

# Proiect APD

## Repository:

<https://github.com/raressclipceadaniel14/APD>

**Tema:** Procesarea de imagini (en., Image Processing)

Limbaj folosit: C#

Framework: .NET

Library: System.Drawing

```
using System.Diagnostics;
using System.Drawing;

class Program
{
    static void Main()
    {
        string imagePath = "D:\\Facultate\\Proiect APD\\ProiectAPD\\ProiectAPD\\input.jpg";
        Bitmap originalImage = new Bitmap(imagePath);

        Stopwatch sequentialStopwatch = Stopwatch.StartNew();
        string outputFilePath = "D:\\Facultate\\Proiect APD\\ProiectAPD\\ProiectAPD\\output.jpg";
        ProcessImageSequentially(originalImage, outputFilePath);
        sequentialStopwatch.Stop();
        Console.WriteLine("Sequential processing time: " + sequentialStopwatch.ElapsedMilliseconds + " ms");
    }

    static void ProcessImageSequentially(Bitmap original, string outputFilePath)
    {
        // Create a new bitmap for the result
        Bitmap result = new Bitmap(200, 200);

        // Loop through each pixel of the original image
        for (int x = 0; x < original.Width; x++)
        {
            for (int y = 0; y < original.Height; y++)
            {
                // Get the color of the pixel in the original image
                Color pixelColor = original.GetPixel(x, y);

                // Filter to black and white
                int avgColor = (pixelColor.R + pixelColor.G + pixelColor.B) / 3;
                Color bwColor = Color.FromArgb(avgColor, avgColor, avgColor);

                // Resize the image to 200x200 pixels
                int newX = (int)((double)x / original.Width * 200);
                int newY = (int)((double)y / original.Height * 200);

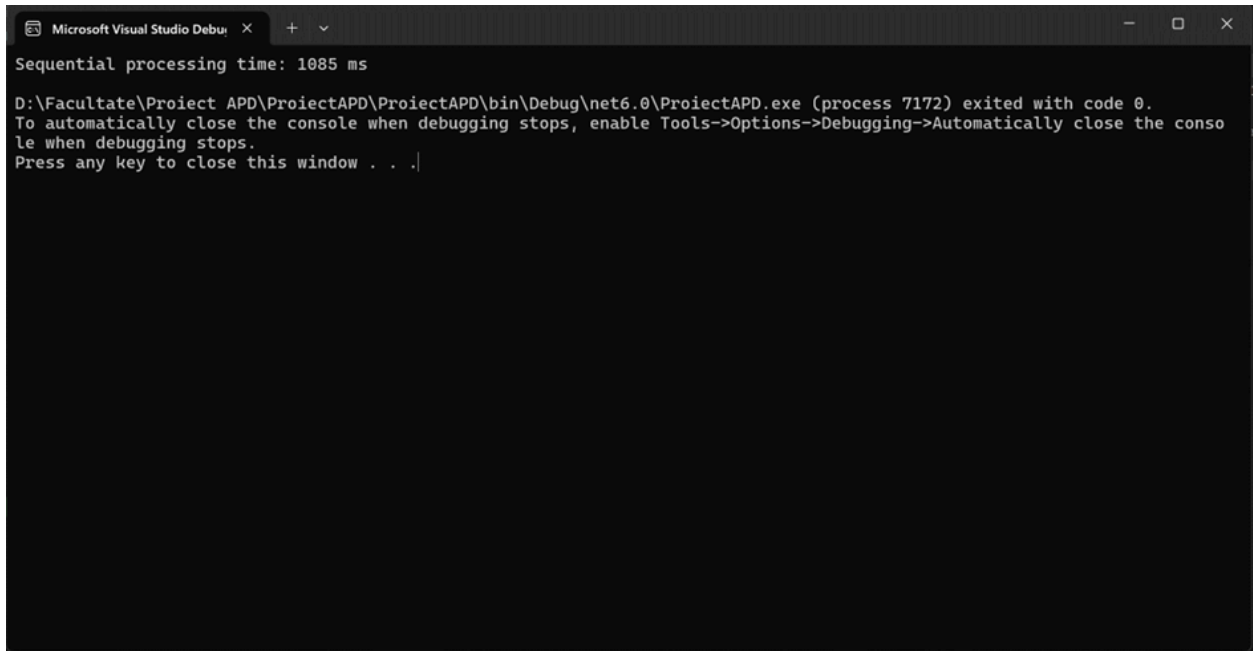
                // Convert all blue pixels to red
                if (pixelColor.B > 0)
                    pixelColor = Color.FromArgb(pixelColor.R, pixelColor.G, 255); // Red color for blue pixels

                // Set the color of the corresponding pixel in the result image
                result.SetPixel(newX, newY, bwColor);
            }
        }

        // Save the processed image
        result.Save(outputFilePath);
    }
}
```

