PRQA QA-C

This page was automatically generated and should not be edited.

The information on this page was provided by outside contributors and has not been verified by SEI CERT.

The table below can be re-ordered, by clicking column headers.

Tool Version: 9.7

Checker	Guideline
0	MSC15-C. Do not depend on undefined behavior
1	MSC14-C. Do not introduce unnecessary platform dependencies
037	MSC14-C. Do not introduce unnecessary platform dependencies
0160	MSC15-C. Do not depend on undefined behavior
0161	FIO47-C. Use valid format strings
0161	MSC15-C. Do not depend on undefined behavior
0162	FIO47-C. Use valid format strings
0162	MSC15-C. Do not depend on undefined behavior
0163	FIO47-C. Use valid format strings
0163	MSC15-C. Do not depend on undefined behavior
0164	FIO47-C. Use valid format strings
0164	MSC15-C. Do not depend on undefined behavior
0165	FIO47-C. Use valid format strings
0165	MSC15-C. Do not depend on undefined behavior
0166	FIO47-C. Use valid format strings
0166	MSC15-C. Do not depend on undefined behavior
0167	FIO47-C. Use valid format strings
0167	MSC15-C. Do not depend on undefined behavior
0168	FIO47-C. Use valid format strings
0168	MSC15-C. Do not depend on undefined behavior
0169	FIO47-C. Use valid format strings
0169	MSC15-C. Do not depend on undefined behavior
0170	FIO47-C. Use valid format strings
0170	MSC15-C. Do not depend on undefined behavior
0171	FIO47-C. Use valid format strings
0171	MSC15-C. Do not depend on undefined behavior
0172	FIO47-C. Use valid format strings
0172	MSC15-C. Do not depend on undefined behavior
0173	FIO47-C. Use valid format strings
0173	MSC15-C. Do not depend on undefined behavior
0174	FIO47-C. Use valid format strings
0174	MSC15-C. Do not depend on undefined behavior
0175	FIO47-C. Use valid format strings
0175	MSC15-C. Do not depend on undefined behavior
0176	FIO47-C. Use valid format strings

0176	MSC15-C. Do not depend on undefined behavior
0177	FIO47-C. Use valid format strings
0177	MSC15-C. Do not depend on undefined behavior
0178	FIO47-C. Use valid format strings
0178	MSC15-C. Do not depend on undefined behavior
0179	MSC15-C. Do not depend on undefined behavior
0179 (U)	DCL11-C. Understand the type issues associated with variadic functions
0179 [U]	FIO47-C. Use valid format strings
0180 [	FIO47-C. Use valid format strings
0184	DCL10-C. Maintain the contract between the writer and caller of variadic functions
0184	MSC15-C. Do not depend on undefined behavior
0184 (U)	DCL11-C. Understand the type issues associated with variadic functions
0184 [U]	FIO47-C. Use valid format strings
0185	FIO47-C. Use valid format strings
0185	DCL10-C. Maintain the contract between the writer and caller of variadic functions
0185	MSC15-C. Do not depend on undefined behavior
0185 (U)	DCL11-C. Understand the type issues associated with variadic functions
0186	MSC15-C. Do not depend on undefined behavior
0186 (U)	DCL11-C. Understand the type issues associated with variadic functions
0190	FIO47-C. Use valid format strings
0190	MSC15-C. Do not depend on undefined behavior
0190 (U)	DCL11-C. Understand the type issues associated with variadic functions
0191	FIO47-C. Use valid format strings
0191	MSC15-C. Do not depend on undefined behavior
0191 (U)	DCL11-C. Understand the type issues associated with variadic functions
0192	FIO47-C. Use valid format strings
0192	MSC15-C. Do not depend on undefined behavior
0192 (U)	DCL11-C. Understand the type issues associated with variadic functions
0193	FIO47-C. Use valid format strings
0193	MSC15-C. Do not depend on undefined behavior
0193 (U)	DCL11-C. Understand the type issues associated with variadic functions
0194	FIO47-C. Use valid format strings
0194	MSC15-C. Do not depend on undefined behavior
0194 (U)	DCL11-C. Understand the type issues associated with variadic functions
0195	FIO47-C. Use valid format strings
0195	MSC15-C. Do not depend on undefined behavior
0195 (U)	DCL11-C. Understand the type issues associated with variadic functions
0196	FIO47-C. Use valid format strings
0196	MSC15-C. Do not depend on undefined behavior
0196 (U)	DCL11-C. Understand the type issues associated with variadic functions
0197	FIO47-C. Use valid format strings
0197	MSC15-C. Do not depend on undefined behavior
0197 (U)	DCL11-C. Understand the type issues associated with variadic functions
0198	FIO47-C. Use valid format strings
0198	MSC15-C. Do not depend on undefined behavior
0198 (U)	DCL11-C. Understand the type issues associated with variadic functions
0199	FIO47-C. Use valid format strings

0199	MSC15-C. Do not depend on undefined behavior
0199 (U)	DCL11-C. Understand the type issues associated with variadic functions
0200	FIO47-C. Use valid format strings
0200	MSC15-C. Do not depend on undefined behavior
0200 (U)	DCL11-C. Understand the type issues associated with variadic functions
0201	FIO47-C. Use valid format strings
0201	MSC15-C. Do not depend on undefined behavior
0201 (U)	DCL11-C. Understand the type issues associated with variadic functions
0202	FIO47-C. Use valid format strings
0202	MSC14-C. Do not introduce unnecessary platform dependencies
203	MSC15-C. Do not depend on undefined behavior
0204	FIO47-C. Use valid format strings
0204	MSC15-C. Do not depend on undefined behavior
0206	FIO47-C. Use valid format strings
0206	MSC15-C. Do not depend on undefined behavior
0206 (U)	DCL11-C. Understand the type issues associated with variadic functions
0207	DCL11-C. Understand the type issues associated with variadic functions
0207	MSC15-C. Do not depend on undefined behavior
0208	DCL11-C. Understand the type issues associated with variadic functions
0208	MSC15-C. Do not depend on undefined behavior
0232	MSC40-C. Do not violate constraints
0233	MSC40-C. Do not violate constraints
0235	MSC15-C. Do not depend on undefined behavior
0240	MSC14-C. Do not introduce unnecessary platform dependencies
0241	MSC14-C. Do not introduce unnecessary platform dependencies
0242	MSC14-C. Do not introduce unnecessary platform dependencies
0243	MSC14-C. Do not introduce unnecessary platform dependencies
0244	MSC40-C. Do not violate constraints
0246	MSC14-C. Do not introduce unnecessary platform dependencies
0268	MSC40-C. Do not violate constraints
0275	MSC15-C. Do not depend on undefined behavior
0278	MSC40-C. Do not violate constraints
0284	MSC14-C. Do not introduce unnecessary platform dependencies
0285	MSC09-C. Character encoding: Use subset of ASCII for safety
0286	MSC09-C. Character encoding: Use subset of ASCII for safety
0287	MSC09-C. Character encoding: Use subset of ASCII for safety
0288	MSC09-C. Character encoding: Use subset of ASCII for safety
0289	MSC09-C. Character encoding: Use subset of ASCII for safety
0299	MSC09-C. Character encoding: Use subset of ASCII for safety
0301	MSC15-C. Do not depend on undefined behavior
0302	MSC15-C. Do not depend on undefined behavior
0303	INT36-C. Converting a pointer to integer or integer to pointer
0304	MSC15-C. Do not depend on undefined behavior
0305	INT36-C. Converting a pointer to integer or integer to pointer
0306	INT36-C. Converting a pointer to integer or integer to pointer
0307	MSC15-C. Do not depend on undefined behavior
0309	INT36-C. Converting a pointer to integer or integer to pointer

