Sofia University
Department of Mathematics and Informatics

Course : **OO Programming with C#.NET**

Date: **November 7, 2014**

Student Name:

Lab No. 6b

Problem No. 1a

Run the Task Manager and open the Performance Wizard to view CPU usage on your computer. Run the attached C# project **MultMatricesInTPL**. Experiment increasing the number of columns and rows in the matrices. The larger the matrices, the greater the performance difference between the parallel and sequential versions of the computation. When the matrix is small, the sequential version will run faster because of the overhead in setting up the parallel loop

Problem No. 1b

Write a Console application to **multiply a matrix by a row** using parallel For. Compare the execution of a sequential and a parallel execution of the task.

Problem No. 1c

Run the attached project **ReturnAverageTPL**. Modify its code so that the parallel **For** returns the average value of all the sums generated inside the loop.

Problem No. 2

Given

```
var customers new[] {
         new { ID = 1,  FirstName = "Sandeep"  , LastName = "Ramani" },
         new { ID = 2,  FirstName = "Dharmik"  , LastName = "Chotaliya" },
         new { ID = 3,  FirstName = "Nisar"    ,  LastName = "Kalia" } ,
         new { ID = 4,  FirstName = "Ravi"     , LastName = "Mapara" } ,
         new { ID = 5,  FirstName = "Hardik"   , LastName = "Mistry" }
         new { ID = 6,  FirstName = "Sandy"    , LastName = "Ramani" },
         new { ID = 7,  FirstName = "Jigar"    , LastName = "Shah" },
         new { ID = 8,  FirstName = "Kaushal"  , LastName = "Parik" } ,
         new { ID = 9,  FirstName = "Abhishek" , LastName = "Swarnker" } ,
```

```
        new { ID = 10, FirstName = "Sanket"   , LastName = "Patel" }
        new { ID = 11, FirstName = "Dinesh"   , LastName = "Prajapati" },
        new { ID = 12, FirstName = "Jayesh"   , LastName = "Patel" },
        new { ID = 13, FirstName = "Nimesh"   , LastName = "Mishra" } ,
        new { ID = 14, FirstName = "Shiva"    , LastName = "Reddy" } ,
        new { ID = 15, FirstName = "Jasmin"   , LastName = "Malviya" }
        new { ID = 16, FirstName = "Haresh"   , LastName = "Bhanderi" },
        new { ID = 17, FirstName = "Ankit"    , LastName = "Ramani" },
        new { ID = 18, FirstName = "Sanket"   , LastName = "Shah" } ,
        new { ID = 19, FirstName = "Amit"     , LastName = "Shah" } ,
        new { ID = 20, FirstName = "Nilesh"   , LastName = "Soni" }       };
```

a) Write a PLINQ query to select the customers with IDs in the range between 5 and 15, while preserving the order of the IDs in the output

b) Write a PLINQ query to select the customers with distinct LastName

c) Write a PLINQ query to select the customer ID, and the FirstName and LastName concatenated by a comma and a space between them

Use Parallel.ForEach and ParallelQuery ForAll methods to output the results