**Class:** Final Year (Computer Science and Engineering)

**Year:** 2022-23 **Semester:** 1

**Course:** High Performance Computing Lab

**Practical No. 1**

**Exam Seat No: 2019BTECS00064**

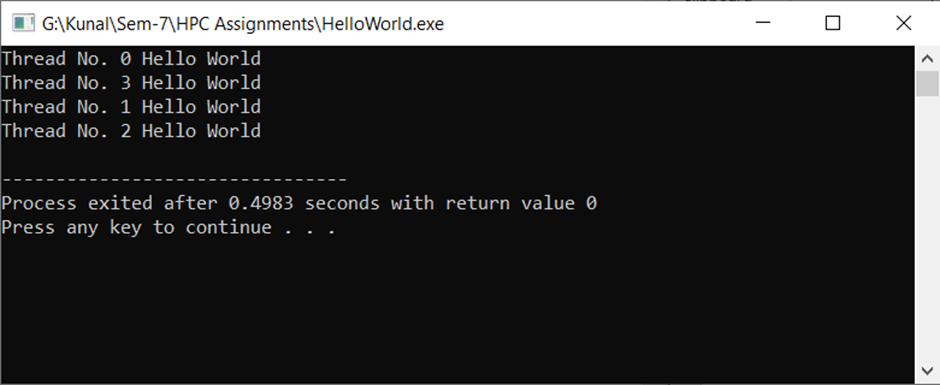
**Name – Kunal Santosh Kadam**

**Title of practical: Use of OpenMP in C**

**Problem Statement 1:**

Write a program that prints Hello World using OpenMP

**Screenshot #:**



**Information #:**

#include<omp.h>

#include<bits/stdc++.h>

int main(int argc, char\* argv[])

{

#pragma omp parallel

{

printf("Thread No. %d Hello World\n", omp\_get\_thread\_num());

}

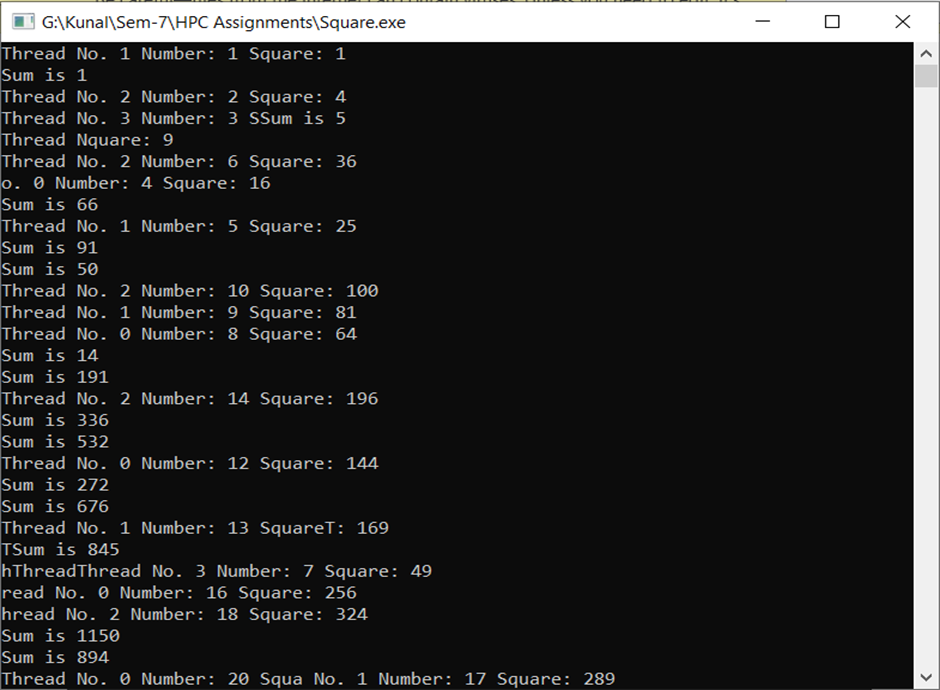
return 0;