0309	MSC15-C. Do not depend on undefined behavior
0310	EXP39-C. Do not access a variable through a pointer of an incompatible type
0310	EXP11-C. Do not make assumptions regarding the layout of structures with bit-fields
0311	EXP05-C. Do not cast away a const qualification
0312	EXP32-C. Do not access a volatile object through a nonvolatile reference
0321	MSC40-C. Do not violate constraints
0322	MSC40-C. Do not violate constraints
0323	MSC15-C. Do not depend on undefined behavior
0324	INT36-C. Converting a pointer to integer or integer to pointer
0326	EXP36-C. Do not cast pointers into more strictly aligned pointer types
0326	INT36-C. Converting a pointer to integer or integer to pointer
0327	MSC15-C. Do not depend on undefined behavior
0337	MSC15-C. Do not depend on undefined behavior
0338	MSC40-C. Do not violate constraints
0339	DCL18-C. Do not begin integer constants with 0 when specifying a decimal value
0341	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0342	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0360	INT36-C. Converting a pointer to integer or integer to pointer
0361	INT36-C. Converting a pointer to integer or integer to pointer
0362	INT36-C. Converting a pointer to integer or integer to pointer
0400	MSC15-C. Do not depend on undefined behavior
0400 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0400 [U]	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
0401	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
0401	MSC15-C. Do not depend on undefined behavior
0401 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0402	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
0402	MSC15-C. Do not depend on undefined behavior
0402 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0403	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
0403	MSC15-C. Do not depend on undefined behavior
0403 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0403 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0403 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0404	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
0404 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0405	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
0405 [U]	EXP30-C. Do not depend on the order of evaluation for side effects
0422	MSC40-C. Do not violate constraints
0423	MSC40-C. Do not violate constraints
0426	MSC40-C. Do not violate constraints
0427	MSC40-C. Do not violate constraints
0428	EXP16-C. Do not compare function pointers to constant values
0429	MSC40-C. Do not violate constraints
0430	MSC40-C. Do not violate constraints
0431	MSC40-C. Do not violate constraints
0431	EXP05-C. Do not cast away a const qualification

0431(C)	DCL13-C. Declare function parameters that are pointers to values not changed by the function as const
0432	STR38-C. Do not confuse narrow and wide character strings and functions
0432	MSC40-C. Do not violate constraints
0432 [C]	STR04-C. Use plain char for characters in the basic character set
0434 (C)	DCL31-C. Declare identifiers before using them
0435	MSC40-C. Do not violate constraints
0436	MSC40-C. Do not violate constraints
0437	MSC40-C. Do not violate constraints
0446	MSC40-C. Do not violate constraints
0447	MSC40-C. Do not violate constraints
0448	MSC40-C. Do not violate constraints
0449	MSC40-C. Do not violate constraints
0450 [U]	EXP35-C. Do not modify objects with temporary lifetime
0451	MSC40-C. Do not violate constraints
0452	MSC40-C. Do not violate constraints
0453	MSC40-C. Do not violate constraints
0454	MSC40-C. Do not violate constraints
0455 [U]	EXP35-C. Do not modify objects with temporary lifetime
0456	MSC40-C. Do not violate constraints
0457	MSC40-C. Do not violate constraints
0458	MSC40-C. Do not violate constraints
0459 [U]	EXP35-C. Do not modify objects with temporary lifetime
0460	MSC40-C. Do not violate constraints
0461	MSC40-C. Do not violate constraints
0462	MSC40-C. Do not violate constraints
0463	MSC40-C. Do not violate constraints
0465 [U]	EXP35-C. Do not modify objects with temporary lifetime
0465 [U]	EXP35-C. Do not modify objects with temporary lifetime
0466	MSC40-C. Do not violate constraints
0467	MSC40-C. Do not violate constraints
0468	MSC40-C. Do not violate constraints
0469	MSC40-C. Do not violate constraints
0475	MSC15-C. Do not depend on undefined behavior
0476	MSC40-C. Do not violate constraints
0477	MSC40-C. Do not violate constraints
0478	MSC40-C. Do not violate constraints
0481	MSC40-C. Do not violate constraints
0482	MSC40-C. Do not violate constraints
0483	MSC40-C. Do not violate constraints
0484	MSC40-C. Do not violate constraints
0485	MSC40-C. Do not violate constraints
0486	MSC40-C. Do not violate constraints
0487	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
0487	MSC40-C. Do not violate constraints
0488	EXP08-C. Ensure pointer arithmetic is used correctly
0493	MSC40-C. Do not violate constraints
0494	MSC40-C. Do not violate constraints

0495	MSC40-C. Do not violate constraints
0496	MSC40-C. Do not violate constraints
0499	INT34-C. Do not shift an expression by a negative number of bits or by greater than or equal to the number of bits that exist in the operand
0513	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
0513	MSC40-C. Do not violate constraints
0514	MSC40-C. Do not violate constraints
0515	MSC40-C. Do not violate constraints
0536	MSC40-C. Do not violate constraints
0537	MSC40-C. Do not violate constraints
0540	MSC40-C. Do not violate constraints
0541	MSC40-C. Do not violate constraints
0542	MSC40-C. Do not violate constraints
0543	MSC15-C. Do not depend on undefined behavior
0544	MSC15-C. Do not depend on undefined behavior
0545	MSC15-C. Do not depend on undefined behavior
0546	MSC40-C. Do not violate constraints
0547	MSC40-C. Do not violate constraints
0550	MSC40-C. Do not violate constraints
0551	MSC14-C. Do not introduce unnecessary platform dependencies
0554	MSC40-C. Do not violate constraints
0555	MSC40-C. Do not violate constraints
0556	STR30-C. Do not attempt to modify string literals
0556	MSC40-C. Do not violate constraints
0557	MSC40-C. Do not violate constraints
0558	MSC40-C. Do not violate constraints
0559	MSC40-C. Do not violate constraints
0560	MSC40-C. Do not violate constraints
0561	MSC40-C. Do not violate constraints
0562	MSC40-C. Do not violate constraints
562	EXP32-C. Do not access a volatile object through a nonvolatile reference
0563	EXP40-C. Do not modify constant objects
0563	MSC40-C. Do not violate constraints
563	EXP32-C. Do not access a volatile object through a nonvolatile reference
0564	MSC40-C. Do not violate constraints
0565	MSC40-C. Do not violate constraints
0580	MSC40-C. Do not violate constraints
0581	MSC14-C. Do not introduce unnecessary platform dependencies
0588	MSC40-C. Do not violate constraints
0589	MSC40-C. Do not violate constraints
0590	MSC40-C. Do not violate constraints
0591	MSC40-C. Do not violate constraints
0601	MSC14-C. Do not introduce unnecessary platform dependencies
0602	DCL37-C. Do not declare or define a reserved identifier
0602	MSC15-C. Do not depend on undefined behavior
0603	DCL37-C. Do not declare or define a reserved identifier
0603	MSC15-C. Do not depend on undefined behavior
0605	MSC40-C. Do not violate constraints

