

Programming Assignment 1

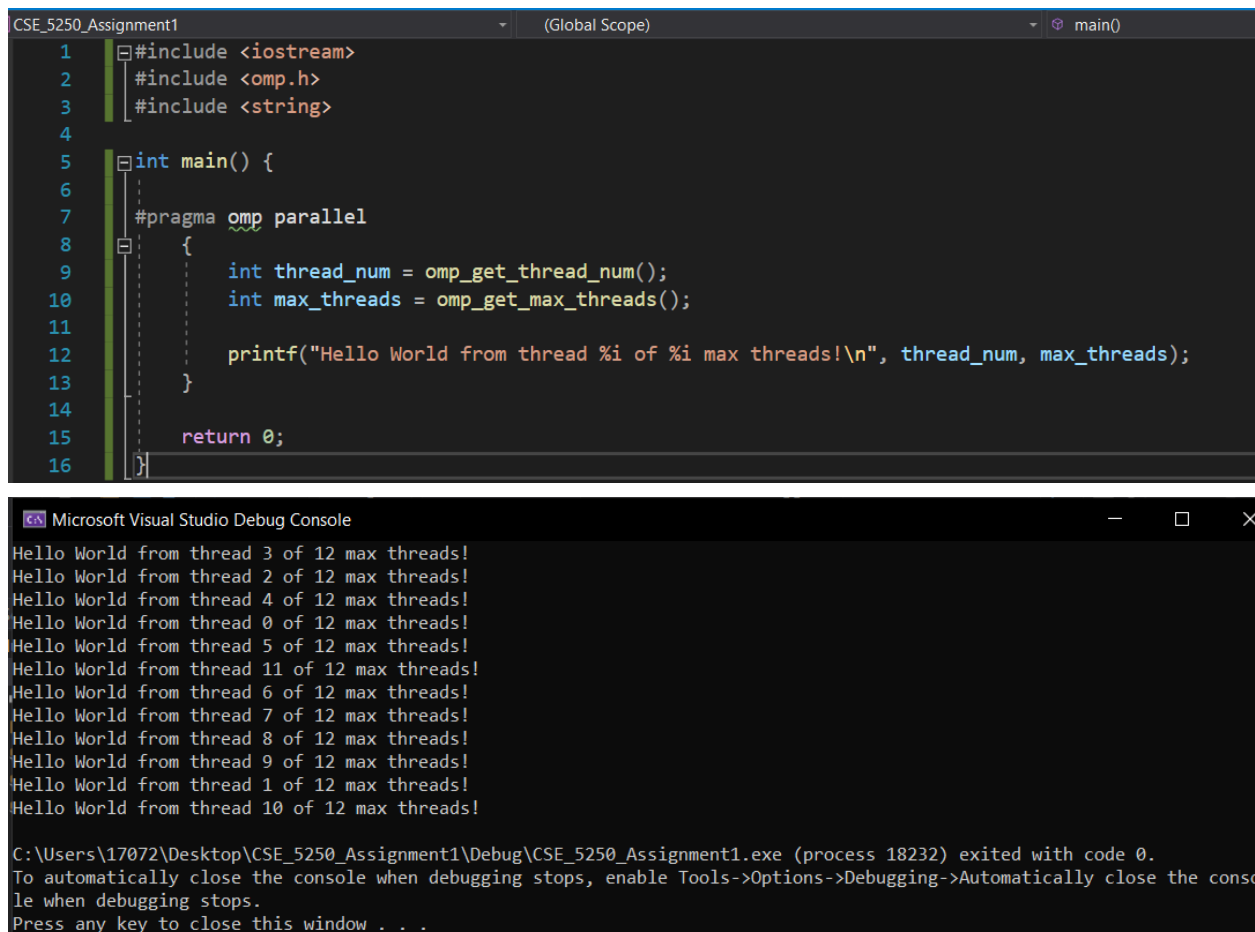
Part 1: Getting Started with OpenMp

Write a multithreaded Hello World program using OpenMP. Have each thread say "Hello world" along with its thread ID and the number of threads the program is using.