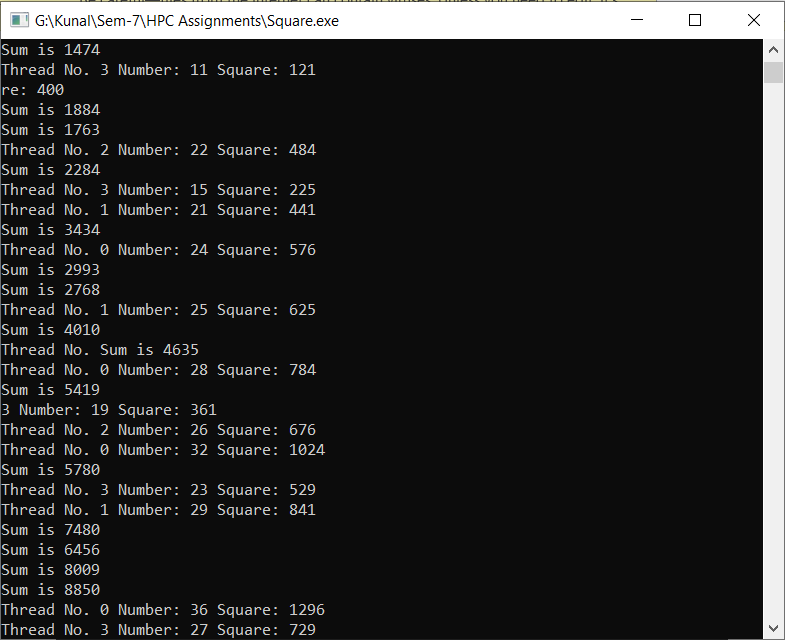
}

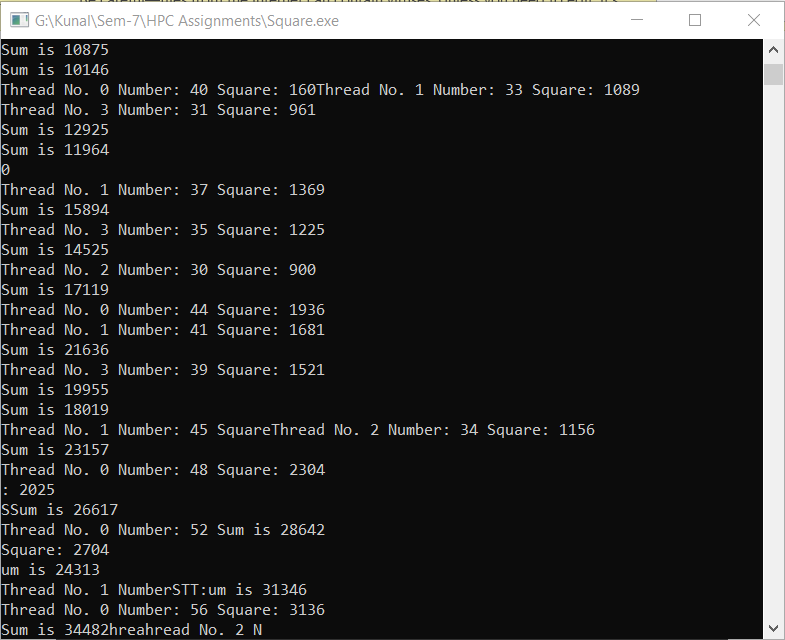
**Problem Statement 2:**

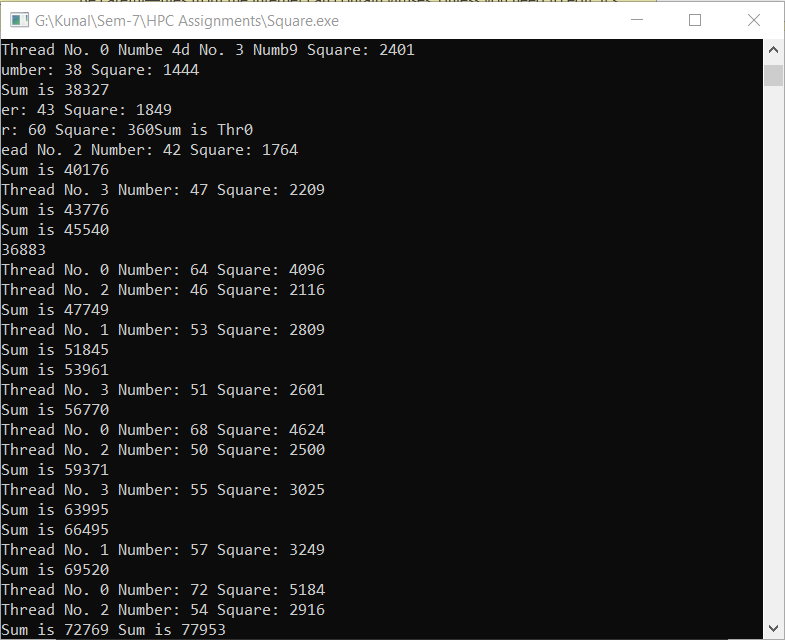
Write a program that print the square an their sum using the OpenMP