0616	MSC40-C. Do not violate constraints
0619	MSC40-C. Do not violate constraints
0620	MSC40-C. Do not violate constraints
0621	MSC40-C. Do not violate constraints
0622	MSC40-C. Do not violate constraints
0623	MSC15-C. Do not depend on undefined behavior
0625	MSC15-C. Do not depend on undefined behavior
0625 (U)	DCL36-C. Do not declare an identifier with conflicting linkage classifications
0626	MSC15-C. Do not depend on undefined behavior
0627	MSC40-C. Do not violate constraints
0627	DCL23-C. Guarantee that mutually visible identifiers are unique
0628	MSC40-C. Do not violate constraints
0629	MSC40-C. Do not violate constraints
0630	MSC15-C. Do not depend on undefined behavior
0631	MSC40-C. Do not violate constraints
0632	MSC15-C. Do not depend on undefined behavior
0633	MSC14-C. Do not introduce unnecessary platform dependencies
0634	MSC14-C. Do not introduce unnecessary platform dependencies
0634 (I)	INT12-C. Do not make assumptions about the type of a plain int bit-field when used in an expression
0635	INT12-C. Do not make assumptions about the type of a plain int bit-field when used in an expression
0635	MSC14-C. Do not introduce unnecessary platform dependencies
0636	MSC15-C. Do not depend on undefined behavior
0638	MSC40-C. Do not violate constraints
0640	MSC40-C. Do not violate constraints
0641	MSC40-C. Do not violate constraints
0642	MSC40-C. Do not violate constraints
0643	MSC40-C. Do not violate constraints
0644	MSC40-C. Do not violate constraints
0645	MSC40-C. Do not violate constraints
0646	MSC40-C. Do not violate constraints
0649	MSC40-C. Do not violate constraints
0650	MSC40-C. Do not violate constraints
0651	MSC40-C. Do not violate constraints
0653	MSC40-C. Do not violate constraints
0654	MSC15-C. Do not depend on undefined behavior
0655	MSC40-C. Do not violate constraints
0656	MSC40-C. Do not violate constraints
0657	MSC40-C. Do not violate constraints
0658	MSC15-C. Do not depend on undefined behavior
0659	MSC40-C. Do not violate constraints
0660	MSC14-C. Do not introduce unnecessary platform dependencies
0661	MSC15-C. Do not depend on undefined behavior
0662	MSC14-C. Do not introduce unnecessary platform dependencies
0664	MSC40-C. Do not violate constraints
0665	MSC40-C. Do not violate constraints
0667	MSC15-C. Do not depend on undefined behavior
0668	MSC15-C. Do not depend on undefined behavior

0669	MSC40-C. Do not violate constraints
0671	MSC40-C. Do not violate constraints
0672	MSC15-C. Do not depend on undefined behavior
0673	MSC40-C. Do not violate constraints
673	EXP32-C. Do not access a volatile object through a nonvolatile reference
0674	MSC40-C. Do not violate constraints
0674	STR04-C. Use plain char for characters in the basic character set
674	EXP32-C. Do not access a volatile object through a nonvolatile reference
0675	MSC40-C. Do not violate constraints
0676	MSC15-C. Do not depend on undefined behavior
0677	MSC40-C. Do not violate constraints
0678	ARR02-C. Explicitly specify array bounds, even if implicitly defined by an initializer
0678	MSC15-C. Do not depend on undefined behavior
0680	MSC15-C. Do not depend on undefined behavior
0682	MSC40-C. Do not violate constraints
0683	MSC40-C. Do not violate constraints
0684	MSC40-C. Do not violate constraints
0685	MSC40-C. Do not violate constraints
0688	ARR02-C. Explicitly specify array bounds, even if implicitly defined by an initializer
0690	MSC40-C. Do not violate constraints
0695	MEM02-C. Immediately cast the result of a memory allocation function call into a pointer to the allocated type
0696	MEM35-C. Allocate sufficient memory for an object
0697	EXP03-C. Do not assume the size of a structure is the sum of the sizes of its members
0698	MSC40-C. Do not violate constraints
0699	MSC40-C. Do not violate constraints
0699	STR04-C. Use plain char for characters in the basic character set
0701	MEM35-C. Allocate sufficient memory for an object
0706	MSC15-C. Do not depend on undefined behavior
0708	MSC40-C. Do not violate constraints
0709	MSC40-C. Do not violate constraints
0724	INT09-C. Ensure enumeration constants map to unique values
0736	MSC40-C. Do not violate constraints
0737	MSC40-C. Do not violate constraints
0738	MSC40-C. Do not violate constraints
0745	MSC15-C. Do not depend on undefined behavior
0746	MSC40-C. Do not violate constraints
0747	MSC40-C. Do not violate constraints
0751	EXP39-C. Do not access a variable through a pointer of an incompatible type
0751	EXP11-C. Do not make assumptions regarding the layout of structures with bit-fields
0752	STR30-C. Do not attempt to modify string literals
0752	STR05-C. Use pointers to const when referring to string literals
0753	STR30-C. Do not attempt to modify string literals
0753	STR05-C. Use pointers to const when referring to string literals
0754	STR30-C. Do not attempt to modify string literals
0755	MSC40-C. Do not violate constraints
0756	MSC40-C. Do not violate constraints
0757	MSC40-C. Do not violate constraints

0758	MSC40-C. Do not violate constraints
0766	MSC40-C. Do not violate constraints
0767	MSC40-C. Do not violate constraints
0768	MSC40-C. Do not violate constraints
0774	MSC40-C. Do not violate constraints
0775	MSC40-C. Do not violate constraints
0776	DCL40-C. Do not create incompatible declarations of the same function or object
0776	DCL23-C. Guarantee that mutually visible identifiers are unique
0777	DCL23-C. Guarantee that mutually visible identifiers are unique
0777	MSC15-C. Do not depend on undefined behavior
0778	DCL40-C. Do not create incompatible declarations of the same function or object
0778	DCL23-C. Guarantee that mutually visible identifiers are unique
0779	DCL40-C. Do not create incompatible declarations of the same function or object
0779	DCL23-C. Guarantee that mutually visible identifiers are unique
0779	MSC15-C. Do not depend on undefined behavior
0789	DCL40-C. Do not create incompatible declarations of the same function or object
0789	DCL23-C. Guarantee that mutually visible identifiers are unique
0790	FLP02-C. Avoid using floating-point numbers when precise computation is needed
0791	DCL23-C. Guarantee that mutually visible identifiers are unique
0793	DCL23-C. Guarantee that mutually visible identifiers are unique
0795	DCL01-C. Do not reuse variable names in subscopes
0796	DCL01-C. Do not reuse variable names in subscopes
0801	MSC40-C. Do not violate constraints
0801	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0802	MSC40-C. Do not violate constraints
0802	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0803	MSC40-C. Do not violate constraints
803	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0804	MSC40-C. Do not violate constraints
0811	MSC40-C. Do not violate constraints
0811	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0812	MSC40-C. Do not violate constraints
0813	MSC15-C. Do not depend on undefined behavior
0814	MSC15-C. Do not depend on undefined behavior
0821	MSC40-C. Do not violate constraints
0821	MSC15-C. Do not depend on undefined behavior
0830	MSC14-C. Do not introduce unnecessary platform dependencies
0831	MSC14-C. Do not introduce unnecessary platform dependencies
0834	MSC40-C. Do not violate constraints
0835	MSC40-C. Do not violate constraints
0836	MSC15-C. Do not depend on undefined behavior
0837	MSC15-C. Do not depend on undefined behavior
0840	MSC14-C. Do not introduce unnecessary platform dependencies
0844	MSC40-C. Do not violate constraints
0845	MSC40-C. Do not violate constraints
0848	MSC15-C. Do not depend on undefined behavior
0851	MSC40-C. Do not violate constraints