Run the program with as many threads as your computer will allow. Then try setting the thread count to a different number.

Default Thread Count:

My computer has 12 logical processors, so OpenMP defaulted to 12 threads:



The image shows a screenshot of a C++ program in Visual Studio and its execution output in the debug console. The program is a multithreaded Hello World program using OpenMP. It includes `<iostream>`, `<omp.h>`, and `<string>`. The `main` function uses `#pragma omp parallel` to create a parallel region. Inside the region, it gets the current thread number and the maximum number of threads using `omp_get_thread_num()` and `omp_get_max_threads()`. It then prints a message for each thread. The output shows 12 threads, each printing a message with their thread ID and the maximum number of threads (12).

```
1 #include <iostream>
2 #include <omp.h>
3 #include <string>
4
5 int main() {
6     #pragma omp parallel
7     {
8         int thread_num = omp_get_thread_num();
9         int max_threads = omp_get_max_threads();
10
11         printf("Hello World from thread %i of %i max threads!\n", thread_num, max_threads);
12     }
13
14     return 0;
15 }
16
```

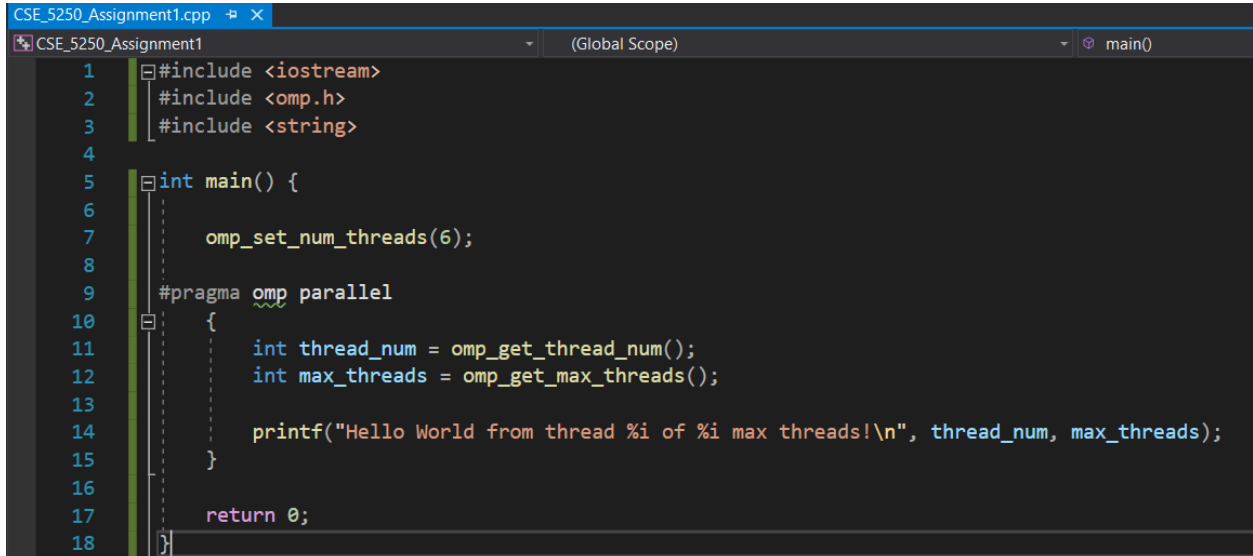
Microsoft Visual Studio Debug Console

```
Hello World from thread 3 of 12 max threads!
Hello World from thread 2 of 12 max threads!
Hello World from thread 4 of 12 max threads!
Hello World from thread 0 of 12 max threads!
Hello World from thread 5 of 12 max threads!
Hello World from thread 11 of 12 max threads!
Hello World from thread 6 of 12 max threads!
Hello World from thread 7 of 12 max threads!
Hello World from thread 8 of 12 max threads!
Hello World from thread 9 of 12 max threads!
Hello World from thread 1 of 12 max threads!
Hello World from thread 10 of 12 max threads!

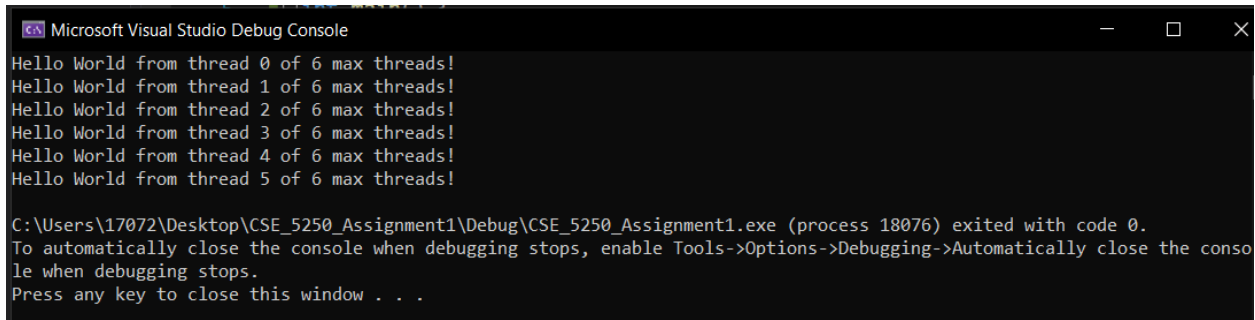
C:\Users\17072\Desktop\CSE_5250_Assignment1\Debug\CSE_5250_Assignment1.exe (process 18232) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Specific Thread Count:

I modified the program to specify 6 threads to run the output.



