Министерство науки и образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных технологий

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**ОТЧЕТ**

по лабораторной работе № 1

по дисциплине «Теория языков программирования и методы трансляции»

**Интерфейс Open MP.**

ОГУ 09.03.04.4023.704 ПЗ

Руководитель

канд. техн. наук, доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_ Н. А. Тишина

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

Исполнитель

Студент группы 22ПИнж(б)РПиС-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.В. Федоров

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

Оренбург 2024

# Содержание

[Содержание 3](#_Toc178333856)

[*Задание 1:* Знакомство с OpenMP 4](#_Toc178333857)

[*Задание 2*: Видимость переменных 5](#_Toc178333858)

[*Задание 3*: Распараллеливание цикла for 6](#_Toc178333859)

[*Задание 4:* Распределение заданий с помощью секций 9](#_Toc178333860)

# *Задание 1:* Знакомство с OpenMP

В данном задании необходимо собрать и запустить параллельную программу, написанную при помощи OpenMP.

Так выглядит код программы

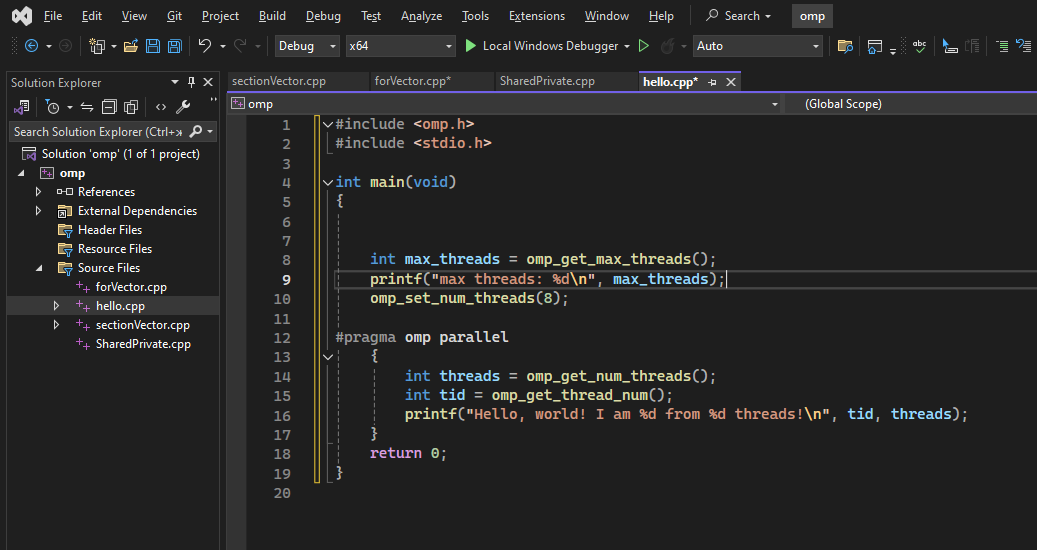


Рисунок 1. Код программы hello

Тут мы включаем или выключаем поддержку Open MP.

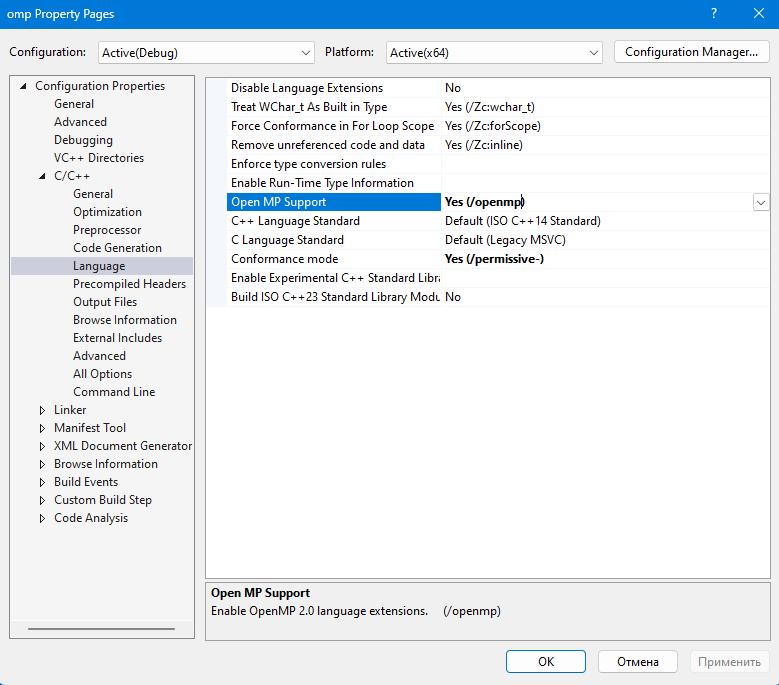


Рисунок 2. Настройки проекта с поддержкой Open MP

Запуск проекта с поддержкой Open MP

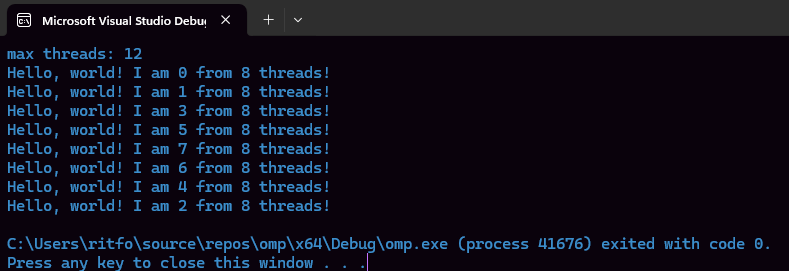


Рисунок 3. Запуск проекта с поддержкой Open MP

А теперь запустим проект без поддержки Open MP. Как видим выполняется только 1 поток.