0852	MSC40-C. Do not violate constraints
0853	PRE32-C. Do not use preprocessor directives in invocations of function-like macros
0853	MSC15-C. Do not depend on undefined behavior
0854	MSC15-C. Do not depend on undefined behavior
0864	MSC15-C. Do not depend on undefined behavior
0865	MSC15-C. Do not depend on undefined behavior
0866	MSC40-C. Do not violate constraints
0867	MSC15-C. Do not depend on undefined behavior
0872	MSC15-C. Do not depend on undefined behavior
0872	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0873	MSC40-C. Do not violate constraints
0874	MSC15-C. Do not depend on undefined behavior
0874	STR10-C. Do not concatenate different type of string literals
0877	MSC40-C. Do not violate constraints
0880	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0881	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0883	MSC14-C. Do not introduce unnecessary platform dependencies
0883	PRE06-C. Enclose header files in an include guard
0884	PRE05-C. Understand macro replacement when concatenating tokens or performing stringification
0885	MSC15-C. Do not depend on undefined behavior
0887	MSC15-C. Do not depend on undefined behavior
0888	MSC15-C. Do not depend on undefined behavior
0899	MSC14-C. Do not introduce unnecessary platform dependencies
0905	PRE30-C. Do not create a universal character name through concatenation
0914	MSC15-C. Do not depend on undefined behavior
0915	MSC15-C. Do not depend on undefined behavior
0940	MSC40-C. Do not violate constraints
0941	MSC40-C. Do not violate constraints
0942	MSC15-C. Do not depend on undefined behavior
0943	MSC40-C. Do not violate constraints
0944	MSC40-C. Do not violate constraints
1001	MSC14-C. Do not introduce unnecessary platform dependencies
1002	MSC14-C. Do not introduce unnecessary platform dependencies
1003	MSC14-C. Do not introduce unnecessary platform dependencies
1006	MSC14-C. Do not introduce unnecessary platform dependencies
1008	MSC14-C. Do not introduce unnecessary platform dependencies
1012	MSC14-C. Do not introduce unnecessary platform dependencies
1014	MSC14-C. Do not introduce unnecessary platform dependencies
1015	MSC14-C. Do not introduce unnecessary platform dependencies
1019	MSC14-C. Do not introduce unnecessary platform dependencies
1020	MSC14-C. Do not introduce unnecessary platform dependencies
1021	MSC14-C. Do not introduce unnecessary platform dependencies
1022	MSC14-C. Do not introduce unnecessary platform dependencies
1023	MSC40-C. Do not violate constraints
1024	MSC40-C. Do not violate constraints
1025	MSC40-C. Do not violate constraints
1026	MSC14-C. Do not introduce unnecessary platform dependencies

1028	MSC14-C. Do not introduce unnecessary platform dependencies
1029	MSC14-C. Do not introduce unnecessary platform dependencies
1033	MSC40-C. Do not violate constraints
1034	MSC14-C. Do not introduce unnecessary platform dependencies
1035	MSC14-C. Do not introduce unnecessary platform dependencies
1036	MSC14-C. Do not introduce unnecessary platform dependencies
1037 1039	DCL38-C. Use the correct syntax when declaring a flexible array member
1038	MSC14-C. Do not introduce unnecessary platform dependencies
1041	MSC14-C. Do not introduce unnecessary platform dependencies
1042	MSC14-C. Do not introduce unnecessary platform dependencies
1043	MSC14-C. Do not introduce unnecessary platform dependencies
1044	MSC14-C. Do not introduce unnecessary platform dependencies
1045	MSC14-C. Do not introduce unnecessary platform dependencies
1046	MSC14-C. Do not introduce unnecessary platform dependencies
1047	MSC40-C. Do not violate constraints
1048	MSC40-C. Do not violate constraints
1050	MSC40-C. Do not violate constraints
1051	ARR32-C. Ensure size arguments for variable length arrays are in a valid range
1051	MEM05-C. Avoid large stack allocations
1054	DCL21-C. Understand the storage of compound literals
1057	EXP43-C. Avoid undefined behavior when using restrict-qualified pointers
1061	MEM33-C. Allocate and copy structures containing a flexible array member dynamically
1061	MSC40-C. Do not violate constraints
1062	MEM33-C. Allocate and copy structures containing a flexible array member dynamically
1062	MSC40-C. Do not violate constraints
1063	MEM33-C. Allocate and copy structures containing a flexible array member dynamically
1064	MEM33-C. Allocate and copy structures containing a flexible array member dynamically
1069	MEM35-C. Allocate sufficient memory for an object
1071	MEM35-C. Allocate sufficient memory for an object
1073	MEM35-C. Allocate sufficient memory for an object
1114	CON40-C. Do not refer to an atomic variable twice in an expression
1115	CON40-C. Do not refer to an atomic variable twice in an expression
1116	CON40-C. Do not refer to an atomic variable twice in an expression
1250	INT02-C. Understand integer conversion rules
1251	INT02-C. Understand integer conversion rules
1252	INT02-C. Understand integer conversion rules
1253	INT02-C. Understand integer conversion rules
1256	INT02-C. Understand integer conversion rules
1257	INT02-C. Understand integer conversion rules
1260	FLP36-C. Preserve precision when converting integral values to floating-point type
1260	INT02-C. Understand integer conversion rules
1263	FLP36-C. Preserve precision when converting integral values to floating-point type
1263	INT02-C. Understand integer conversion rules
1266	INT02-C. Understand integer conversion rules
1272	DCL18-C. Do not begin integer constants with 0 when specifying a decimal value
1274	INT02-C. Understand integer conversion rules
1280	DCL16-C. Use "L," not "l," to indicate a long value

1290	INT02-C. Understand integer conversion rules
1291	INT02-C. Understand integer conversion rules
1292	INT02-C. Understand integer conversion rules
1292	INT07-C. Use only explicitly signed or unsigned char type for numeric values
1293	INT02-C. Understand integer conversion rules
1293	INT07-C. Use only explicitly signed or unsigned char type for numeric values
1294	INT02-C. Understand integer conversion rules
1295	INT02-C. Understand integer conversion rules
1296	INT02-C. Understand integer conversion rules
1297	INT02-C. Understand integer conversion rules
1298	FLP36-C. Preserve precision when converting integral values to floating-point type
1298	INT02-C. Understand integer conversion rules
1299	FLP36-C. Preserve precision when converting integral values to floating-point type
1299	INT02-C. Understand integer conversion rules
1302	DCL31-C. Declare identifiers before using them
1304	DCL07-C. Include the appropriate type information in function declarators
1312	STR11-C. Do not specify the bound of a character array initialized with a string literal
1331	EXP37-C. Call functions with the correct number and type of arguments
1332	EXP37-C. Call functions with the correct number and type of arguments
1333	EXP37-C. Call functions with the correct number and type of arguments
1434	MSC14-C. Do not introduce unnecessary platform dependencies
1485	FIO38-C. Do not copy a FILE object
1488	EXP42-C. Do not compare padding data
1492	ENV30-C. Do not modify the object referenced by the return value of certain functions
1493	ENV30-C. Do not modify the object referenced by the return value of certain functions
1494	ENV30-C. Do not modify the object referenced by the return value of certain functions
1500	MSC13-C. Detect and remove unused values
1501	MSC07-C. Detect and remove dead code
1502	MSC13-C. Detect and remove unused values
1503	MSC07-C. Detect and remove dead code
1504	DCL15-C. Declare file-scope objects or functions that do not need external linkage as static
1504	DCL19-C. Minimize the scope of variables and functions
1505	DCL19-C. Minimize the scope of variables and functions
1509	MSC15-C. Do not depend on undefined behavior
1510	DCL40-C. Do not create incompatible declarations of the same function or object
1510	MSC15-C. Do not depend on undefined behavior
1520	MEM05-C. Avoid large stack allocations
1531	DCL15-C. Declare file-scope objects or functions that do not need external linkage as static
1531	DCL19-C. Minimize the scope of variables and functions
1532	DCL19-C. Minimize the scope of variables and functions
1800	FLP36-C. Preserve precision when converting integral values to floating-point type
1800	INT02-C. Understand integer conversion rules
1802	FLP36-C. Preserve precision when converting integral values to floating-point type
1802	INT02-C. Understand integer conversion rules
1803	FLP36-C. Preserve precision when converting integral values to floating-point type
1803	INT02-C. Understand integer conversion rules
1804	FLP36-C. Preserve precision when converting integral values to floating-point type