```
1  #include <iostream>
2  #include <omp.h>
3  #include <string>
4
5  int main() {
6
7      omp_set_num_threads(6);
8
9      #pragma omp parallel
10     {
11         int thread_num = omp_get_thread_num();
12         int max_threads = omp_get_max_threads();
13
14         printf("Hello World from thread %i of %i max threads!\n", thread_num, max_threads);
15     }
16
17     return 0;
18 }
```



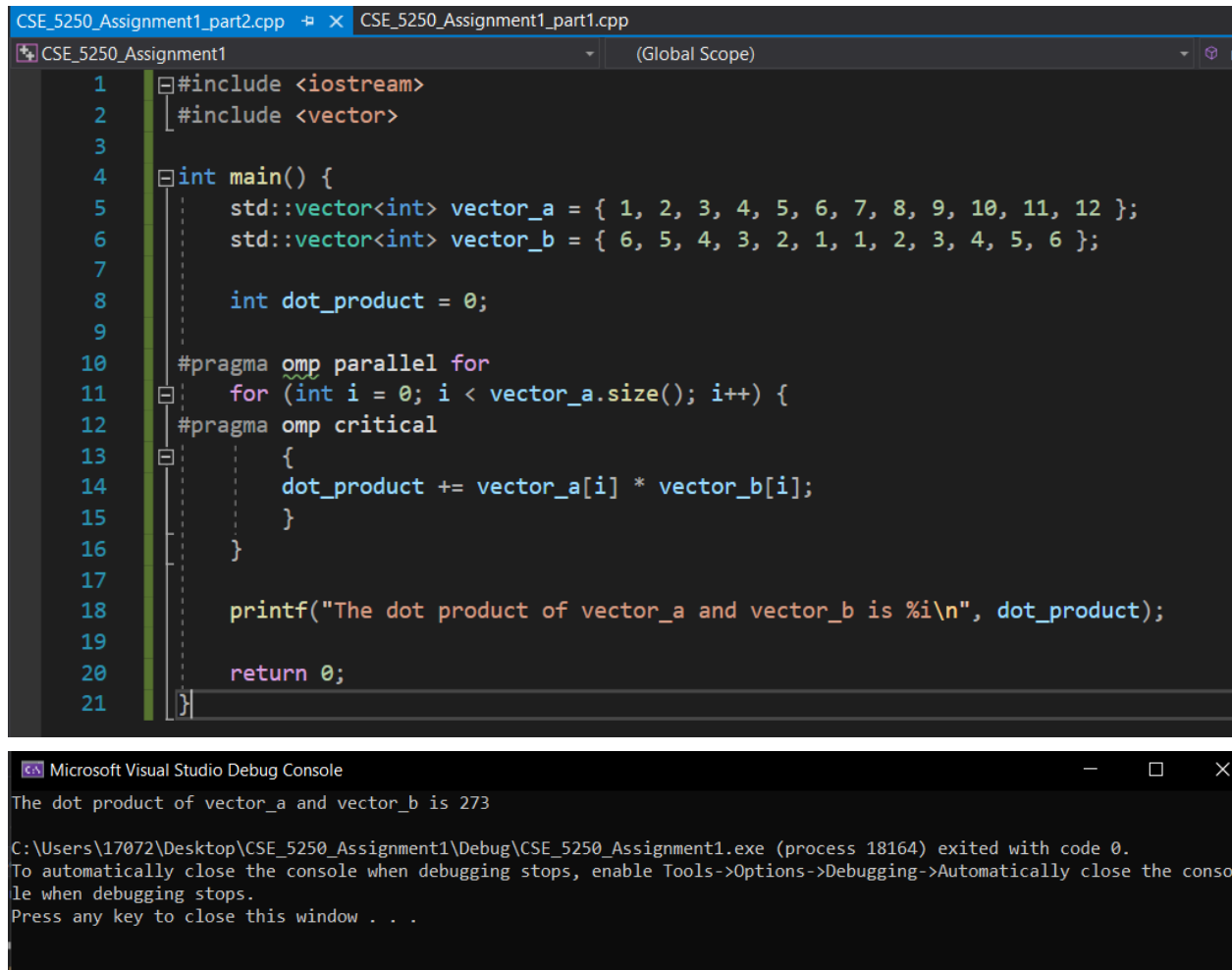
```
Microsoft Visual Studio Debug Console

