Assignment 3

Program: CSE

Course Code: ***411***

Course Name: ***Distributed Systems***

Student Name: ***Amr Ehab***

Student Code: ***1700923***

Assignment 3

$$cos(x) = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k}}{(2k)!}$$

In this implementation, I start by taking inputs from the user:

1- The required number of terms (accuracy).
2- The input value of x.

Then for the number of terms the numerator and denominator are calculated, divided and the result is added to the total result.

In the parallel solution, the terms are evenly divided the processes.

Example: Process with rank 0 calculates term 0 and terms 0 + n * num_of_processes.

Process 0 is responsible for dealing with user input and broadcasting it to other processes. It's also responsible for printing the output.

Example 1:

6 processes, upper value of i is 8000, value of x = 0.785

```
Sequential runtime is 0.091270 seconds
amr@Laptop-Amr:~/assignment3-distributed-systems$ mpirun -n 6  ./output
enter number of terms: 8000
enter value of x: 0.785
result is = 0.70738826916719976291032107029721487378992605954408645629882812 5
Parallel runtime is 1.523672 seconds
Sequential runtime is 8.101009 seconds
amr@Laptop-Amr:~/assignment3-distributed-systems$
```

For large number of iterations, the speedup is difference is clearly visible between the sequential code and the parallel code. Because the computations are distributed among the 6 processes.

```
amr@Laptop-Amr:~/assignment3-distributed-systems$ mpirun -n 3  ./output
enter number of terms: 8000
enter value of x: 0.785
result is = 0.70738826916719976280190085304866443038918077945709228515625 0000
Parallel runtime is 2.875349 seconds
Sequential runtime is 8.068195 seconds
```

As expected, When the number of processes was decreased to 3. The time needed to complete the 8000 iterations increases.

The speedup is clear in case of a large number of iterations. But at low numbers of iterations. The overhead of distribution of computations makes the sequential code a bit faster than the parallel code.

```
amr@Laptop-Amr:~/assignment3-distributed-systems$ mpirun -n 6  ./output
enter number of terms: 20
enter value of x: 2
result is = -0.41614683654714238706263246259098309565160889178514480590820 3125
Parallel runtime is 0.000116 seconds
Sequential runtime is 0.000003 seconds
amr@Laptop-Amr:~/assignment3-distributed-systems$
```

GitHub Repository:

https://github.com/amrehab-98/open-mpi-assignment