20220528-C++

1.学习内容

1.1 C++多线程

C++ | 复制代码

```cpp
#include <iostream>
#include <thread>
#include <chrono>

/*一个程序运行时间计时器*/
class Timer
{
public:
    Timer()
    {
        startTimePoint=std::chrono::high_resolution_clock::now();
    }
    ~Timer()
    {
        Stop();
    }
    void Stop()
    {
        auto endTimePoint = std::chrono::high_resolution_clock::now();
        auto start = std::chrono::time_point_cast<std::chrono::microseconds>(startTimePoint).time_since_epoch().count();
        auto end = std::chrono::time_point_cast<std::chrono::microseconds>(endTimePoint).time_since_epoch().count();
        auto duration = end - start;
        double ms = duration * 0.001;
        std::cout << duration << "us (" << ms << "ms)\n";
    }
private:
    std::chrono::time_point<std::chrono::high_resolution_clock> startTimePoint;
};

/*线程初试*/
void hello()
{
    std::cout << "hello world" << std::endl;
}


/*线程的开始与终止*/
struct func
{
    int& i;
```

```
41          func(int& i_):i(i_){}
42          void operator()()
43          {
44              for (int k = 0; k < 10000; k++)
45              {
46                  std::cout << i << std::endl;
47              }
48          }
49      };
50      void holy()
51      {
52          int some_state = 0;
53          func myFunc(some_state);
54          std::thread myThread(myFunc);
55          myThread.detach();//不等待线程;执行oops的线程可能先于myThread结束，而这时候
        myThread中的函数还在引用已销毁的变量
56      }
57
58
59      int main()
60      {
61          {
62              Timer timer;
63              std::thread t(hello);
64              t.join();
65          }
66          {
67              Timer timer;
68              std::cout << "hello world" << std::endl;
69          }
70          {
71              holy();
72          }
73      }
```

2.结果描述

明天回家，今天一天都没啥心情学习。之后要坚持住。