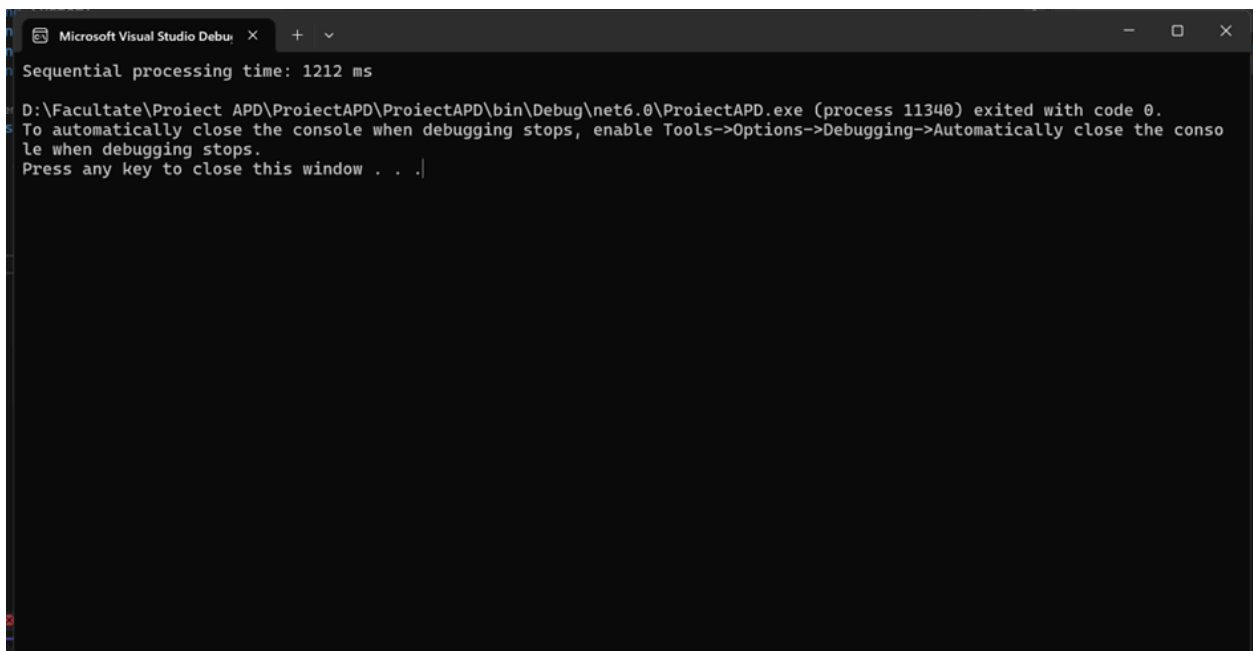
Codul a fost rulat pe: i5 11300h, 16gb ram, rtx 3050 4gb si avem urmatoarele rezultate:

Pentru resize la 200x200px + black and white filtering:



```
Microsoft Visual Studio Debug Console
Sequential processing time: 1085 ms
D:\Facultate\Proiect APD\ProiectAPD\ProiectAPD\bin\Debug\net6.0\ProiectAPD.exe (process 7172) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Pentru resize la 200x200px + schimbare toti pixelii albastrii in rosii:



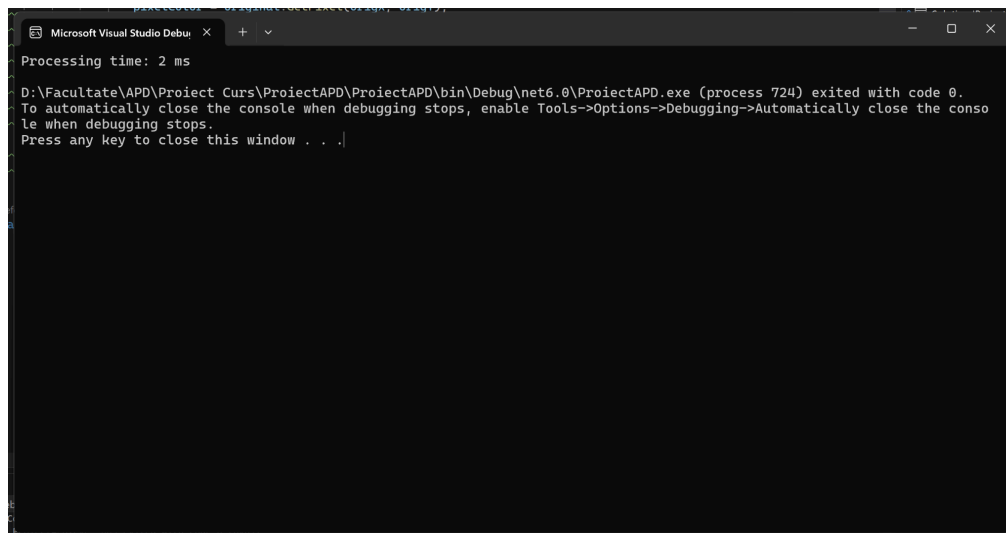
```
Microsoft Visual Studio Debug Console
Sequential processing time: 1212 ms
D:\Facultate\Proiect APD\ProiectAPD\ProiectAPD\bin\Debug\net6.0\ProiectAPD.exe (process 11340) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Varianta paralela:

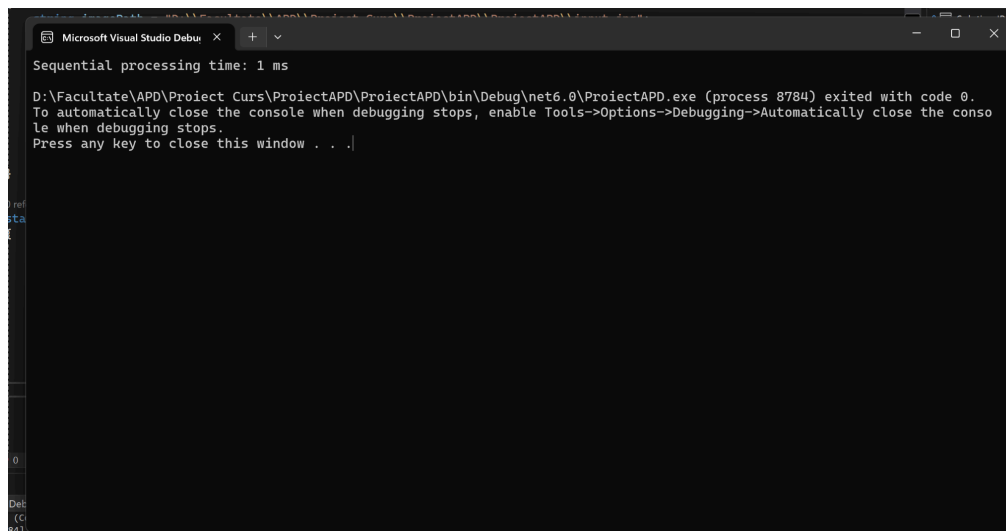
Am folosit `System.Threading.Tasks.Task` pentru a face cele 3 operatii separat in paralel.

Am folosit pentru iterat intre pixelii imaginii `System.Threading.Tasks.Parallel`, iar pentru a itera prin `Bitmap` si pe `height` si pe `width` in acelasi timp si a evita accesarea unui pixel in acelasi timp de ambele iteratii, am folosit locks.

Resultatele in eficienta sunt total diferite:



```
Microsoft Visual Studio Debug Console
Processing time: 2 ms
D:\Facultate\APD\Proiect Curs\ProiectAPD\ProiectAPD\bin\Debug\net6.0\ProiectAPD.exe (process 724) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```



```
Microsoft Visual Studio Debug Console
Sequential processing time: 1 ms
D:\Facultate\APD\Proiect Curs\ProiectAPD\ProiectAPD\bin\Debug\net6.0\ProiectAPD.exe (process 8784) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

## **Varianta cu Libraria PLINQ:**

În varianta paralelă cu PLINQ, procesul de prelucrare a imaginii este împărțit în etape pe care le putem realiza în același timp. Iată cum funcționează:

Extragem datele de pixeli: Înainte să începem procesarea, luăm datele despre pixeli din imaginea originală și le stocăm într-un tablou special. Acest tablou conține informații despre fiecare pixel, cum ar fi culoarea și intensitatea.

Procesăm în paralel cu PLINQ: Folosim PLINQ, o unealtă care ne ajută să procesăm datele în paralel. Asta înseamnă că mai multe bucăți ale imaginii sunt procesate simultan, pe fire de execuție separate. Astfel, transformăm imaginea în alb-negru, o redimensionăm și facem alte modificări dorite.

Generăm o nouă imagine: După ce am terminat procesarea, luăm datele modificate și le folosim pentru a crea o imagine nouă. Această imagine conține modificările făcute și poate fi salvată pe disk sau folosită în continuare în program.

Prin această abordare, evităm conflictele și erorile care pot apărea când lucrăm cu date concurente. Totodată, prelucrarea în paralel ne ajută să accelerăm procesul, făcând lucrurile mai rapid și mai eficiente.

**Tabel cu exemple de rulare:**

<b>Input</b>	<b>Secvential</b>	<b>TPL (ms)</b>	<b>PLINQ (ms)</b>
input980x653	1245ms	1	24ms
input1000x1000	3546ms	1	71ms
input2000x2000	10398ms	2	207ms
input3000x3000	30857ms	3	617ms
input4000x4000	60298ms	4	1636ms
input7000x7000	128545ms	5-6	81801ms
input10000x10000	204503ms	6-7	4090ms
input15000x15000	385517ms	8	7710ms