CS 259 - Assignment -3

Name : Hitesh Kumar

SJSU ID : 015237989

Question 1

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q1
 Hello

 GoodBye – Team Destroyed – Exiting Program
```

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q1
 Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello

 GoodBye – Team Destroyed – Exiting Program
```

Explanation :

If this program is run without -fopenmp, it ran sequqntlially and only one "hello" is printed.

If ran with **fopenmp** option**, 8 Hello** are printed since there are 8 cores in my machine,

Question 2

Thread : 2

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q2
 Hello  Hello

 GoodBye – Team Destroyed – Exiting Program
```

Thread : 4

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q2
 Hello  Hello  Hello  Hello

 GoodBye - Team Destroyed - Exiting Program
```

Thread : 8

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q2
 Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello

 GoodBye - Team Destroyed - Exiting Program
```

Thread : 5 (Odd number)

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q2
 Hello  Hello  Hello  Hello  Hello

 GoodBye - Team Destroyed - Exiting Program
```

Thread : 15

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q2
 Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello  Hello

 GoodBye - Team Destroyed - Exiting Program
```

Explanation :

As per the provided value of nthreads, we observe that we can create odd number of threads.

As per the provided value of nthreads=15, we can specify nThreads which are greater than number of cores.

Question 3

nThread= 5

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q3

 Hello from thread = 1
 Hello from thread = 4
 Hello from thread = 3
 Hello from thread = 0
 Hello from thread = 2

 GoodBye - Team Destroyed - Exiting Program
```

nThread = 8

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q3

 Hello from thread = 4
 Hello from thread = 1
 Hello from thread = 2
 Hello from thread = 3
 Hello from thread = 5
 Hello from thread = 6
 Hello from thread = 0
 Hello from thread = 7

 GoodBye - Team Destroyed - Exiting Program
```

Explanation :

There is no specific order as the threads are running parallel and are printed in random order.

nThread = 1

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % gcc-11 -o q3 -fopenmp q3.c
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q3

 Hello from thread = 0

 GoodBye – Team Destroyed – Exiting Program
```

Explanation :

There is only one thread created which is the master thread when nThread=0 is assigned.

Question 4

nthreads = 2

Thread 0 : 0-9

Thread 0 : 10-15

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q4
Hello from thread number: 0 Iteration: 0
Hello from thread number: 0 Iteration: 1
Hello from thread number: 0 Iteration: 2
Hello from thread number: 0 Iteration: 3
Hello from thread number: 0 Iteration: 4
Hello from thread number: 0 Iteration: 5
Hello from thread number: 0 Iteration: 6
Hello from thread number: 0 Iteration: 7
Hello from thread number: 1 Iteration: 8
Hello from thread number: 1 Iteration: 9
Hello from thread number: 1 Iteration: 10
Hello from thread number: 1 Iteration: 11
Hello from thread number: 1 Iteration: 12
Hello from thread number: 1 Iteration: 13
Hello from thread number: 1 Iteration: 14
Hello from thread number: 1 Iteration: 15

 GoodBye – Team Destroyed – Exiting Program
```

nthreads =4

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q4
Hello from thread number: 0 Iteration: 0
Hello from thread number: 0 Iteration: 1
Hello from thread number: 0 Iteration: 2
Hello from thread number: 0 Iteration: 3
Hello from thread number: 2 Iteration: 8
Hello from thread number: 2 Iteration: 9
Hello from thread number: 2 Iteration: 10
Hello from thread number: 2 Iteration: 11
Hello from thread number: 1 Iteration: 4
Hello from thread number: 1 Iteration: 5
Hello from thread number: 1 Iteration: 6
Hello from thread number: 1 Iteration: 7
Hello from thread number: 3 Iteration: 12
Hello from thread number: 3 Iteration: 13
Hello from thread number: 3 Iteration: 14
Hello from thread number: 3 Iteration: 15

 GoodBye – Team Destroyed – Exiting Program
```

nThreads = 8

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q4
Hello from thread number: 1 Iteration: 2
Hello from thread number: 1 Iteration: 3
Hello from thread number: 3 Iteration: 6
Hello from thread number: 3 Iteration: 7
Hello from thread number: 4 Iteration: 8
Hello from thread number: 4 Iteration: 9
Hello from thread number: 0 Iteration: 0
Hello from thread number: 0 Iteration: 1
Hello from thread number: 7 Iteration: 14
Hello from thread number: 7 Iteration: 15
Hello from thread number: 5 Iteration: 10
Hello from thread number: 5 Iteration: 11
Hello from thread number: 2 Iteration: 4
Hello from thread number: 2 Iteration: 5
Hello from thread number: 6 Iteration: 12
Hello from thread number: 6 Iteration: 13

