

```
1 // Copyright (C) 2015 Vicente J. Botet Escriba
2 //
3 // Distributed under the Boost Software License, Version 1.0. (See accompanying
4 // file LICENSE_1_0.txt or copy at http://www.boost.org/LICENSE_1_0.txt)
5 // https://www.codeproject.com/Articles/15351/Implementing-a-simple-smart-
    pointer-in-c
6
7 #include "stdafx.h"
8 #include <thread>
9 #include <stack>
10 #include <mutex>
11 #include <memory>
12 #include <atomic>
13 #include <vector>
14 #include <deque>
15 #include <list>
16 #include <set>
17 #include <map>
18 #include <string>
19 #include <algorithm>
20 #include <iterator>
21 #include <functional>
22 #include <numeric>
23 #include <iostream>
24 #include <vector>
25 #include <list>
26 #include <queue>
27 #include <algorithm>
28 #include <typeinfo>
29 #include <algorithm>
30 #include <iterator>
31 #include <memory>
32 #include <fstream>
33 #include <vector>
34 #include <type_traits>
35 #include <future>
36
37 std::mutex m_io_m;
38 std::promise<void> init_done;
39 std::shared_future<void> ready_future(init_done.get_future());
40
41 void print(const std::string &ms) {
42     std::unique_lock<std::mutex> lc;
43     std::cout << ms << std::endl;
44 }
45
46 void init() {
47     print("inti_start");
48     std::this_thread::sleep_for(std::chrono::seconds(5));
```

```
49     print("inti_end");
50     init_done.set_value();
51 }
52
53 void f1() {
54     ready_future.wait();
55     print("f1_start");
56     std::this_thread::sleep_for(std::chrono::seconds(5));
57     print("f1_end");
58 }
59
60 void f2() {
61     ready_future.wait();
62     print("f2_start");
63     std::this_thread::sleep_for(std::chrono::seconds(5));
64     print("f2_end");
65 }
66
67 int main() {
68
69     std::thread t0(init); std::thread t1(f1); std::thread t2(f2);
70     t0.join(); t1.join(); t2.join();
71
72
73     return 0;
74 }
75
```