1804	INT02-C. Understand integer conversion rules
1810	INT02-C. Understand integer conversion rules
1811	INT02-C. Understand integer conversion rules
1812	INT02-C. Understand integer conversion rules
1813	INT02-C. Understand integer conversion rules
1820	INT02-C. Understand integer conversion rules
1821	INT02-C. Understand integer conversion rules
1822	INT02-C. Understand integer conversion rules
1823	INT02-C. Understand integer conversion rules
1824	INT02-C. Understand integer conversion rules
1830	INT02-C. Understand integer conversion rules
1831	INT02-C. Understand integer conversion rules
1832	INT02-C. Understand integer conversion rules
1833	INT02-C. Understand integer conversion rules
1834	INT02-C. Understand integer conversion rules
1840	INT02-C. Understand integer conversion rules
1841	INT02-C. Understand integer conversion rules
1842	INT02-C. Understand integer conversion rules
1843	INT02-C. Understand integer conversion rules
1844	INT02-C. Understand integer conversion rules
1850	INT02-C. Understand integer conversion rules
1851	INT02-C. Understand integer conversion rules
1852	INT02-C. Understand integer conversion rules
1853	INT02-C. Understand integer conversion rules
1854	INT02-C. Understand integer conversion rules
1860	INT02-C. Understand integer conversion rules
1861	INT02-C. Understand integer conversion rules
1862	INT02-C. Understand integer conversion rules
1863	INT02-C. Understand integer conversion rules
1864	INT02-C. Understand integer conversion rules
1880	INT02-C. Understand integer conversion rules
1881	INT02-C. Understand integer conversion rules
1882	INT02-C. Understand integer conversion rules
1890	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
1891	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
1892	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
1893	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
1894	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
1895	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
2000	MSC01-C. Strive for logical completeness
2002	MSC01-C. Strive for logical completeness
2003	MSC17-C. Finish every set of statements associated with a case label with a break statement
2004	MSC01-C. Strive for logical completeness
2008	DCL41-C. Do not declare variables inside a switch statement before the first case label
2008	MSC07-C. Detect and remove dead code
2019	MSC20-C. Do not use a switch statement to transfer control into a complex block
2026	CON41-C. Wrap functions that can fail spuriously in a loop

2027	CON36-C. Wrap functions that can spuriously wake up in a loop
2028	SIG30-C. Call only asynchronous-safe functions within signal handlers
2029	SIG31-C. Do not access shared objects in signal handlers
2030	SIG30-C. Call only asynchronous-safe functions within signal handlers
2030	SIG31-C. Do not access shared objects in signal handlers
2031	ERR32-C. Do not rely on indeterminate values of errno
2050	DCL31-C. Declare identifiers before using them
2050	DCL07-C. Include the appropriate type information in function declarators
2051	DCL31-C. Declare identifiers before using them
2052	ARR32-C. Ensure size arguments for variable length arrays are in a valid range
2052	MEM05-C. Avoid large stack allocations
2100	INT02-C. Understand integer conversion rules
2101	INT02-C. Understand integer conversion rules
2102	INT02-C. Understand integer conversion rules
2103	INT02-C. Understand integer conversion rules
2104	INT02-C. Understand integer conversion rules
2105	INT02-C. Understand integer conversion rules
2106	INT02-C. Understand integer conversion rules
2106	STR09-C. Don't assume numeric values for expressions with type plain character
2107	INT02-C. Understand integer conversion rules
2107	STR09-C. Don't assume numeric values for expressions with type plain character
2109	INT02-C. Understand integer conversion rules
2110	INT02-C. Understand integer conversion rules
2111	INT02-C. Understand integer conversion rules
2112	INT02-C. Understand integer conversion rules
2113	INT02-C. Understand integer conversion rules
2114	INT02-C. Understand integer conversion rules
2115	INT02-C. Understand integer conversion rules
2116	INT02-C. Understand integer conversion rules
2117	INT02-C. Understand integer conversion rules
2118	INT02-C. Understand integer conversion rules
2119	INT02-C. Understand integer conversion rules
2120	INT02-C. Understand integer conversion rules
2122	INT02-C. Understand integer conversion rules
2124	INT02-C. Understand integer conversion rules
2130	INT02-C. Understand integer conversion rules
2132	INT02-C. Understand integer conversion rules
2134	INT02-C. Understand integer conversion rules
2140	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2141	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2143	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2144	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2145	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2147	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2148	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2149	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2151	STR34-C. Cast characters to unsigned char before converting to larger integer sizes

2152	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2153	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2155	STR34-C. Cast characters to unsigned char before converting to larger integer sizes
2212	EXP19-C. Use braces for the body of an if, for, or while statement
2500	ERR30-C. Set errno to zero before calling a library function known to set errno, and check errno only after the function returns a value indicating failure
2501	ERR30-C. Set errno to zero before calling a library function known to set errno, and check errno only after the function returns a value indicating failure
2502	ERR30-C. Set errno to zero before calling a library function known to set errno, and check errno only after the function returns a value indicating failure
2503	ERR30-C. Set errno to zero before calling a library function known to set errno, and check errno only after the function returns a value indicating failure
2547	DCL01-C. Do not reuse variable names in subscopes
2668	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2669	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2676	FIO34-C. Distinguish between characters read from a file and EOF or WEOF
2678	FIO34-C. Distinguish between characters read from a file and EOF or WEOF
2681	ENV34-C. Do not store pointers returned by certain functions
2682	ENV34-C. Do not store pointers returned by certain functions
2683	ENV34-C. Do not store pointers returned by certain functions
2696	FIO46-C. Do not access a closed file
2697	FIO46-C. Do not access a closed file
2698	FIO46-C. Do not access a closed file
2701	FIO42-C. Close files when they are no longer needed
2702	FIO42-C. Close files when they are no longer needed
2703	FIO42-C. Close files when they are no longer needed
2706	MEM31-C. Free dynamically allocated memory when no longer needed
2707	MEM31-C. Free dynamically allocated memory when no longer needed
2708	MEM31-C. Free dynamically allocated memory when no longer needed
2721	MEM34-C. Only free memory allocated dynamically
2722	MEM34-C. Only free memory allocated dynamically
2723	MEM34-C. Only free memory allocated dynamically
2726	EXP33-C. Do not read uninitialized memory
2727	EXP33-C. Do not read uninitialized memory
2728	EXP33-C. Do not read uninitialized memory
2761	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2762	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2763	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2766	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2767	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2768	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2771	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2772	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2773	ARR36-C. Do not subtract or compare two pointers that do not refer to the same array
2790 [C]	INT34-C. Do not shift an expression by a negative number of bits or by greater than or equal to the number of bits that exist in the operand
2791 [D]	INT34-C. Do not shift an expression by a negative number of bits or by greater than or equal to the number of bits that exist in the operand
2792 [A]	INT34-C. Do not shift an expression by a negative number of bits or by greater than or equal to the number of bits that exist in the operand
2793 [S]	INT34-C. Do not shift an expression by a negative number of bits or by greater than or equal to the number of bits that exist in the operand
2800	INT32-C. Ensure that operations on signed integers do not result in overflow
2800	INT08-C. Verify that all integer values are in range
2801	INT32-C. Ensure that operations on signed integers do not result in overflow

2801	INT08-C. Verify that all integer values are in range
2802	INT32-C. Ensure that operations on signed integers do not result in overflow
2802	INT08-C. Verify that all integer values are in range
2803	INT32-C. Ensure that operations on signed integers do not result in overflow
2803	INT08-C. Verify that all integer values are in range
2810	EXP34-C. Do not dereference null pointers
2811	EXP34-C. Do not dereference null pointers
2812	EXP34-C. Do not dereference null pointers
2813	EXP34-C. Do not dereference null pointers
2814	EXP34-C. Do not dereference null pointers
2820	EXP34-C. Do not dereference null pointers
2821	EXP34-C. Do not dereference null pointers
2822	EXP34-C. Do not dereference null pointers
2823	EXP34-C. Do not dereference null pointers
2824	EXP34-C. Do not dereference null pointers
2830 [C]	INT33-C. Ensure that division and remainder operations do not result in divide-by-zero errors
2831 [D]	INT33-C. Ensure that division and remainder operations do not result in divide-by-zero errors
2832 [A]	INT33-C. Ensure that division and remainder operations do not result in divide-by-zero errors
2833 [S]	INT33-C. Ensure that division and remainder operations do not result in divide-by-zero errors
2834 [P]	INT33-C. Ensure that division and remainder operations do not result in divide-by-zero errors
2835	STR32-C. Do not pass a non-null-terminated character sequence to a library function that expects a string
2836	STR32-C. Do not pass a non-null-terminated character sequence to a library function that expects a string
2839	STR32-C. Do not pass a non-null-terminated character sequence to a library function that expects a string
2840	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2841	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2842	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2843	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2844	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2845	ARR38-C. Guarantee that library functions do not form invalid pointers
2845	STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator
2846	ARR38-C. Guarantee that library functions do not form invalid pointers
2846	STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator
2847	ARR38-C. Guarantee that library functions do not form invalid pointers
2847	STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator
2848	ARR38-C. Guarantee that library functions do not form invalid pointers
2848	STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator
2849	ARR38-C. Guarantee that library functions do not form invalid pointers
2849	STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator
2850	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2851	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2852	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2853	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2855	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2856	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2857	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2858	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2860	INT32-C. Ensure that operations on signed integers do not result in overflow

