```
 1: // $Id: enumhashing.cpp,v 1.6 2018-11-07 11:44:47-08 - - $
 2:
 3: //
 4: // Example of how to convert constants of an enum class into strings.
 5: //
 6:
 7: #include <bitset>
 8: #include <iostream>
 9: #include <stdexcept>
10: #include <unordered_map>
11: #include <vector>
12: using namespace std;
13:
14: enum class attr: unsigned {
15:        VOID, INT, NULLX, STRING, STRUCT, ARRAY, FUNCTION, VARIABLE,
16:        FIELD, TYPEID, PARAM, LVAL, CONST, VREG, VADDR, BITSET_SIZE,
17: };
18: using attr_bitset = bitset<unsigned(attr::BITSET_SIZE)>;
19:
20: const string to_string (attr attribute) {
21:    static const unordered_map<attr,string> hash {
22:       {attr::VOID       , "void"       },
23:       {attr::INT        , "int"        },
24:       {attr::NULLX      , "null"       },
25:       {attr::STRING     , "string"     },
26:       {attr::STRUCT     , "struct"     },
27:       {attr::ARRAY      , "array"      },
28:       {attr::FUNCTION   , "function"   },
29:       {attr::VARIABLE   , "variable"   },
30:       {attr::FIELD      , "field"      },
31:       {attr::TYPEID     , "typeid"     },
32:       {attr::PARAM      , "param"      },
33:       {attr::LVAL       , "lval"       },
34:       {attr::CONST      , "const"      },
35:       {attr::VREG       , "vreg"       },
36:       {attr::VADDR      , "vaddr"      },
37:       {attr::BITSET_SIZE, "bitset_size"},
38:    };
39:    auto str = hash.find (attribute);
40:    if (str == hash.end()) {
41:       throw invalid_argument (string (__PRETTY_FUNCTION__) + ": "
42:                               + to_string (unsigned (attribute)));
43:    }
44:    return str->second;
45: }
46:
```

```
47:
48: int main() {
49:    static vector<attr> attrs {
50:       attr::VOID, attr::INT, attr::NULLX, attr::STRING, attr::STRUCT,
51:       attr::ARRAY, attr::FUNCTION, attr::VARIABLE, attr::FIELD,
52:       attr::TYPEID, attr::PARAM, attr::LVAL, attr::CONST, attr::VREG,
53:       attr::VADDR, attr::BITSET_SIZE,
54:    };
55:    for (const auto attrib: attrs) {
56:       cout << unsigned(attrib) << " = " << to_string (attrib) << endl;
57:    }
58:    try {
59:       cout << to_string (static_cast<attr> (1024)) << endl;
60:    }catch (invalid_argument& what) {
61:       cout << "invalid_argument: " << what.what() << endl;
62:    }
63: }
64:
65: //TEST// ./enumhashing >enumhashing.lis 2>&1
66: //TEST// mkpspdf enumhashing.ps enumhashing.cpp* enumhashing.lis
67:
```

```
    1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting enumhashing.cpp
    2: enumhashing.cpp:
    3:      $Id: enumhashing.cpp,v 1.6 2018-11-07 11:44:47-08 - - $
    4: cpplint.py.perl enumhashing.cpp
    5: Done processing enumhashing.cpp
    6: g++ -g -O0 -Wall -Wextra -fdiagnostics-color=never -std=gnu++17 -Wold-st
yle-cast enumhashing.cpp -o enumhashing -lm
    7: rm -f enumhashing.o
    8: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished enumhashing.cpp
```

```
 1: 0 = void
 2: 1 = int
 3: 2 = null
 4: 3 = string
 5: 4 = struct
 6: 5 = array
 7: 6 = function
 8: 7 = variable
 9: 8 = field
10: 9 = typeid
11: 10 = param
12: 11 = lval
13: 12 = const
14: 13 = vreg
15: 14 = vaddr
16: 15 = bitset_size
17: invalid_argument: const string to_string(attr): 1024
```