**OpenMP Programs**

* Write a program to sort an array on n elements using both sequential and parallel merge sort (using Section). Record the difference in execution time.
* Estimate the value of pi using:

Parallelize the code by removing loop carried dependency and record both serial and parallel execution times.

* Write an OpenMP program that divides the Iterations into chunks containing 2 iterations, respectively (OMP\_SCHEDULE=static,2). Its input should be the number of iterations, and its output should be which iterations of a parallelized for loop are executed by which thread. For example, if there are two threads and four iterations, the output might be the following:

Thread 0 : Iterations 0 −− 1

Thread 1 : Iterations 2 --- 3

* Write a program to calculate n Fibonacci numbers using Parallel Directive. Demonstrate elimination of the race condition using Schedule directive.
* Write a program to find the prime numbers from 1 to n employing parallel for directive. Record both serial and parallel execution times.
* Parallel Vector Addition
* Write a program calculate the sum of first 100 Numbers. (Using critical directive)