 GoodBye – Team Destroyed – Exiting Program
```

nThreads = 16

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q4
Hello from thread number: 1 Iteration: 1
Hello from thread number: 3 Iteration: 3
Hello from thread number: 2 Iteration: 2
Hello from thread number: 4 Iteration: 4
Hello from thread number: 5 Iteration: 5
Hello from thread number: 6 Iteration: 6
Hello from thread number: 7 Iteration: 7
Hello from thread number: 8 Iteration: 8
Hello from thread number: 9 Iteration: 9
Hello from thread number: 10 Iteration: 10
Hello from thread number: 11 Iteration: 11
Hello from thread number: 13 Iteration: 13
Hello from thread number: 14 Iteration: 14
Hello from thread number: 12 Iteration: 12
Hello from thread number: 0 Iteration: 0
Hello from thread number: 15 Iteration: 15

 GoodBye – Team Destroyed – Exiting Program
```

nthread = 10

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % gcc-11 -o q4 -fopenmp q4.c
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q4
Hello from thread number: 1 Iteration: 2
Hello from thread number: 1 Iteration: 3
Hello from thread number: 2 Iteration: 4
Hello from thread number: 2 Iteration: 5
Hello from thread number: 3 Iteration: 6
Hello from thread number: 3 Iteration: 7
Hello from thread number: 6 Iteration: 12
Hello from thread number: 7 Iteration: 13
Hello from thread number: 8 Iteration: 14
Hello from thread number: 0 Iteration: 0
Hello from thread number: 0 Iteration: 1
Hello from thread number: 4 Iteration: 8
Hello from thread number: 4 Iteration: 9
Hello from thread number: 9 Iteration: 15
Hello from thread number: 5 Iteration: 10
Hello from thread number: 5 Iteration: 11

 GoodBye – Team Destroyed – Exiting Program
```

nthread =32

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q4
Hello from thread number: 1 Iteration: 1
Hello from thread number: 2 Iteration: 2
Hello from thread number: 4 Iteration: 4
Hello from thread number: 8 Iteration: 8
Hello from thread number: 9 Iteration: 9
Hello from thread number: 10 Iteration: 10
Hello from thread number: 6 Iteration: 6
Hello from thread number: 5 Iteration: 5
Hello from thread number: 3 Iteration: 3
Hello from thread number: 7 Iteration: 7
Hello from thread number: 11 Iteration: 11
Hello from thread number: 12 Iteration: 12
Hello from thread number: 13 Iteration: 13
Hello from thread number: 14 Iteration: 14
Hello from thread number: 15 Iteration: 15
Hello from thread number: 0 Iteration: 0

 GoodBye – Team Destroyed – Exiting Program
```

nThread = 5

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q4
Hello from thread number: 1 Iteration: 4
Hello from thread number: 1 Iteration: 5
Hello from thread number: 1 Iteration: 6
Hello from thread number: 0 Iteration: 0
Hello from thread number: 0 Iteration: 1
Hello from thread number: 0 Iteration: 2
Hello from thread number: 0 Iteration: 3
Hello from thread number: 2 Iteration: 7
Hello from thread number: 2 Iteration: 8
Hello from thread number: 2 Iteration: 9
Hello from thread number: 3 Iteration: 10
Hello from thread number: 3 Iteration: 11
Hello from thread number: 3 Iteration: 12
Hello from thread number: 4 Iteration: 13
Hello from thread number: 4 Iteration: 14
Hello from thread number: 4 Iteration: 15