2861	INT32-C. Ensure that operations on signed integers do not result in overflow
2862	INT32-C. Ensure that operations on signed integers do not result in overflow
2863	INT32-C. Ensure that operations on signed integers do not result in overflow
2877	MSC07-C. Detect and remove dead code
2880	MSC07-C. Detect and remove dead code
2881	MSC07-C. Detect and remove dead code
2882	DCL41-C. Do not declare variables inside a switch statement before the first case label
2882	MSC07-C. Detect and remove dead code
2883	MSC07-C. Detect and remove dead code
2888	MSC37-C. Ensure that control never reaches the end of a non-void function
2890	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2891	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2892	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2893	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2895	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2896	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2897	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2898	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2900	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2901	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2902	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2903	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2905	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2906	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2907	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2908	INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
2910	INT08-C. Verify that all integer values are in range
2910 [C]	INT30-C. Ensure that unsigned integer operations do not wrap
2911	INT08-C. Verify that all integer values are in range
2911 [D]	INT30-C. Ensure that unsigned integer operations do not wrap
2912	INT08-C. Verify that all integer values are in range
2912 [A]	INT30-C. Ensure that unsigned integer operations do not wrap
2913	INT08-C. Verify that all integer values are in range
2913 [S]	INT30-C. Ensure that unsigned integer operations do not wrap
2930	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2930	ARR37-C. Do not add or subtract an integer to a pointer to a non-array object
2930	ARR38-C. Guarantee that library functions do not form invalid pointers
2930	EXP08-C. Ensure pointer arithmetic is used correctly
2931	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2931	ARR37-C. Do not add or subtract an integer to a pointer to a non-array object
2931	EXP08-C. Ensure pointer arithmetic is used correctly
2932	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2932	ARR37-C. Do not add or subtract an integer to a pointer to a non-array object
2932	ARR38-C. Guarantee that library functions do not form invalid pointers
2932	EXP08-C. Ensure pointer arithmetic is used correctly
2933	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2933	ARR37-C. Do not add or subtract an integer to a pointer to a non-array object

2933	ARR38-C. Guarantee that library functions do not form invalid pointers
2933	EXP08-C. Ensure pointer arithmetic is used correctly
2934	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2934	ARR37-C. Do not add or subtract an integer to a pointer to a non-array object
2934	ARR38-C. Guarantee that library functions do not form invalid pointers
2934	EXP08-C. Ensure pointer arithmetic is used correctly
2935	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2936	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2937	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2938	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2939	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2940	INT16-C. Do not make assumptions about representation of signed integers
2941	INT16-C. Do not make assumptions about representation of signed integers
2942	INT16-C. Do not make assumptions about representation of signed integers
2943	INT16-C. Do not make assumptions about representation of signed integers
2945	INT16-C. Do not make assumptions about representation of signed integers
2946	INT16-C. Do not make assumptions about representation of signed integers
2947	INT16-C. Do not make assumptions about representation of signed integers
2948	INT16-C. Do not make assumptions about representation of signed integers
2950	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2951	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2952	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2953	ARR30-C. Do not form or use out-of-bounds pointers or array subscripts
2961	EXP33-C. Do not read uninitialized memory
2962	EXP33-C. Do not read uninitialized memory
2963	EXP33-C. Do not read uninitialized memory
2966	EXP33-C. Do not read uninitialized memory
2967	EXP33-C. Do not read uninitialized memory
2968	EXP33-C. Do not read uninitialized memory
2971	EXP33-C. Do not read uninitialized memory
2972	EXP33-C. Do not read uninitialized memory
2973	EXP33-C. Do not read uninitialized memory
2976	EXP33-C. Do not read uninitialized memory
2977	EXP33-C. Do not read uninitialized memory
2978	EXP33-C. Do not read uninitialized memory
2980	MSC07-C. Detect and remove dead code
2980	MSC13-C. Detect and remove unused values
2981	MSC07-C. Detect and remove dead code
2981	MSC13-C. Detect and remove unused values
2982	MSC07-C. Detect and remove dead code
2982	MSC13-C. Detect and remove unused values
2983	MSC07-C. Detect and remove dead code
2983	MSC13-C. Detect and remove unused values
2984	MSC07-C. Detect and remove dead code
2984	MSC13-C. Detect and remove unused values
2985	MSC07-C. Detect and remove dead code
2985	MSC13-C. Detect and remove unused values

2986	MSC07-C. Detect and remove dead code
2986	MSC13-C. Detect and remove unused values
3001	DCL20-C. Explicitly specify void when a function accepts no arguments
3002	EXP37-C. Call functions with the correct number and type of arguments
3004	EXP16-C. Do not compare function pointers to constant values
3007	DCL20-C. Explicitly specify void when a function accepts no arguments
3103	INT10-C. Do not assume a positive remainder when using the % operator
3108	MSC04-C. Use comments consistently and in a readable fashion
3109	EXP15-C. Do not place a semicolon on the same line as an if, for, or while statement
3110	MSC12-C. Detect and remove code that has no effect or is never executed
3112	MSC12-C. Detect and remove code that has no effect or is never executed
3113	MSC15-C. Do not depend on undefined behavior
3114	MSC15-C. Do not depend on undefined behavior
3120	DCL06-C. Use meaningful symbolic constants to represent literal values
3120	EXP07-C. Do not diminish the benefits of constants by assuming their values in expressions
3121	DCL06-C. Use meaningful symbolic constants to represent literal values
3121	EXP07-C. Do not diminish the benefits of constants by assuming their values in expressions
3122	DCL06-C. Use meaningful symbolic constants to represent literal values
3122	EXP07-C. Do not diminish the benefits of constants by assuming their values in expressions
3123	DCL06-C. Use meaningful symbolic constants to represent literal values
3123	EXP07-C. Do not diminish the benefits of constants by assuming their values in expressions
3131	DCL06-C. Use meaningful symbolic constants to represent literal values
3131	EXP07-C. Do not diminish the benefits of constants by assuming their values in expressions
3132	DCL06-C. Use meaningful symbolic constants to represent literal values
3132	EXP07-C. Do not diminish the benefits of constants by assuming their values in expressions
3200	ERR33-C. Detect and handle standard library errors
3200	POS54-C. Detect and handle POSIX library errors
3200	EXP12-C. Do not ignore values returned by functions
3202	MSC07-C. Detect and remove dead code
3203	MSC07-C. Detect and remove dead code
3203	MSC13-C. Detect and remove unused values
3204	DCL00-C. Const-qualify immutable objects
3205	MSC07-C. Detect and remove dead code
3205	MSC13-C. Detect and remove unused values
3206	MSC07-C. Detect and remove dead code
3206	MSC13-C. Detect and remove unused values
3207	MSC07-C. Detect and remove dead code
3207	MSC13-C. Detect and remove unused values
3210	DCL19-C. Minimize the scope of variables and functions
3210	MSC07-C. Detect and remove dead code
3217	DCL30-C. Declare objects with appropriate storage durations
3217	DCL21-C. Understand the storage of compound literals
3218	DCL19-C. Minimize the scope of variables and functions
3219	MSC07-C. Detect and remove dead code
3225	DCL30-C. Declare objects with appropriate storage durations
3226	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
3227	DCL00-C. Const-qualify immutable objects

