

Data Abstraction Implementation in C++ (P2)

• Use more advanced C++ class features, including move() optimizations

```
Edit View Navigate Code Refactor Build Run Tools VCS Window Help
                                           CPP-data-abstraction 3b - main.cpp - CLion
                                                                              — January main.cpp
 3b D:\Douglas Schmidt\Dropbox\ 1 0
                            * This example of the stack shows how to implement it as a C++ ADT.
 make-build-debug
  A CMakeLists.txt
                            * https://en.wikipedia.org/wiki/Abstract_data_type. It also demonstrates
                            * the C++ "Rule of 5", as well as demonstrates the difference between copy
  a simple_string.cpp
  simple_string.h
                            * semantics and move semantics for constructors and assignment operators.
   stack.cpp
  stack.h
  atack.i
 | External Libraries
                  15
                           int main() {
 Scratches and Consoles
                             // Multiple stacks that are created automatically.
                   16
                             cout << "begin constructors" << endl;</pre>
                  17
                             stack s1( size: 1), s2( size: 0);
                             cout << "end constructors" << endl;</pre>
                  19
                             auto item = 0;
                             cout << "begin push(const T&)" << endl:</pre>
                  23
                             while (!s1.full()) {
                               simple_string s = std::to_string(item++).c_str();
                               s1.push(s);
                             cout << "end push(const T&)" << endl:
1:1 CRLF UTF-8 4 spaces C++: CPP-data-abstraction 3b | Debug
Build finished in 3 s 737 ms (5 minutes ago)
```



Pros of Data Abstraction in C++ (P2)

- Optimizes pass-by-value semantics via std::move()
 & rvalue references
- Inlines small methods to avoid function call overhead





Cons of Data Abstraction in C++ (P2)

- Error handling is still obtrusive
 - Use exception handling to solve this (but be careful)!
- The example is limited to a single type of stack element (simple_string in this case)
 - We can use C++ "parameterized types" to remove this limitation



End of C++ Data Abstraction Stack Implementations