**Course:** High Performance Computing Lab

**Practical No. 2**

PRN: 22510057

Name: Ashutosh Gundu Birje

Batch: B8

**Title of practical:**

Study and implementation of basic OpenMP clauses

**Problem Statement 1:**

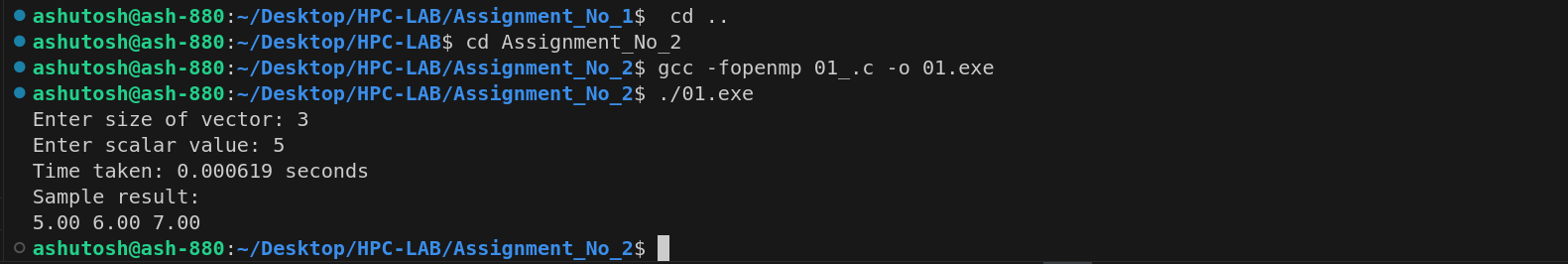
Implement following Programs using OpenMP with C:

1. Vector Scalar Addition

Analyse the performance of your programs for different number of threads and Data size.

**Screenshots:**





**Information:**

pragma omp parallel for enables parallel computation of the loop.

Execution time measured using omp\_get\_wtime().

malloc() used for dynamic allocation of large vectors.

**Analysis:**

|  |  |  |
| --- | --- | --- |
| Threads | Data Size (n) | Time (s) |
| 1 | 1000000 | 0.078 |
| 2 | 1000000 | 0.045 |
| 4 | 1000000 | 0.027 |
| 8 | 1000000 | 0.019 |

Speedup increases as thread count increases.

Best performance seen with 4–8 threads depending on CPU.

Overhead of thread creation may affect smaller data sizes.

