```cpp
// Copyright (C) 2015 Vicente J. Botet Escriba
//
//  Distributed under the Boost Software License, Version 1.0. (See accompanying
//  file LICENSE_1_0.txt or copy at http://www.boost.org/LICENSE_1_0.txt)
// https://www.codeproject.com/Articles/15351/Implementing-a-simple-smart-pointer-in-c

#include "stdafx.h"
#include <thread>
#include <stack>
#include <mutex>
#include <memory>
#include <atomic>
#include <vector>
#include <deque>
#include <list>
#include <set>
#include <map>
#include <string>
#include <algorithm>
#include <iterator>
#include <functional>
#include <numeric>
#include <iostream>
#include <vector>
#include <list>
#include <queue>
#include <algorithm>
#include <typeinfo>
#include <algorithm>
#include <iterator>
#include <memory>
#include <fstream>
#include <vector>
#include <type_traits>
#include <future>

std::mutex m_io_m;
std::promise<void> init_done;
std::shared_future<void> ready_future(init_done.get_future());

void print(const std::string &ms) {
    std::unique_lock<std::mutex> lc;
    std::cout << ms << std::endl;
}

void init() {
    print("inti_start");
    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
```