Berikut adalah **penjelasan baris demi baris** dari program di atas 👇

```
using System.Drawing.Drawing2D;
```

➡️ Mengimpor namespace `System.Drawing.Drawing2D` yang menyediakan fungsi tambahan untuk manipulasi grafis dua dimensi, seperti path, brush, dan transformasi. (Meskipun di kode ini tidak digunakan langsung, biasanya diperlukan untuk pengolahan gambar).

```
namespace MiniPhotoshop.Logic.Histogram
```

➡️ Mendefinisikan namespace tempat kelas ini berada, yaitu `MiniPhotoshop.Logic.Histogram`. Namespace berfungsi sebagai pengelompokan logis dari kode program yang berkaitan dengan logika perhitungan histogram.

```
public static class HistogramCalculator
```

➡️ Mendefinisikan kelas bernama `HistogramCalculator` sebagai **static**, artinya tidak perlu membuat objek untuk menggunakannya. Semua metode di dalamnya juga harus bersifat static.

```
// Menghitung array histogram dari Bitmap untuk channel tertentu
```

➡️ Komentar yang menjelaskan tujuan metode berikutnya, yaitu menghitung histogram dari gambar (`Bitmap`) untuk saluran warna tertentu (merah, hijau, biru, atau grayscale).

```
public static int[] Calculate(Bitmap bmp, int channel)
```

➡️ Mendefinisikan metode static `Calculate` yang menerima dua parameter:

- `bmp` → objek gambar bertipe `Bitmap`
- `channel` → angka penanda channel warna (0=Red, 1=Green, 2=Blue, 3=Grayscale).
  Metode ini mengembalikan **array integer** (`int[]`) berisi data histogram.

```
int[] hist = new int[256];
```

➡️ Membuat array `hist` berukuran 256 elemen untuk menyimpan jumlah kemunculan setiap nilai intensitas piksel (0–255).

```
for (int y = 0; y < bmp.Height; y++)
```

➡ Memulai perulangan vertikal dari baris pertama (`y = 0`) hingga baris terakhir (`y < bmp.Height`).

```
for (int x = 0; x < bmp.Width; x++)
```

➡ Di dalam loop `y`, dilakukan loop horizontal dari kolom pertama (`x = 0`) hingga kolom terakhir (`x < bmp.Width`). Dengan ini, setiap piksel gambar akan diproses satu per satu.

```
Color pixelColor = bmp.GetPixel(x, y);
```

➡ Mengambil warna dari piksel pada posisi `(x, y)` menggunakan metode `GetPixel()` dan menyimpannya dalam variabel `pixelColor` bertipe `Color`.

```
int val = 0;
```

➡ Menginisialisasi variabel `val` yang akan digunakan untuk menyimpan nilai intensitas warna piksel sesuai channel yang dipilih.

```
switch (channel)
```

➡ Struktur kontrol `switch` digunakan untuk menentukan channel warna mana yang akan dihitung berdasarkan nilai parameter `channel`.

```
case 0: // Red
    val = pixelColor.R;
    break;
```

➡ Jika channel = 0, ambil komponen warna **merah (R)** dari piksel.

```
case 1: // Green
    val = pixelColor.G;
    break;
```

➡ Jika channel = 1, ambil komponen warna **hijau (G)** dari piksel.

```
case 2: // Blue
    val = pixelColor.B;
    break;
```

➡ Jika channel = 2, ambil komponen warna **biru (B)** dari piksel.

```
case 3: // Grayscale (Penting: gunakan rumus Grayscale)
    val = (int)(0.3 * pixelColor.R + 0.59 * pixelColor.G + 0.11 *
pixelColor.B);
    break;
```

➡ Jika channel = 3, hitung nilai **grayscale** piksel menggunakan rumus standar pencampuran:
[
Grayscale = 0.3R + 0.59G + 0.11B
]
Kemudian hasilnya dikonversi ke `int`.

```
default:
    val = pixelColor.R;
    break;
```

➡ Jika nilai `channel` tidak sesuai dengan pilihan yang ada, maka secara default digunakan channel **Red**.

```
// Pastikan val berada dalam batas [0, 255]
```

➡ Komentar yang menjelaskan bahwa nilai `val` harus berada antara 0 sampai 255 agar valid.

```
if (val >= 0 && val <= 255)
{
    hist[val]++;
}
```

➡ Mengecek apakah nilai intensitas valid. Jika ya, maka **menambah 1** pada posisi `hist[val]` yang sesuai — ini berarti nilai intensitas tersebut muncul sekali lagi pada gambar.

```
return hist;
```

➡️ Setelah semua piksel diproses, metode mengembalikan array histogram (`hist`) yang berisi distribusi intensitas warna.

🔍 Kesimpulan

Kode ini membaca setiap piksel dari gambar, mengambil nilai warna sesuai channel (R, G, B, atau grayscale), dan menghitung berapa kali setiap nilai 0–255 muncul. Hasil akhirnya adalah histogram digital yang menggambarkan distribusi intensitas warna dari gambar tersebut.