Nathan Bush
CSE 5250 - SPR23
007463099

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 threa count to a different number.

Default Thread Count:

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

CSE_5250_Assignment1 | (Global Scope) | main()

```
#include <iostream>
#include <omp.h>
#include <string>

int main() {

#pragma omp parallel
    {
        int thread_num = omp_get_thread_num();
        int max_threads = omp_get_max_threads();

        printf("Hello World from thread %i of %i max threads!\n", thread_num, max_threads);
    }

    return 0;
}
```

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 consc
le 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.

CSE_5250_Assignment1.cpp

CSE_5250_Assignment1 | (Global Scope) | main()

```cpp
#include <iostream>
#include <omp.h>
#include <string>

int main() {

    omp_set_num_threads(6);

#pragma omp parallel
    {
        int thread_num = omp_get_thread_num();
        int max_threads = omp_get_max_threads();

        printf("Hello World from thread %i of %i max threads!\n", thread_num, max_threads);
    }

    return 0;
}
```

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 conso
le 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)

```cpp
#include <iostream>
#include <vector>

int main() {
    std::vector<int> vector_a = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 };
    std::vector<int> vector_b = { 6, 5, 4, 3, 2, 1, 1, 2, 3, 4, 5, 6 };

    int dot_product = 0;

#pragma omp parallel for
    for (int i = 0; i < vector_a.size(); i++) {
#pragma omp critical
        {
        dot_product += vector_a[i] * vector_b[i];
        }
    }

    printf("The dot product of vector_a and vector_b is %i\n", dot_product);

    return 0;
}
```

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 conso
le when debugging stops.
Press any key to close this window . . .
```