**Task 1-A program that binarizes an image.**

**Matriculation Number: 10004430**

Date Of submission: 26-06-2025

Introduction:

In this report we are doing image binarization. In this particular process we are converting a grey scale image to two different thresholding i.e.,

1. Global Adaptive Thresholding

2. Local Adaptive Thresholding.

# Methodology:

**1.Global Adaptive Thresholding-**

Global Adaptive Thresholding is a technique used to convert a grayscale image into a binary image (black and white) based on a *single threshold value* computed from the image itself. Let’s break it down intuitively:

**What It Does**

Instead of using a fixed threshold (like 127 out of 255), global adaptive thresholding **calculates the best threshold** based on the image's content. It adapts to the image’s pixel intensities by refining the threshold over several iterations.

**2. Local Adaptive Thresholding-**

Local Adaptive Thresholding is a technique used to binarize an image based on **local statistics**—meaning the threshold value is calculated for each pixel using its surrounding neighbourhood rather than using a global value.

**Why Use Local Adaptive Thresholding?**

It’s especially useful when the image has:

* **Uneven lighting**
* **Shadows or gradients**
* **Variable background intensity**

Unlike global thresholding, this method adapts to brightness changes over small regions, preserving fine details better in diverse lighting conditions.

# Algorithm:

**1.Global Adaptive Thresholding-**

* Initial threshold = mean image intensity
* Iterative refinement using class means μ1,μ2\mu\_1, \mu\_2μ1​,μ2​
* Converges when ΔT<ϵ\Delta T < \epsilon<ϵ