Hello World from thread 0 of 6 max threads!
Hello World from thread 1 of 6 max threads!
Hello World from thread 2 of 6 max threads!
Hello World from thread 3 of 6 max threads!
Hello World from thread 4 of 6 max threads!
Hello World from thread 5 of 6 max threads!

C:\Users\17072\Desktop\CSE_5250_Assignment1\Debug\CSE_5250_Assignment1.exe (process 18076) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Part 2: Writing a parallel for loop

Write a program that calculates the dot product of two vectors, using a parallel for loop. (Note: this requires declaring part of the code a critical section). Try to have the two vectors have as many elements as your computer can handle. For example, if in part1, you found that your program will create 8 threads, then have your vectors be at least 8 elements in size.



```
CSE_5250_Assignment1_part2.cpp  CSE_5250_Assignment1_part1.cpp
CSE_5250_Assignment1 (Global Scope)
1  #include <iostream>
2  #include <vector>
3
4  int main() {
5      std::vector<int> vector_a = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 };
6      std::vector<int> vector_b = { 6, 5, 4, 3, 2, 1, 1, 2, 3, 4, 5, 6 };
7
8      int dot_product = 0;
9
10     #pragma omp parallel for
11     for (int i = 0; i < vector_a.size(); i++) {
12         #pragma omp critical
13         {
14             dot_product += vector_a[i] * vector_b[i];
15         }
16     }
17
18     printf("The dot product of vector_a and vector_b is %i\n", dot_product);
19
20     return 0;
21 }
```

```
Microsoft Visual Studio Debug Console
The dot product of vector_a and vector_b is 273

C:\Users\17072\Desktop\CSE_5250_Assignment1\Debug\CSE_5250_Assignment1.exe (process 18164) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```