# ASSIGNMENT HPC-3

**Roll No:** 41205

**Problem Statement:**

For Bubble Sort and Merge Sort, based on existing sequential algorithms, design, and implement parallel algorithm utilizing all resources available.

**Objective:**

1. To under basics of OpenMP
2. Apply parallel programming concepts on and sort arrays

**Outcome:** One will be able to write parallel programs to sort arrays of large sizes using OpenMP

**Pre-requisites:**

1. 64-bit Linux OS
2. Programming Languages: C/C++

**Hardware Specification:**

1. x86\_64 bit
2. 2/4 GB DDR RAM
3. 80 - 500 GB SATA HD
4. 1GB NIDIA TITAN X Graphics Card

**Software Specification:**

1. Ubuntu 14.04

# Theory:

* OpenMP is a set of compiler directives as well as an API for programs written in C, C++, or FORTRAN that provides support for parallel programming in shared-memory environments.
* OpenMP identifies parallel regions as blocks of code that may run in parallel.
* Application developers insert compiler directives into their code at parallel regions, and these directives instruct the OpenMP run-time library to execute the region in parallel.

**Syntax:**

1. Parallel creation of threads:  
     
   #pragma omp parallel
2. Create specific number of threads:  
     
   #pragma omp parallel num\_threads(count)
3. Run for loop:  
     
   #pragma omp parallel for
4. Create sections:  
     
   #pragma omp parallel sections num\_threads(3)

{

#pragma omp section

{

printf("Hello World One");

}

#pragma omp section

{

printf("Hello World Two");

}

#pragma omp section

{

printf("Hello World Three");

}

}

Running the program:

!g++ -fopenmp file.cpp

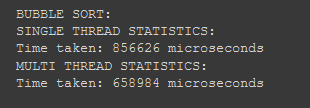
!./a.out

**Test Cases:**

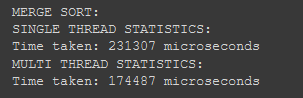
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Input | Expected Output | Actual Output | Result |
| 1 | Sort array using bubble sort | Array sorted  Multithread faster than single thread | Array sorted  Single: 856626 microseconds  Multi: 658984 microseconds | Success |
| 2 | Sort array using merge sort | Value: 26185517  Multithread faster than single thread | Array sorted  Single: 231307 microseconds  Multi: 174487 microseconds | Success |

**Output:**

Bubble Sort

****

Merge Sort

****

**Conclusion:** We were thus able to sort arrays using multithreading with the help of OpenMP using merge and bubble sort algorithms.