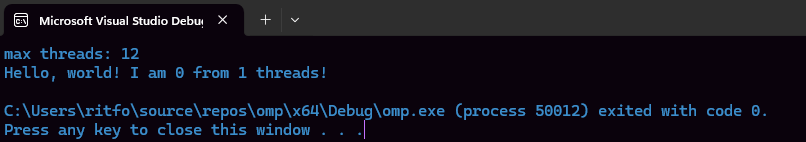


Рисунок 4. Запуск проекта без поддержки Open MP

# *Задание 2*: Видимость переменных

Создайте консольное приложение Microsoft Visual Studio с поддержкой OpenMP. Добавьте к проекту файл SharedPrivate.



Рисунок 5. Код программы SharedPrivate

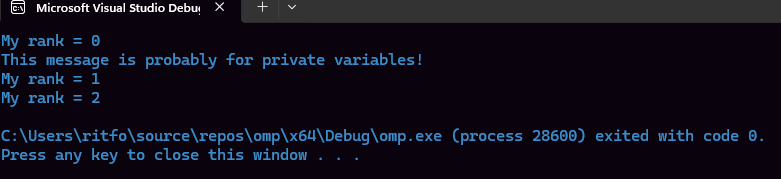


Рисунок 6. Запуск проекта с параметром private

Теперь изменим параметр private на shared:

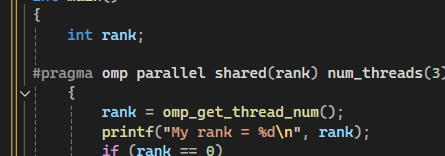


Рисунок 7. Изменен параметр на shared

И скомпилируем программу и посмотрим на вывод в консоли:

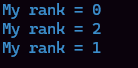


Рисунок 8. Вывод в консоли

# *Задание 3*: Распараллеливание цикла for

Вставим код программы forVector:

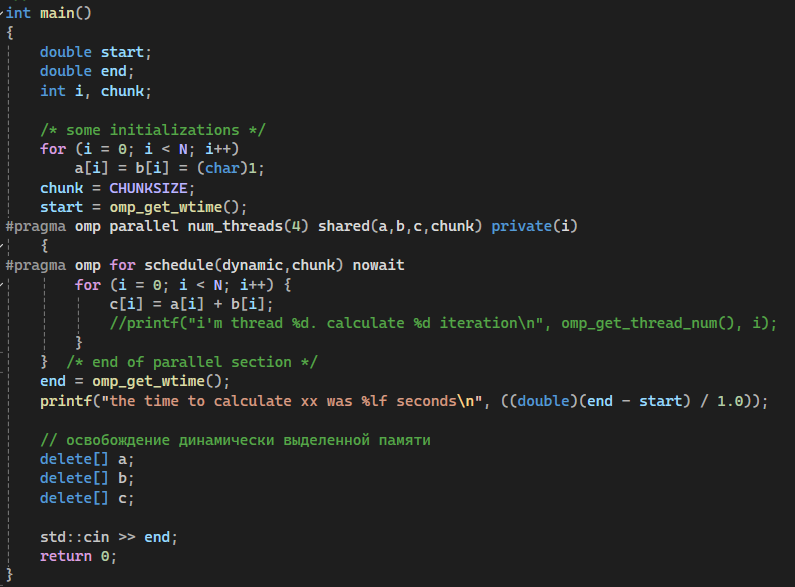


Рисунок 9. Код программы forVector

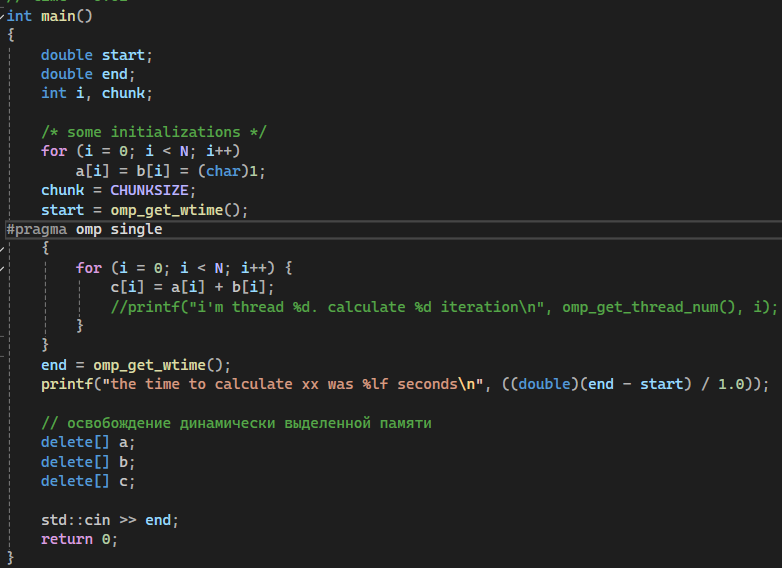
Время компиляции программы составило:



Рисунок 10. Время выполнения программы

При 4 потоках время выполнения является оптимальным из всех возможных вариантов.