 GoodBye – Team Destroyed – Exiting Program
```

Explanation :

The for loop iterations are split into contiguous chunks and each thread gets one chunk of iterations.

For nthread=10 value which is not evenly distributed among the loop's iteration, the threads are unevenly distributed.

For odd value of nthread=5 , the thread 0 gets the 0th chunk and the rest of the threads are unevenly distributed.

For value of nthread=32, only first 16 threads are used as the loop has only 16 iterations.

Question 5

Percent of Incorrect Answers Generated by Flawed Parallel OpenMP Program

COUNT	NTHREADS=2	NTHREADS=4	NTHREADS=8	NTHREADS=32	NTHREADS=64
10	0	0.1	0.3	0.2	0
100	0	0.2	0.4	0.4	0.4
1000	0	0.4	0.5	0.6	0.7

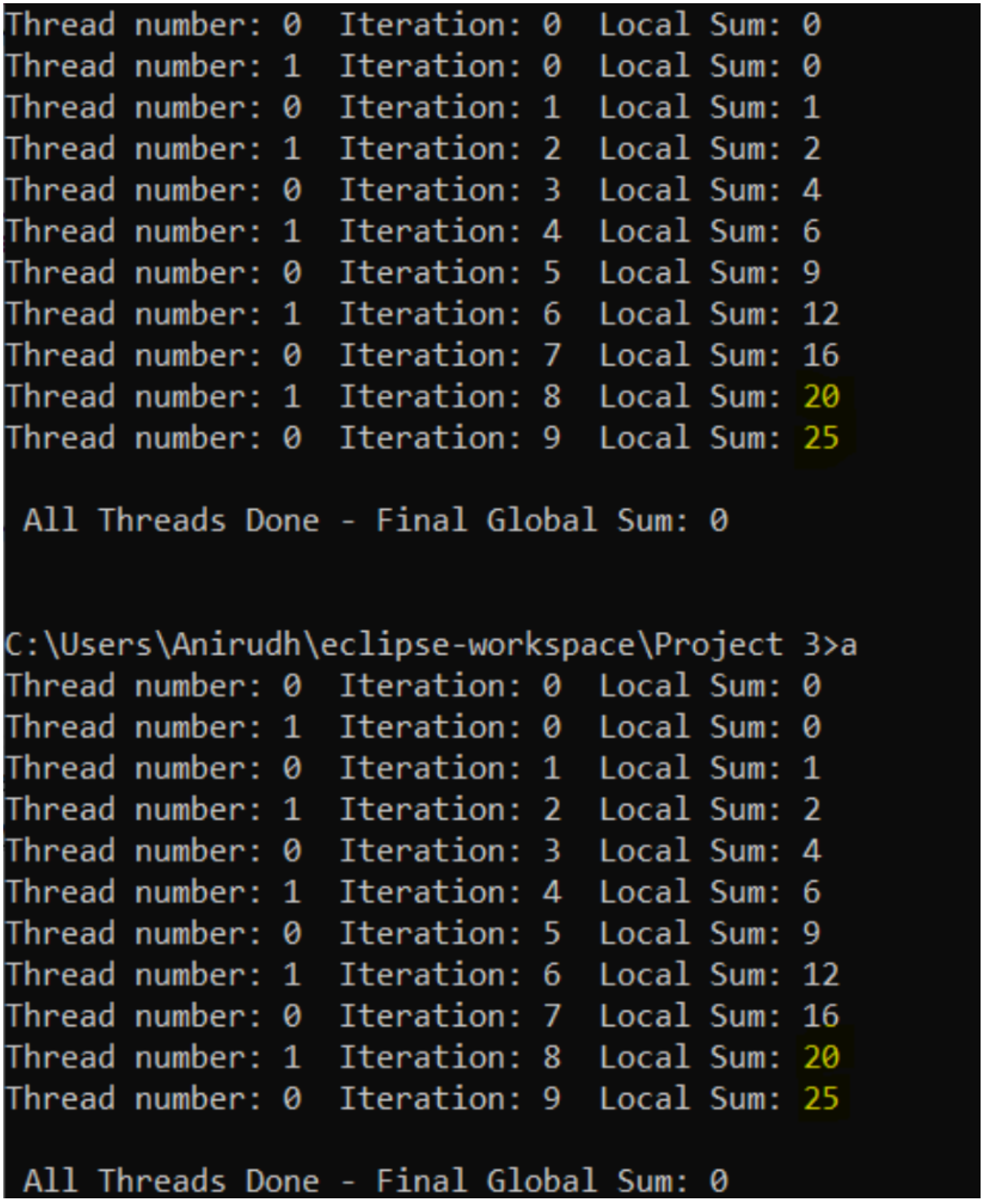
Explanation :

We get incorrect answers due to the degree of parallelization. The variable 'sum' is a shared variable and when we do parallelization, its value is accessed multiple times from various threads.Multiple race conditions occur due to data hazard.

As we increase the number of threads, the probability of getting wrong answer increases.

Hence, the probability of getting an incorrect answer depends directly on the values of COUNT and nThreads.

Question 6



Explanation :

Since sum is in private, the global sum is displayed as 0 irrespective of nThread value. The program ran for nThread = 2, 4,8,16 , 32, the sum value is incorrect showing that the program will not resolve the non-deterministic issue.

Question 7

nthread =4 , Count = 10

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % gcc-11 -o q7 -fopenmp q7.c
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % ./q7
Thread number: 1  Iteration: 3  Local Sum: 3
Thread number: 1  Iteration: 4  Local Sum: 7
Thread number: 1  Iteration: 5  Local Sum: 12
Thread number: 3  Iteration: 8  Local Sum: 8
Thread number: 3  Iteration: 9  Local Sum: 17
Thread number: 0  Iteration: 0  Local Sum: 0
Thread number: 0  Iteration: 1  Local Sum: 1
Thread number: 0  Iteration: 2  Local Sum: 3
Thread number: 2  Iteration: 6  Local Sum: 6
Thread number: 2  Iteration: 7  Local Sum: 13

 All Threads Done – Final Global Sum: 45
```