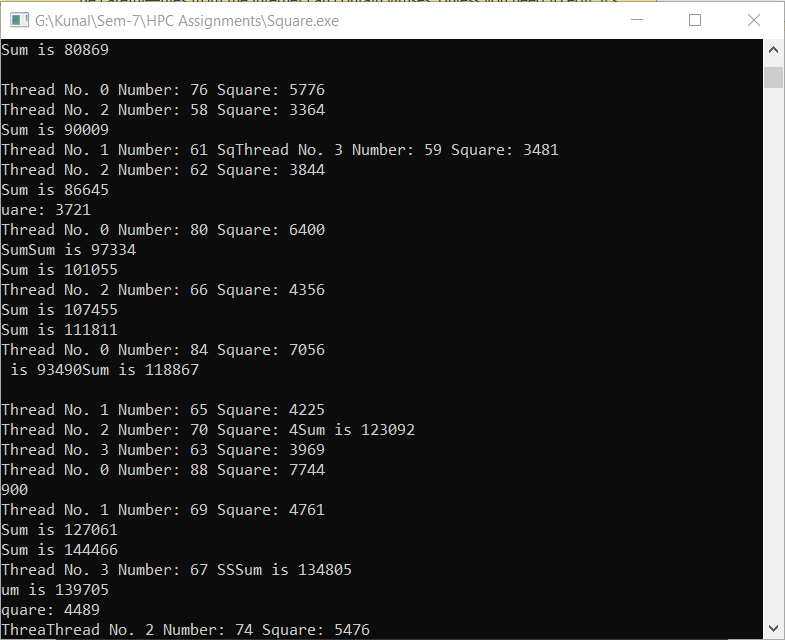
**Screenshot #:**

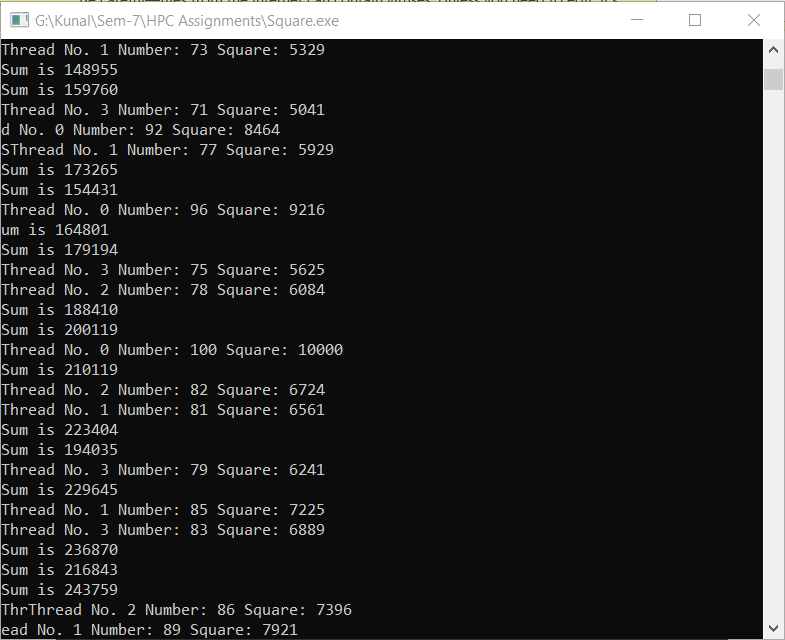


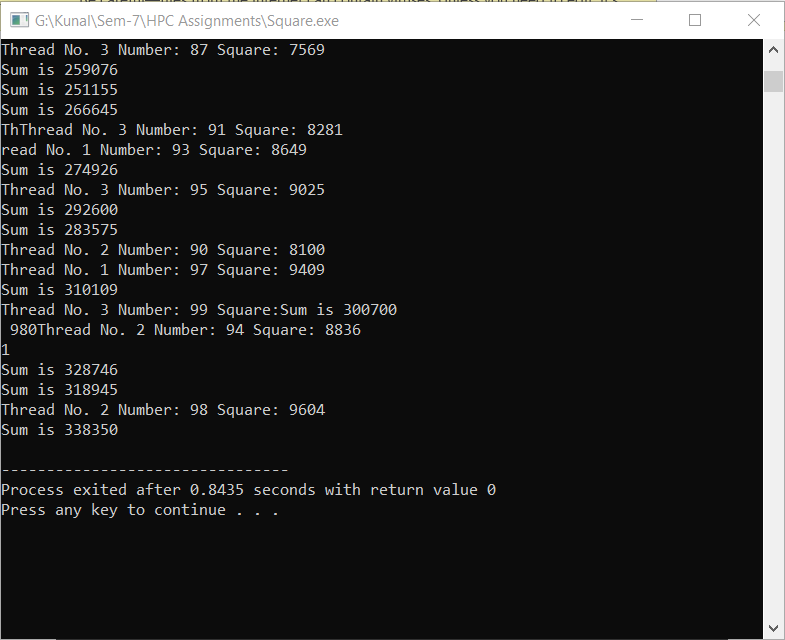












**Information #:**

#include<omp.h>

#include<bits/stdc++.h>

static int sum = 0;

int main()