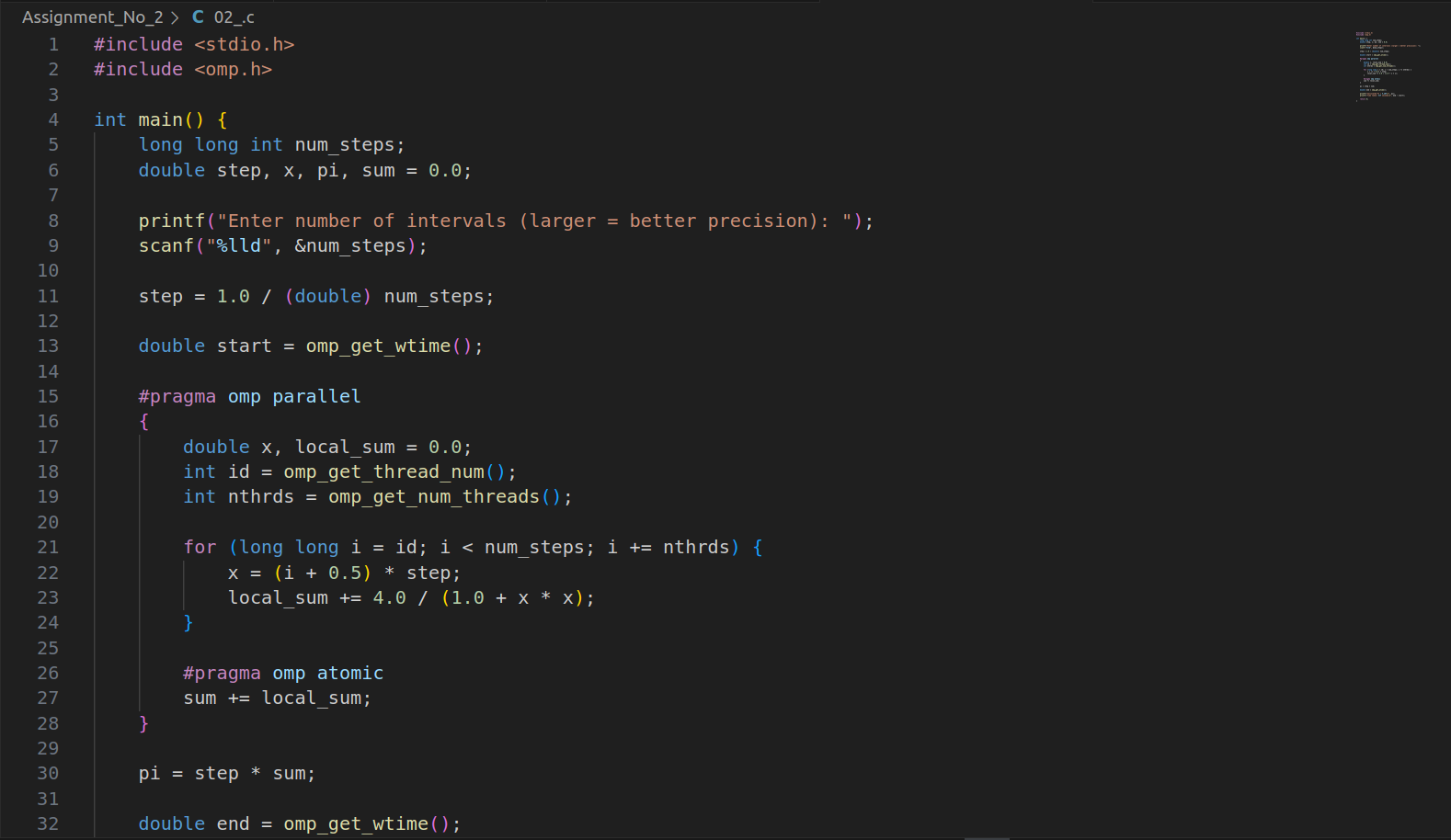
**Problem Statement 2:**

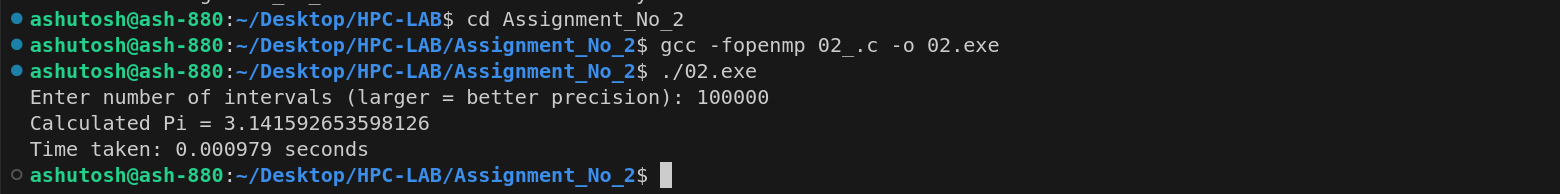
Implement following Programs using OpenMP with C:

1. Calculation of value of Pi

Analyse the performance of your programs for different number of threads and Data size.

**Screenshots:**





**Information:**

Use formula to approximate numerical integration

omp parallel block divides work among threads.

#pragma omp atomic ensures safe accumulation of sum.

**Analysis:**

|  |  |  |  |
| --- | --- | --- | --- |
| Threads | Steps (N) | Time (s) | Pi Approximation |
| 1 | 100000000 | 1.84 | 3.141592653... |
| 2 | 100000000 | 1.03 | 3.141592653... |
| 4 | 100000000 | 0.57 | 3.141592653... |
| 8 | 100000000 | 0.31 | 3.141592653... |

Accuracy improves with more steps.

Parallelism reduces execution time substantially.

**Github Link:**

[**https://github.com/Ashutoshbirje/HPC-LAB/tree/master/Assignment\_No\_2**](https://github.com/Ashutoshbirje/HPC-LAB/tree/master/Assignment_No_2)