Matrix Multiplication

```
//**********************************************************************
// Programming F#
// Example 11-10 Using Parallel.For
// Modified code. HF
//**********************************************************************
open System
open System.IO
open System.Net
open System.Threading.Tasks


let matrixMultiply (a: float[,]) (b: float[,]) =
    let aRow = Array2D.length1 a
    let aCol = Array2D.length2 a
    let bRow = Array2D.length1 b
    let bCol = Array2D.length2 b
    if aCol <> bRow then failwith "Array dimension mismatch."

    // allocate space for the resulting matrix c
    let c = Array2D.create aRow bCol 0.0
    let cRow = aRow
    let cCol = bCol

    // Compute each row of the resulting matrix
    let rowTask rowIdx = for colIdx = 0 to cCol - 1 do
                             for x = 0 to aRow - 1 do
                                 c.[rowIdx, colIdx] <- c.[rowIdx, colIdx] +
                                                       a.[rowIdx, x] *
                                                       b.[x, colIdx]
                         ()

    // Compute each row in parallel
    let _ = Parallel.For(0, cRow, new Action<int> (rowTask))

    // Return the matrix
    c


//**********************************************************************
// Program
//**********************************************************************
[<EntryPoint>]
let main argv =

    // Example 1
    let a: float[ , ] = Array2D.zeroCreate 2 2
    a.[0,0] <- 1.0
    a.[0,1] <- 2.0
    a.[1,0] <- 3.0
    a.[1,1] <- 4.0

    let b: float[ , ] = Array2D.zeroCreate 2 2
    b.[0,0] <- 5.0
    b.[0,1] <- 6.0
    b.[1,0] <- 7.0
    b.[1,1] <- 8.0

    let c = matrixMultiply a b
    printfn "%A" c
```

```
printfn ""

// Example 2
a.[0,0] <- 4.0
a.[0,1] <- 2.0
a.[1,0] <- -3.0
a.[1,1] <- 1.0

b.[0,0] <- 1.0
b.[0,1] <- 5.0
b.[1,0] <- 2.0
b.[1,1] <- 7.0

let c = matrixMultiply a b
printfn "%A" c
printfn ""

// Example 3
let d: float[ , ] = Array2D.zeroCreate 2 3
d.[0,0] <- 1.0
d.[0,1] <- 5.0
d.[0,2] <- 3.0
d.[1,0] <- 2.0
d.[1,1] <- 7.0
d.[1,2] <- -4.0

let c = matrixMultiply a d
printfn "%A" c
printfn ""


Console.ReadKey(true) |> ignore
0
```