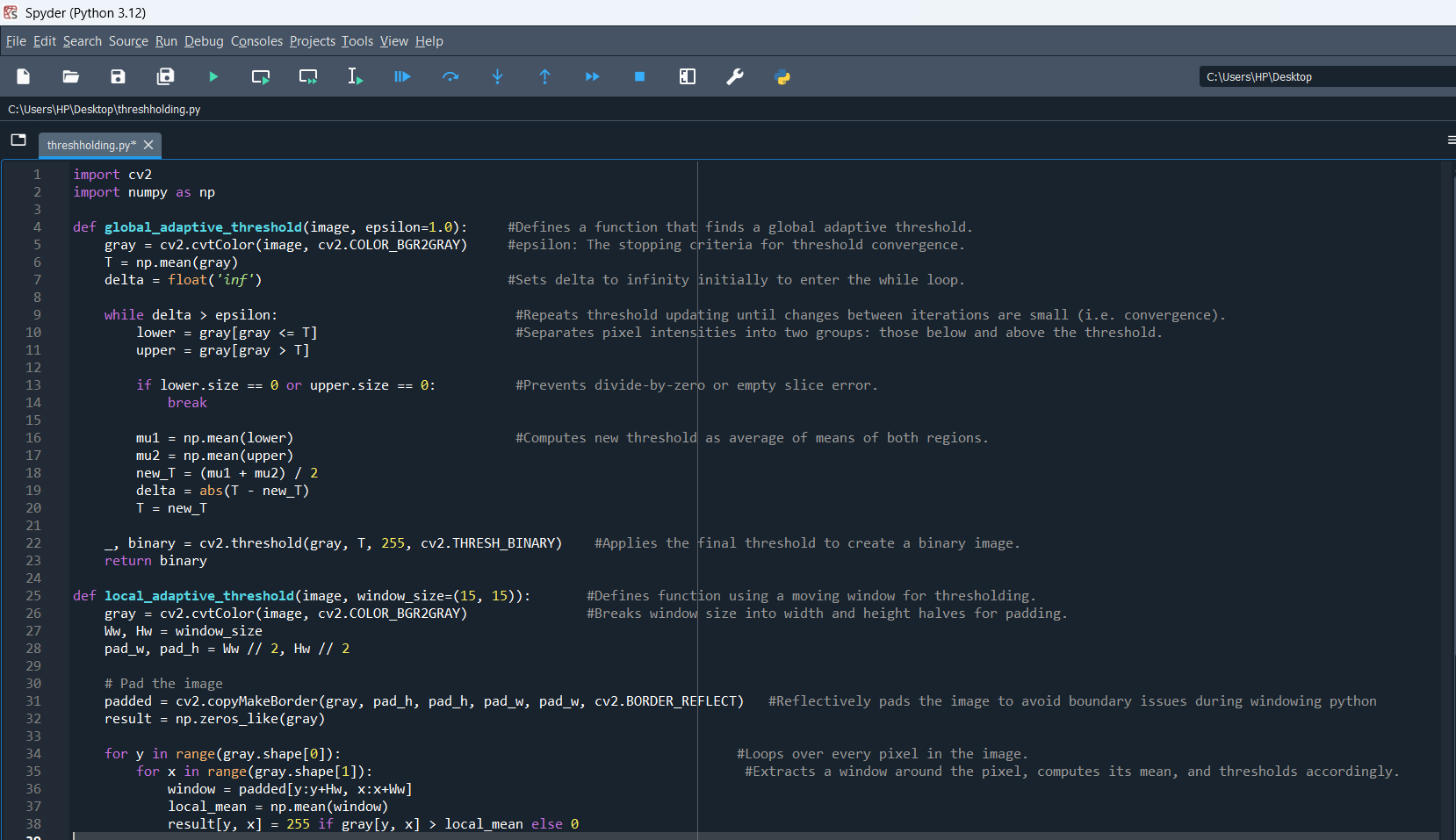
2.**Local Adaptive Thresholding**

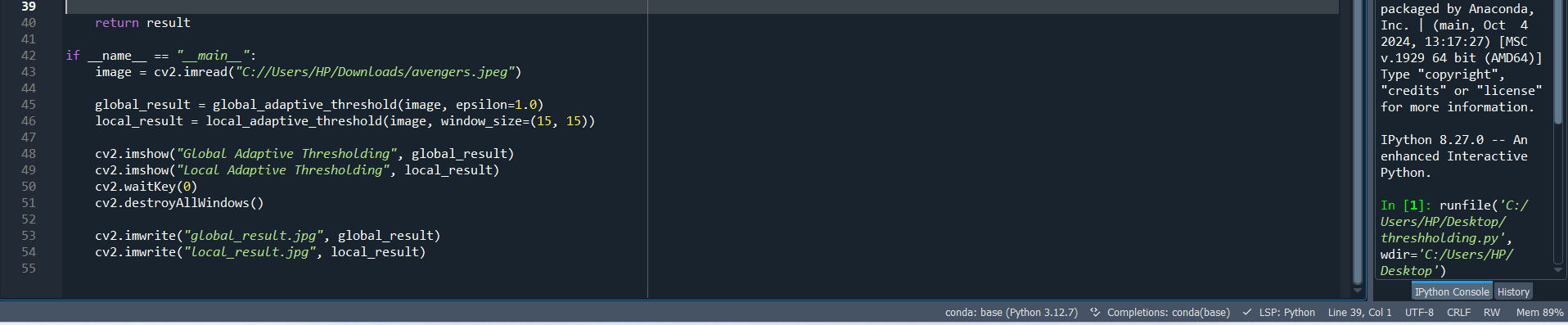
* Window size Ww×HwW\_w \times H\_wWw​×Hw​
* Each pixel compared to the local mean
* Binarization is locally adaptive to lighting variations

# Implementation:

* Platform: Python
* IDE: Spyder
* Input: Original image
* Output: Two binarized images
* Adjustable parameters: ϵ\epsilonϵ, window size Ww×HwW\_w \times H\_wWw​×Hw​

# Code:





# Code file:



# Actual Image Used:



# Global Adaptive Threshold:



# Local Adaptive Threshold:



# Conclusion:

* Both techniques are useful.
* Local thresholding is more robust in real-world cases.

# Reference Used:

Youtube : <https://www.youtube.com/watch?v=l1dhyw-EjSw>

AI Platform: Copilot

Lecture notes