```
/* The following code example is taken from the book
 * "The C++ Standard Library - A Tutorial and Reference, 2nd Edition"
 * by Nicolai M. Josuttis, Addison-Wesley, 2012
 *
 * (C) Copyright Nicolai M. Josuttis 2012.
 * Permission to copy, use, modify, sell and distribute this software
 * is granted provided this copyright notice appears in all copies.
 * This software is provided "as is" without express or implied
 * warranty, and with no claim as to its suitability for any purpose.
 */
#include <regex>
#include <string>

template <typename T>
std::string regexCode (T code)
{
    switch (code) {
      case std::regex_constants::error_collate:
        return "error_collate: "
               "regex has invalid collating element name";
      case std::regex_constants::error_ctype:
        return "error_ctype: "
               "regex has invalid character class name";
      case std::regex_constants::error_escape:
        return "error_escape: "
               "regex has invalid escaped char. or trailing escape";
      case std::regex_constants::error_backref:
        return "error_backref: "
               "regex has invalid back reference";
      case std::regex_constants::error_brack:
        return "error_brack: "
               "regex has mismatched '[' and ']'";
      case std::regex_constants::error_paren:
        return "error_paren: "
               "regex has mismatched '(' and ')'";
      case std::regex_constants::error_brace:
        return "error_brace: "
               "regex has mismatched '{' and '}'";
      case std::regex_constants::error_badbrace:
        return "error_badbrace: "
               "regex has invalid range in {} expression";
      case std::regex_constants::error_range:
        return "error_range: "
               "regex has invalid character range, such as '[b-a]'";
      case std::regex_constants::error_space:
        return "error_space: "
               "insufficient memory to convert regex into finite state";
      case std::regex_constants::error_badrepeat:
        return "error_badrepeat: "
               "one of *?+{ not preceded by valid regex";
      case std::regex_constants::error_complexity:
        return "error_complexity: "
               "complexity of match against regex over pre-set level";
      case std::regex_constants::error_stack:
        return "error_stack: "
               "insufficient memory to determine regex match";
```

```
    }
    return "unknown/non-standard regex error code";
}
```