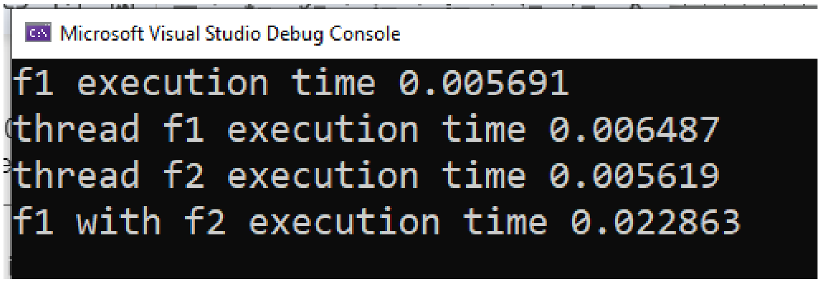
Standard Thread in CPP

A multi-thread or concurrent code at some point runs multiple tasks simultaneously. For example, an app can run some calculations with one thread in the background while the GUI is still responsive with another thread. C++ provides a thread library to write multi thread codes.

Resource: https://iamsorush.com/posts/cpp-std-thread/

Obtain the code P2Q1.cpp and run it, you will get the following output:



```
f1 execution time 0.005691
thread f1 execution time 0.006487
thread f2 execution time 0.005619
f1 with f2 execution time 0.022863
```

Question 1:

Modify the code to support 8 threads running concurrently, with a higher number of count, and compare with sequence processing (without using thread)

Observe the answer and understand the benefit of the running process concurrently using thread.

POSIX Threads in CPP

Resource: cs.cmu.edu

We introduced POSIX Threads, or Pthreads, which is a Portable Operating System Interface (POSIX) standard for threads. The standard, POSIX.1c, Threads extensions (IEEE Std 1003.1c-1995), defines an API for creating and manipulating threads.

Implementations of the API are available on many Unix-like POSIX systems such as FreeBSD, NetBSD, GNU/Linux, Mac OS X and Solaris, but Microsoft Windows implementations also exist. For example, the pthreads-w32 is available and supports a subset of the Pthread API for the Windows 32-bit platform.

Pthreads are defined as a set of C language programming types and procedure calls, implemented with a pthread.h header file. In GNU/Linux, the pthread functions are not included in the standard C library. They are in libpthrea, therefore, we should add -lpthread to link our program

Creating Threads:

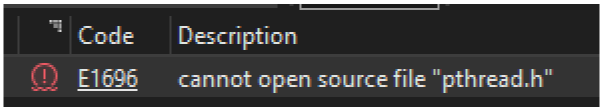
1. Our **main()** program is a single, default thread. All other threads must be explicitly created by the programmer.
2. **pthread_create** creates a new thread and makes it executable. This routine can be called any number of times from anywhere within our code.
3. **pthread_create (pthread_t *thread, pthread_attr_t *attr, void *(*start_routine)(void *), void *arg)** arguments:
    - **Thread:**
      An identifier for the new thread returned by the subroutine. This is a pointer to **pthread_t** structure. When a thread is created, an identifier is written to the memory location to which this variable points. This identifier enables us to refer to the thread.
    - **attr:**
      An attribute object that may be used to set thread attributes. We can specify a thread attribute object, or NULL for the default values.
    - **start_routine:**
      The routine that the thread will execute once it is created. We should pass the address of a function taking a pointer to void as a parameter and the function will return a pointer to void. So, we can pass any type of single

argument and return a pointer to any type. While using **fork()** causes execution to continue in the same location with a different return code, using a new thread explicitly provides a pointer to a function where the new thread should start executing.

- **arg:**
  A single argument that may be passed to **start_routine**. It must be passed as a **void pointer**. NULL may be used if no argument is to be passed.

4. The maximum number of threads that may be created by a process is implementation dependent.
5. Once created, threads are peers, and may create other threads. There is no implied hierarchy or dependency between threads.

Obtain the code P2Q2.cpp and try to compile it. If you have the following error:



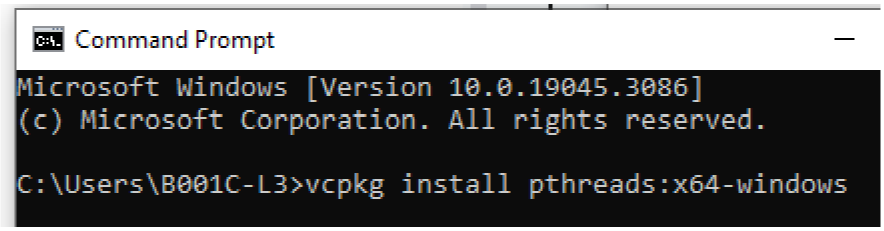
Code	Description
E1696	cannot open source file "pthread.h"

It means your computer has not installed the pthread library.

You need to exit the visual studio and install vcpkg. Download the vcpkg from https://github.com/Microsoft/vcpkg and execute the batch file:
`bootstrap-vcpkg.bat`

Then, type the following command in the command prompt.

```
vcpkg install pthread
vcpkg integrate install
vcpkg install pthreads:x64-windows
```



```
Microsoft Windows [Version 10.0.19045.3086]
(c) Microsoft Corporation. All rights reserved.

C:\Users\B001C-L3>vcpkg install pthreads:x64-windows
```

Question 2:

Run the P2Q2.cpp.

UDP Communication

Question 3：

Repeat Lecture 2 UDP Exercise by using your computer in practical class.

Show the result of UDP (send and receive) at server and client side (localhost / 127.0.0.1) to your practical instructor.

Question 4:

Use thread / pthread class to establish multiple UDP communication with your group mate, using different computers (different ip addresses).

Show the result of UDP (send and receive) at server and client side to your practical instructor.