**

Explanation: XML conversion uses Language Environment callable services to register a user-written handler to handle errors that occur during conversion. An attempt to register such a handler failed.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Ensure that Language Environment is configured properly and that no conditions exist in the chain of execution leading up to the converter that would prevent proper operation.

Symbolic Feedback Code: IGZ094

IGZ0293S **XML conversion could not complete in program** *PROGRAM-NAME* **because an attempt to unregister an exception handler failed with Language Environment error** *LE-ERROR***.**

Explanation: XML conversion uses Language Environment callable services to register a user-written handler to handle errors that occur during conversion. An attempt to unregister such a handler failed.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Ensure that Language Environment is configured properly and that no conditions exist in the chain of execution leading up to the converter that would prevent proper operation.

Symbolic Feedback Code: IGZ095

IGZ0294S **XML conversion could not complete in program** *PROGRAM-NAME* **because the address of one or more required parameters to the main entry point did not have valid addresses.**

Explanation: Ensure that each parameter passed to the XML converter has a valid non-null storage address unless a particular parameter can be null as defined by the converter call interface. For example, passing a null address for the XML buffer address to the language structure to XML converter will cause the required size of the XML buffer to be returned instead of performing XML conversion.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Ensure that each parameter passed to the converter has a valid storage address unless a particular parameter can be null as defined by the converter call interface. For example, a null message address passed to the outbound converter will be interpreted as a request for the maximum outbound message size to be returned instead of performing a conversion.

Symbolic Feedback Code: IGZ096

IGZ0295S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because a response code of** *RESPONSE-CODE* **with reason code** *REASON-CODE* **was received from the bidirectional data conversion module** *MODULE-NAME* **while processing the XML element** *ELEMENT-NAME***.**

Explanation: XML to language structure conversion uses an external module to handle bidirectional data conversions.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Refer to the bidirectional data conversion module documentation for explanations of return codes.

Symbolic Feedback Code: IGZ097

IGZ0296S **Language structure to XML conversion could not complete in program** *PROGRAM-NAME* **because a return code of** *RETURN-CODE* **with reason code** *REASON-CODE* **was received from the bidirectional data conversion module** *MODULE-NAME* **while processing member** *MEMBER-NAME* **of language structure** *STRUCTURE-NAME***.**

Explanation: Language structure to XML conversion uses an external module to handle bidirectional data conversions.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Refer to the bidirectional data conversion module documentation for explanations of return codes.

Symbolic Feedback Code: IGZ098

IGZ0297S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because the content for attribute** *ATTRIBUTE-NAME* **had a length greater than or equal to INPUTLEN characters, which was too long for the converter to process.**

Explanation: XML to language structure conversion maintains a buffer for accumulating the contents of XML attributes. The maximum size of the buffer is based on attribute and namespace declarations in the XML Schema.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the XML instance document such that it conforms to the inbound XML Schema.

IGZ0298S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because the target namespace of element** *ELEMENT-NAME* **was NAMESPACE-ACTUAL which is inconsistent with the expected target namespace of NAMESPACE-EXPECTED.**

Explanation: XML to language structure conversion can optionally validate the namespace name of XML elements.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the XML instance document such that the XML elements have namespace names consistent with those specified in the XML Schema.

IGZ0299S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because the minimum count of language structure** *STRUCT-NAME* **was not met.**

Explanation: XML to language structure conversion detected that the count of a language structure does not meet the minimum count that was specified when the converter was generated.

System action: The converter will signal a Language Environment condition. But if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the XML document or replace the converter with one having a different minimum count for the language structure.

IGZ0300S **XML to language structure conversion could not complete in program** *PROGRAM-NAME* **because the maximum count of language structure** *STRUCT-NAME* **was exceeded.**

Explanation: XML to language structure conversion detected that the count of a language structure exceeds the maximum that was specified when the XML converter was generated.

System action: The converter will signal a Language Environment condition. But if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the XML document or replace the XML converter with one having a higher maximum.

IGZ0301S **Language structure to XML conversion could not complete in program** *PROGRAM-NAME* **because the total count of language structure instances received was** *ACTUAL-COUNT* **which is less than the minimum of** *MINIMUM-COUNT* **instances required by the message layout.**

Explanation: Language structure to XML conversion detected that the language structure buffer it received contained fewer language structure instances in total than is required by the message layout.

System action: The converter will signal a Language Environment condition. But if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the language structure buffer so that each language structure occurs at least as many times as is required per the message layout.

IGZ0302S **Language structure to XML conversion could not complete in program** *program-name* **because the length prefix of language structure** *struct-name* **specified a length of** *specified-length* **bytes, which is less than the structure's minimum length of** *minimum-length* **bytes.**

Explanation: Language structure to XML conversion detected that the length prefix of a language structure specified a length that is less than the minimum length of the structure. The minimum length of a language structure is computed as the sum of the bytes consumed by all subordinate fields. In the case of fields that specify OCCURS DEPENDING ON, the minimum number of occurrences is assumed.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correctly set the two-byte length prefix of each language structure in the language structure buffer.

Symbolic Feedback Code: IGZ09E

IGZ0303S **Language structure to XML conversion could not complete in program** *program-name* **because the total count of language structure instances received was** *actual-count* **which is greater than the maximum of** *maximum-count* **instances allowed by the message layout.**

Explanation: Language structure to XML conversion detected that the language structure buffer it received contained more language structure instances in total than is allowed by the message layout.

System action: The converter will either signal a Language Environment condition or, if the optional feedback code argument is provided, a condition token representing the condition will be placed in the argument.

Programmer response: Correct the language structure buffer so that it contains no more than the maximum number of instances of each language structure in the message layout.

Symbolic Feedback Code: IGZ09F