3229	MSC07-C. Detect and remove dead code
3229	MSC13-C. Detect and remove unused values
3230	DCL30-C. Declare objects with appropriate storage durations
3232	DCL00-C. Const-qualify immutable objects
3234	DCL41-C. Do not declare variables inside a switch statement before the first case label
3236	MSC40-C. Do not violate constraints
3237	MSC40-C. Do not violate constraints
3238	MSC40-C. Do not violate constraints
3239	MSC15-C. Do not depend on undefined behavior
3244	MSC40-C. Do not violate constraints
3305	EXP36-C. Do not cast pointers into more strictly aligned pointer types
3305	EXP39-C. Do not access a variable through a pointer of an incompatible type
3307	EXP44-C. Do not rely on side effects in operands to sizeof, _Alignof, or _Generic
3307	MSC12-C. Detect and remove code that has no effect or is never executed
3311	MSC15-C. Do not depend on undefined behavior
3312	MSC15-C. Do not depend on undefined behavior
3314	EXP45-C. Do not perform assignments in selection statements
3319	MSC15-C. Do not depend on undefined behavior
3320	EXP37-C. Call functions with the correct number and type of arguments
3326	EXP45-C. Do not perform assignments in selection statements
3326	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
3331	DCL07-C. Include the appropriate type information in function declarators
3334	DCL01-C. Do not reuse variable names in subscopes
3335	DCL31-C. Declare identifiers before using them
3335	EXP37-C. Call functions with the correct number and type of arguments
3335	DCL07-C. Include the appropriate type information in function declarators
3339	FLP30-C. Do not use floating-point variables as loop counters
3340	FLP30-C. Do not use floating-point variables as loop counters
3342	FLP30-C. Do not use floating-point variables as loop counters
3344	EXP45-C. Do not perform assignments in selection statements
3344	EXP46-C. Do not use a bitwise operator with a Boolean-like operand
3344	EXP16-C. Do not compare function pointers to constant values
3344	EXP20-C. Perform explicit tests to determine success, true and false, and equality
3383	INT30-C. Ensure that unsigned integer operations do not wrap
3384	INT30-C. Ensure that unsigned integer operations do not wrap
3385	INT30-C. Ensure that unsigned integer operations do not wrap
3386	INT30-C. Ensure that unsigned integer operations do not wrap
3389	EXP00-C. Use parentheses for precedence of operation
3390	EXP00-C. Use parentheses for precedence of operation
3391	EXP00-C. Use parentheses for precedence of operation
3392	EXP00-C. Use parentheses for precedence of operation
3392	EXP13-C. Treat relational and equality operators as if they were nonassociative
3393	EXP00-C. Use parentheses for precedence of operation
3394	EXP00-C. Use parentheses for precedence of operation
3395	EXP00-C. Use parentheses for precedence of operation
3396	EXP00-C. Use parentheses for precedence of operation
3397	EXP00-C. Use parentheses for precedence of operation

3398	EXP00-C. Use parentheses for precedence of operation
3399	EXP00-C. Use parentheses for precedence of operation
3400	EXP00-C. Use parentheses for precedence of operation
3401	EXP13-C. Treat relational and equality operators as if they were nonassociative
3404	MSC07-C. Detect and remove dead code
3404	MSC12-C. Detect and remove code that has no effect or is never executed
3408	DCL07-C. Include the appropriate type information in function declarators
3409	PRE02-C. Macro replacement lists should be parenthesized
3410	PRE01-C. Use parentheses within macros around parameter names
3412	PRE10-C. Wrap multistatement macros in a do-while loop
3412	PRE11-C. Do not conclude macro definitions with a semicolon
3413	PRE03-C. Prefer typedefs to defines for encoding non-pointer types
3415	EXP02-C. Be aware of the short-circuit behavior of the logical AND and OR operators
3416	EXP45-C. Do not perform assignments in selection statements
3422	MSC07-C. Detect and remove dead code
3423	MSC07-C. Detect and remove dead code
3425	MSC07-C. Detect and remove dead code
3426	MSC12-C. Detect and remove code that has no effect or is never executed
3427	MSC12-C. Detect and remove code that has no effect or is never executed
3437	MSC38-C. Do not treat a predefined identifier as an object if it might only be implemented as a macro
3437	MSC15-C. Do not depend on undefined behavior
3438	MSC15-C. Do not depend on undefined behavior
3450	DCL07-C. Include the appropriate type information in function declarators
3453	PRE00-C. Prefer inline or static functions to function-like macros
3456	PRE12-C. Do not define unsafe macros
3458	PRE10-C. Wrap multistatement macros in a do-while loop
3462	PRE31-C. Avoid side effects in arguments to unsafe macros
3463	PRE31-C. Avoid side effects in arguments to unsafe macros
3464	PRE31-C. Avoid side effects in arguments to unsafe macros
3465	PRE31-C. Avoid side effects in arguments to unsafe macros
3466	PRE31-C. Avoid side effects in arguments to unsafe macros
3467	PRE31-C. Avoid side effects in arguments to unsafe macros
3470	MSC07-C. Detect and remove dead code
3475	MSC38-C. Do not treat a predefined identifier as an object if it might only be implemented as a macro
3601	PRE07-C. Avoid using repeated question marks
3664	MSC14-C. Do not introduce unnecessary platform dependencies
3670	MEM05-C. Avoid large stack allocations
3673	DCL00-C. Const-qualify immutable objects
3673	DCL13-C. Declare function parameters that are pointers to values not changed by the function as const
3674	ARR02-C. Explicitly specify array bounds, even if implicitly defined by an initializer
3677	DCL00-C. Const-qualify immutable objects
3677	DCL13-C. Declare function parameters that are pointers to values not changed by the function as const
3684	ARR02-C. Explicitly specify array bounds, even if implicitly defined by an initializer
4111	EXP13-C. Treat relational and equality operators as if they were nonassociative
4112	EXP13-C. Treat relational and equality operators as if they were nonassociative
4113	EXP13-C. Treat relational and equality operators as if they were nonassociative
4116	EXP20-C. Perform explicit tests to determine success, true and false, and equality

4117	FLP36-C. Preserve precision when converting integral values to floating-point type
4117	FLP06-C. Convert integers to floating point for floating-point operations
4118	FLP06-C. Convert integers to floating point for floating-point operations
4140	DCL30-C. Declare objects with appropriate storage durations
4401	INT02-C. Understand integer conversion rules
4401	INT07-C. Use only explicitly signed or unsigned char type for numeric values
4402	INT02-C. Understand integer conversion rules
4403	INT02-C. Understand integer conversion rules
4404	INT02-C. Understand integer conversion rules
4405	INT02-C. Understand integer conversion rules
4410	INT02-C. Understand integer conversion rules
4412	INT02-C. Understand integer conversion rules
4413	STR37-C. Arguments to character-handling functions must be representable as an unsigned char
4413	INT02-C. Understand integer conversion rules
4414	STR37-C. Arguments to character-handling functions must be representable as an unsigned char
4414	INT02-C. Understand integer conversion rules
4415	INT02-C. Understand integer conversion rules
4420	INT02-C. Understand integer conversion rules
4421	INT02-C. Understand integer conversion rules
4421	INT07-C. Use only explicitly signed or unsigned char type for numeric values
4422	INT02-C. Understand integer conversion rules
4423	INT02-C. Understand integer conversion rules
4424	INT02-C. Understand integer conversion rules
4425	INT02-C. Understand integer conversion rules
4430	INT02-C. Understand integer conversion rules
4431	INT02-C. Understand integer conversion rules
4431	INT07-C. Use only explicitly signed or unsigned char type for numeric values
4432	INT02-C. Understand integer conversion rules
4434	INT02-C. Understand integer conversion rules
4435	FLP36-C. Preserve precision when converting integral values to floating-point type
4435	INT02-C. Understand integer conversion rules
4436	INT02-C. Understand integer conversion rules
4437	FLP36-C. Preserve precision when converting integral values to floating-point type
4437	INT02-C. Understand integer conversion rules
4440	INT02-C. Understand integer conversion rules
4441	INT02-C. Understand integer conversion rules
4441	INT07-C. Use only explicitly signed or unsigned char type for numeric values
4442	INT02-C. Understand integer conversion rules
4443	INT02-C. Understand integer conversion rules
4445	FLP36-C. Preserve precision when converting integral values to floating-point type
4445	INT02-C. Understand integer conversion rules
4446	INT02-C. Understand integer conversion rules
4447	INT02-C. Understand integer conversion rules
4450	FLP34-C. Ensure that floating-point conversions are within range of the new type
4451	FLP34-C. Ensure that floating-point conversions are within range of the new type
4451	INT07-C. Use only explicitly signed or unsigned char type for numeric values
4452	FLP34-C. Ensure that floating-point conversions are within range of the new type