Преобразуем программу к последовательной заменив parallel на single

Рисунок 11. Измененная программа с последовательным выполнением

Теперь скомпилируем программу и посмотрим сколько это будет по времени составлять.



Рисунок 12. Время выполнения последовательного способа

Как можем заметить время компиляции сократилось в **~3 раза**.

Теперь вернемся к параллельному способу и увеличим размр чанков – итерационных блоков **CHUNKSIZE**. Скомпилируем программу и сравним время выполнения.

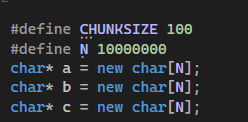


Рисунок 12. Измененная переменная CHUNKSIZE



Рисунок 13. Время выполнения параллельным способом с измененным CHUNKSIZE

Как видим время сократилось в **~ 3 раза**. То есть примерно равно выполнению последовательным способом.

# *Задание 4:* Распределение заданий с помощью секций

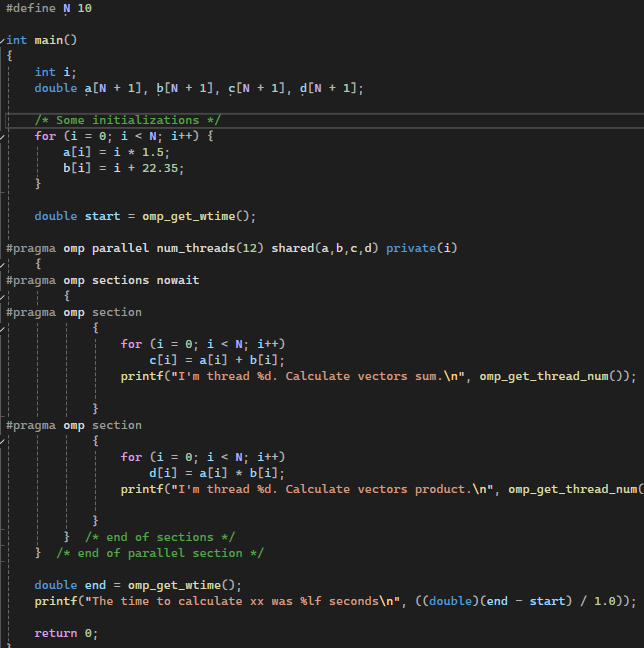


Рисунок 14. Код программы sectionsVector

Скомпилируем и измерим время выполнения с помощью функции omp\_get\_wtime().



Рисунок 15. Время выполнения sectionsVector

Преобразуем программу к последовательному выполению и опять измерим время вычислений.

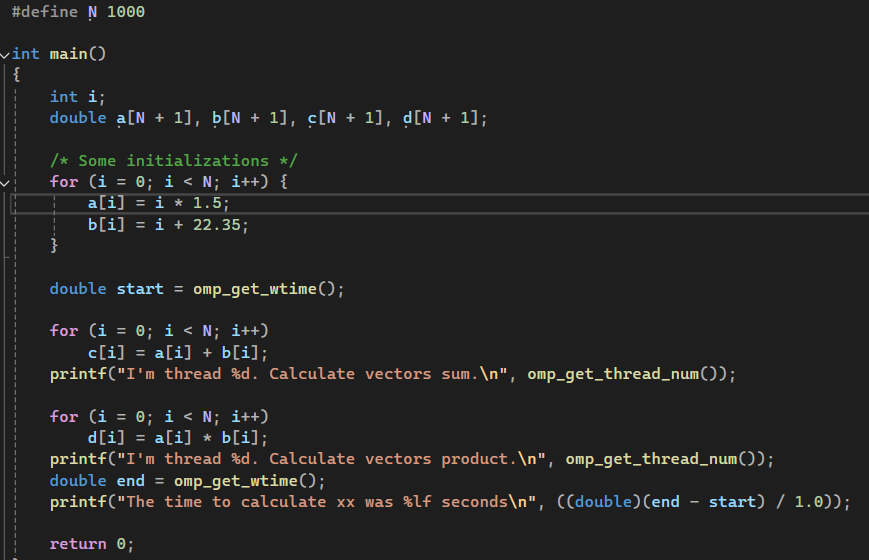
Рисунок 16. Программа с последовательным выполнением



Рисунок 17. Время выполнения последовательным способом sectionsVector

Время вычисления последовательным способом сократилось в **~45 раз**.