```
/* 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 <condition_variable>
#include <mutex>
#include <future>
#include <iostream>

bool readyFlag;
std::mutex readyMutex;
std::condition_variable readyCondVar;

void thread1()
{
    // do something thread2 needs as preparation
    std::cout << "<return>" << std::endl;
    std::cin.get();

    // signal that thread1 has prepared a condition
    {
        std::lock_guard<std::mutex> lg(readyMutex);
        readyFlag = true;
    } // release lock
    readyCondVar.notify_one();
}

void thread2()
{
    // wait until thread1 is ready (readyFlag is true)
    {
        std::unique_lock<std::mutex> ul(readyMutex);
        readyCondVar.wait(ul, []{ return readyFlag; });
    } // release lock

    // do whatever shall happen after thread1 has prepared things
    std::cout << "done" << std::endl;
}

int main()
{
    auto f1 = std::async(std::launch::async,thread1);
    auto f2 = std::async(std::launch::async,thread2);
}
```