4453	FLP34-C. Ensure that floating-point conversions are within range of the new type
4454	FLP34-C. Ensure that floating-point conversions are within range of the new type
4460	INT02-C. Understand integer conversion rules
4461	INT02-C. Understand integer conversion rules
4462	FLP34-C. Ensure that floating-point conversions are within range of the new type
4463	INT02-C. Understand integer conversion rules
4464	INT02-C. Understand integer conversion rules
4465	FLP34-C. Ensure that floating-point conversions are within range of the new type
4470	INT02-C. Understand integer conversion rules
4471	INT02-C. Understand integer conversion rules
4480	INT02-C. Understand integer conversion rules
4481	INT02-C. Understand integer conversion rules
4490	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
4491	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
4492	INT18-C. Evaluate integer expressions in a larger size before comparing or assigning to that size
4502	EXP46-C. Do not use a bitwise operator with a Boolean-like operand
4532	INT13-C. Use bitwise operators only on unsigned operands
4533	INT13-C. Use bitwise operators only on unsigned operands
4534	INT13-C. Use bitwise operators only on unsigned operands
4543	INT13-C. Use bitwise operators only on unsigned operands
4544	INT13-C. Use bitwise operators only on unsigned operands
4600	DCL37-C. Do not declare or define a reserved identifier
4601	DCL37-C. Do not declare or define a reserved identifier
4602	DCL37-C. Do not declare or define a reserved identifier
4603	DCL37-C. Do not declare or define a reserved identifier
4604	DCL37-C. Do not declare or define a reserved identifier
4605	DCL37-C. Do not declare or define a reserved identifier
4606	DCL37-C. Do not declare or define a reserved identifier
4607	DCL37-C. Do not declare or define a reserved identifier
4608	DCL37-C. Do not declare or define a reserved identifier
4620	DCL37-C. Do not declare or define a reserved identifier
4621	DCL37-C. Do not declare or define a reserved identifier
4622	DCL37-C. Do not declare or define a reserved identifier
4623	DCL37-C. Do not declare or define a reserved identifier
4624	DCL37-C. Do not declare or define a reserved identifier
4640	DCL37-C. Do not declare or define a reserved identifier
4641	DCL37-C. Do not declare or define a reserved identifier
4642	DCL37-C. Do not declare or define a reserved identifier
4643	DCL37-C. Do not declare or define a reserved identifier
4644	DCL37-C. Do not declare or define a reserved identifier
4645	DCL37-C. Do not declare or define a reserved identifier
4926	POS50-C. Declare objects shared between POSIX threads with appropriate storage durations
4927	POS50-C. Declare objects shared between POSIX threads with appropriate storage durations
4928	POS50-C. Declare objects shared between POSIX threads with appropriate storage durations
4951	POS38-C. Beware of race conditions when using fork and file descriptors
4952	POS38-C. Beware of race conditions when using fork and file descriptors
4955	ARR39-C. Do not add or subtract a scaled integer to a pointer

4956	ARR39-C. Do not add or subtract a scaled integer to a pointer
4957	ARR39-C. Do not add or subtract a scaled integer to a pointer
4961	CON31-C. Do not destroy a mutex while it is locked
4962	CON31-C. Do not destroy a mutex while it is locked
4966	POS52-C. Do not perform operations that can block while holding a POSIX lock
4967	POS52-C. Do not perform operations that can block while holding a POSIX lock
4971	POS48-C. Do not unlock or destroy another POSIX thread's mutex
4972	POS48-C. Do not unlock or destroy another POSIX thread's mutex
4976	CON33-C. Avoid race conditions when using library functions
4977	CON33-C. Avoid race conditions when using library functions
4981	POS48-C. Do not unlock or destroy another POSIX thread's mutex
4982	POS48-C. Do not unlock or destroy another POSIX thread's mutex
4991	ENV31-C. Do not rely on an environment pointer following an operation that may invalidate it
4992	ENV31-C. Do not rely on an environment pointer following an operation that may invalidate it
4993	ENV31-C. Do not rely on an environment pointer following an operation that may invalidate it
5001	PRE04-C. Do not reuse a standard header file name
5002	PRE08-C. Guarantee that header file names are unique
5003	PRE09-C. Do not replace secure functions with deprecated or obsolescent functions
5004	DCL05-C. Use typedefs of non-pointer types only
5005	INT05-C. Do not use input functions to convert character data if they cannot handle all possible inputs
5007	STR06-C. Do not assume that strtok() leaves the parse string unchanged
5008	STR07-C. Use the bounds-checking interfaces for string manipulation
5009	STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator
5010	MEM03-C. Clear sensitive information stored in reusable resources
5011	FIO01-C. Be careful using functions that use file names for identification
5012	FIO03-C. Do not make assumptions about fopen() and file creation
5013	FIO06-C. Create files with appropriate access permissions
5014	FIO08-C. Take care when calling remove() on an open file
5015	FIO10-C. Take care when using the rename() function
5016	FIO21-C. Do not create temporary files in shared directories
5017	ENV03-C. Sanitize the environment when invoking external programs
5018	ENV33-C. Do not call system()
5019	SIG00-C. Mask signals handled by noninterruptible signal handlers
5020	SIG01-C. Understand implementation-specific details regarding signal handler persistence
5021	SIG34-C. Do not call signal() from within interruptible signal handlers
5021	CON37-C. Do not call signal() in a multithreaded program
5022	MSC30-C. Do not use the rand() function for generating pseudorandom numbers
5023	POS33-C. Do not use vfork()
5024	POS34-C. Do not call putenv() with a pointer to an automatic variable as the argument
5025	FLP32-C. Prevent or detect domain and range errors in math functions
5026	FLP37-C. Do not use object representations to compare floating-point values
5027	MEM36-C. Do not modify the alignment of objects by calling realloc()
5028	FIO38-C. Do not copy a FILE object
5029	FIO39-C. Do not alternately input and output from a stream without an intervening flush or positioning call
5030	ERR34-C. Detect errors when converting a string to a number
5031	MSC32-C. Properly seed pseudorandom number generators
5032	MSC33-C. Do not pass invalid data to the asctime() function

5033	POS30-C. Use the readlink() function properly
5034	POS44-C. Do not use signals to terminate threads
5035	POS47-C. Do not use threads that can be canceled asynchronously
5036	FIO41-C. Do not call getc(), putc(), getwc(), or putwc() with a stream argument that has side effects
[I]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	FIO47-C. Use valid format strings
[U]	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
[U]	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
[U]	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
[U]	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
[U]	EXP10-C. Do not depend on the order of evaluation of subexpressions or the order in which side effects take place
]	FIO47-C. Use valid format strings
C99	FIO47-C. Use valid format strings