Sromek Mateusz
Is 3 P3

Zadanie:

1. Dopisz dyrektywę num_threads do programu z PWIR_02_01.cpp. Przetestuj czas wykonywania programu dla dwóch i więcej wątków.
Wyniki oraz program prześlij do swojego repozytorium. Umieść je w folderze o tej samie nazwie co ten PDF.

Dla dwóch wątków

```
    start = std::chrono::high_resolution_clock::now();
    for (uint32_t p = 0; p < MUL_TIME; p++) {
#pragma omp parallel for \
        if (p) num_threads(2) \ //dla dwóch wątków S
        shared(matrix) private(k, i) \
        reduction(+ : result[i])
        for (i = 0; i < MATRIX_H; i++) {
            for (k = 0; k < MATRIX_W; k++) {
                result[i] += matrix[i][k] * vector[k];
            }
        }
    }
```

Wynik:

```
PS C:\Users\user\Desktop\PWIR_05> .\pwir021.exe
Calculated normal way in 97450 miliseconds
Calculated parallel way in 60448 miliseconds
PS C:\Users\user\Desktop\PWIR_05>
```

Dla sześciu wątków:

```
    start = std::chrono::high_resolution_clock::now();
    for (uint32_t p = 0; p < MUL_TIME; p++) {
#pragma omp parallel for \
        if (p) num_threads(6) \
        shared(matrix) private(k, i) \
        reduction(+ : result[i])
        for (i = 0; i < MATRIX_H; i++) {
            for (k = 0; k < MATRIX_W; k++) {
                result[i] += matrix[i][k] * vector[k];
            }
        }
    }
    end = std::chrono::high_resolution_clock::now();

    printf("Calculated parallel way with num_threads 6 in %lu miliseconds\n",
        std::chrono::duration_cast<std::chrono::milliseconds>(end - start).count());
```

Wynik:

```
PS C:\Users\user\Desktop\PWIR_05> cd C:\Users\user\Desktop\PWIR_05
PS C:\Users\user\Desktop\PWIR_05> g++ -fopenmp PWIR_02_2.cpp -o pwir022.exe
PS C:\Users\user\Desktop\PWIR_05> .\pwir022.exe
Calculated normal way in 79433 miliseconds
Calculated parallel way in 48124 miliseconds
PS C:\Users\user\Desktop\PWIR_05>
```