{

#pragma omp parallel

{

for(int i = 1;i <= 100;i++)

{

if(i % 4 == omp\_get\_thread\_num())

{

printf("Thread No. %d Number: %d Square: %d \n",omp\_get\_thread\_num(), i , i\*i);

sum += i\*i;

printf("Sum is %d \n" ,sum);

}

}

}

return 0;

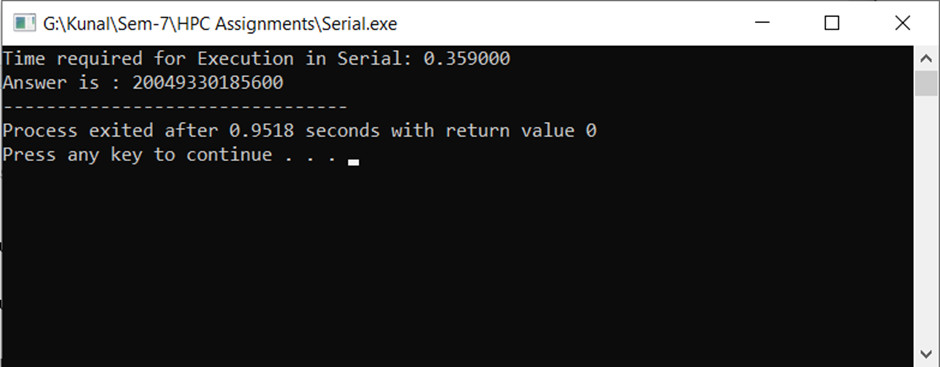
}

**Problem Statement 3:**

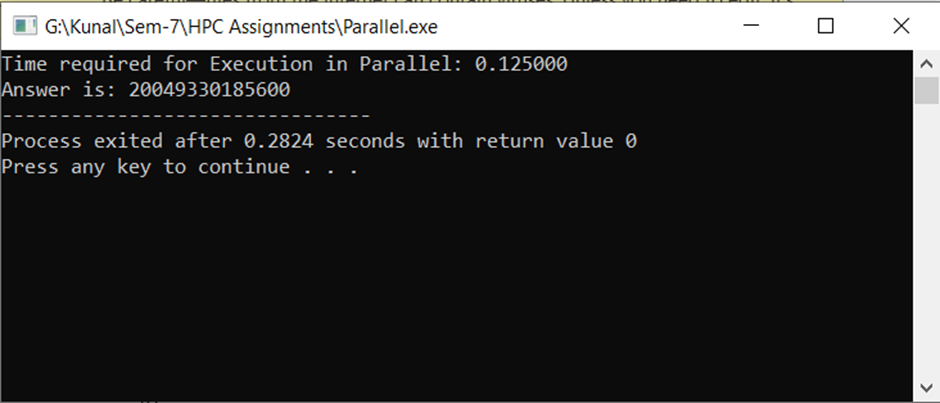
Write a Serial and Parallel Program for Addition of Square and see the difference between the Speedup

**Screenshot #:**

Serial Code:



Parallel Code:



**Information #:**

**Serial Code:**

#include<bits/stdc++.h>

#include<omp.h>

int main()

{

long long sum =0 ;

double inTime = omp\_get\_wtime();

int i;

for(i=1;i <= 100000000;i++)

sum += (i\*i);

double outTime = omp\_get\_wtime();

double expcTime = outTime - inTime;

printf("Time required for Execution in Serial: %f\n", expcTime);

printf("Answer is : %lld",sum);

return 0;

}

**Parallel Code:**

#include<bits/stdc++.h>

#include<omp.h>

int main()

{

long long sum = 0;

double getInTime = omp\_get\_wtime();

#pragma omp parallel for reduction(+ : sum)

for (int i=1;i <= 100000000; i++)

sum += i\*i;

double getOutTime = omp\_get\_wtime();

double exptTime= getOutTime - getInTime;

printf("Time required for Execution in Parallel: %f\n", exptTime);

printf("Answer is: %lld", sum);

return 0;

}

**Github Link:**