Отчёт по лабораторной работе №2 СРВ

Морозов Андрей 3.3 МО

*1. Текст задания:*

**Подготовить отчет в word-файле со скриншотами**

Создать 3 потока следующим способом:

*#include <thread>*

*#include <mutex>*

*#include <string>*

*void Func(string name)*

*{*

*...*

*}*

*int main()*

*{*

*std::thread thread1(Func, "t1");*

*std::thread thread2(Func, "t2");*

*std::thread thread3(Func, "t3");*

*thread1.join();*

*thread2.join();*

*thread3.join();*

*system("pause");*

*}*

где Func функция, выполняемая в потоке, в которой в течении одной секунды к числу, объявленному внутри этой функции:

*long double i = 0;*

прибавляется 10^(-9) при каждой итерации, пока не выйдет время.  
В конце функции вывести итоговое число следующим способом :

*m.lock();*

*std::cout << name << ": " << i << std::endl;*

*m.unlock();*

где *std::mutex m;* объявлено глобально*.*

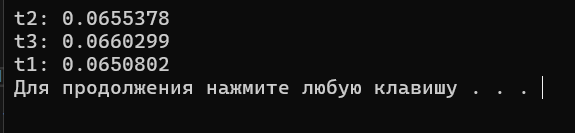
Запустите несколько раз приложение в debug и release. Посмотрите какие получаются значения. Как эти значения отличаются? Попытайтесь ответить – почему?

Удалите переменную m*,* перезапустите программу несколько раз. Как изменился вывод в консоль?

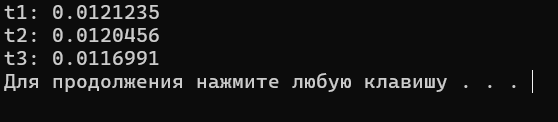
*2. Ответы на поставленные в задании вопросы*

“Запустите несколько раз приложение в debug и release. Посмотрите какие получаются значения”

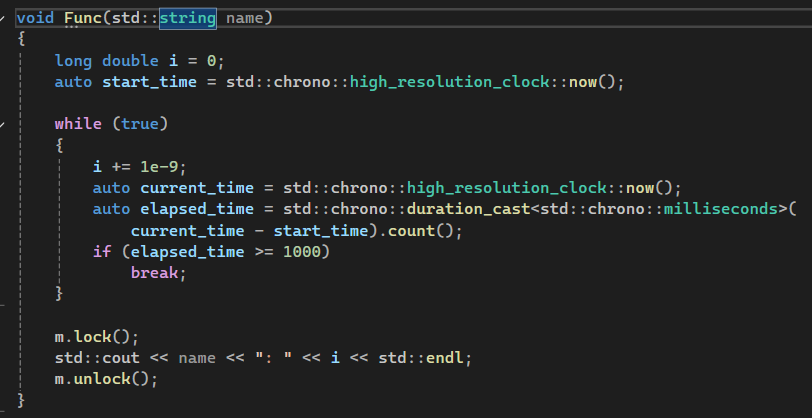
**Release**



**Debug**



**Void Func**

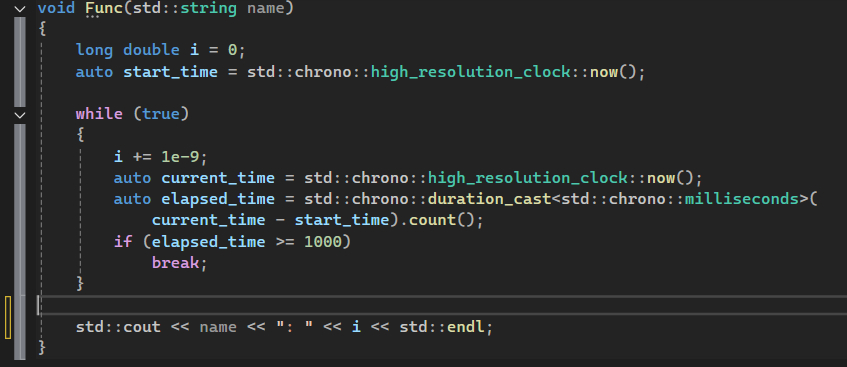


“Как эти значения отличаются? Попытайтесь ответить – почему?”

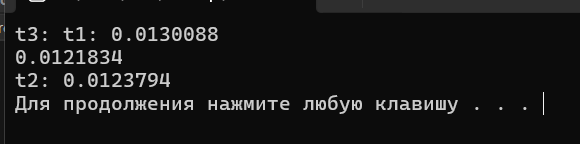
Как видно, в случае Release сборки, значения каждой из локальных переменных, объявленных в функции, примерно в 6 раз больше соответствующих значений при Debug конфигруации. Отсюда можно сделать вывод, что за один и тот же отрезок времени (1 секунда), при release конфигурации программа отрабатывает быстрее. Причиной тому является то, что Debug предназначен для отладки программы => код компилируется “как есть” для точного соотношения исходного кода с инструкциями для процессора. Также в процессе отладки необходимо знать дополнительную информацию по значениям переменных, входных параметров функций и т.п. в определенный момент выполнения программы. Release конфигурация сопровождается оптимизацией кода при компиляции в сравнении с debug и не имеет лишней для работы программы отладочной информации.

“Удалите переменную m*,* перезапустите программу несколько раз. Как изменился вывод в консоль?”

**Удаление mutex**



**Результат вывода**



Иногда вывод в консоль выглядит так, как на скриншоте сверху: сообщения накладываются друг на друга. Причина в том, что потоки, параллельно вызывая std::cout, одновременно выводили информацию по переменным. В случае наличия mutex, вывод блокируется для других потоков, как только один из них достигает ограниченного m.lock()\unlock() участка кода.