nthread =8 , Count = 100

```
Thread number: 0  Iteration: 10  Local Sum: 55
Thread number: 0  Iteration: 11  Local Sum: 66
Thread number: 0  Iteration: 12  Local Sum: 78
Thread number: 4  Iteration: 56  Local Sum: 270
Thread number: 4  Iteration: 57  Local Sum: 327
Thread number: 4  Iteration: 58  Local Sum: 385
Thread number: 4  Iteration: 59  Local Sum: 444
Thread number: 4  Iteration: 60  Local Sum: 504
Thread number: 4  Iteration: 61  Local Sum: 565
Thread number: 4  Iteration: 62  Local Sum: 627
Thread number: 4  Iteration: 63  Local Sum: 690

 All Threads Done – Final Global Sum: 4950
```

Explanation :

Now each thread has a local copy of the reduction “sum” variable which means each iteration will update it and will display the updated value.

Question 8

On running the script, the output is below :

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % time ./q8
The value of PI is  3.141592653590
./q8  3.48s user 0.01s system 99% cpu 3.497 total
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 %
```

Question 9

The below experiments are done on a 8 core Mac M1 machine.

nThread = 2

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % time ./q9
The value of PI is  3.141592653590
./q9  3.64s user 0.01s system 178% cpu 2.041 total
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 %
```

nThread = 4

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % time ./q9
The value of PI is  3.141592653590
./q9  3.84s user 0.01s system 336% cpu 1.142 total
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 %
```

nThread = 8

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % time ./q9
The value of PI is  3.141592653590
./q9  4.41s user 0.02s system 543% cpu 0.816 total
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 %
```

nThread = 16

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % time ./q9
The value of PI is  3.141592653590
./q9  4.38s user 0.02s system 574% cpu 0.767 total
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 %
```

nThread = 32

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % time ./q9
The value of PI is  3.141592653590
./q9  4.35s user 0.02s system 657% cpu 0.666 total
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 %
```

nThread = 34

```
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 % time ./q9
The value of PI is  3.141592653590
./q9  4.36s user 0.03s system 598% cpu 0.733 total
(base) hiteshkumar@Hiteshs-MacBook-Pro Assignment-3 %
```

Explanation :

nThreads	Time
2	2.041
4	1.142
8	0.816
16	0.767
32	0.666
34	0.733

When nThreads =8, the execution has drastically decreased as compared to nThread =2.

Going nThread >32, the execution time increased again due to overhead cost as we spawn more threads. So the bottom line is, we will see an increase in the execution time if number of threads > number of cores in the machine.