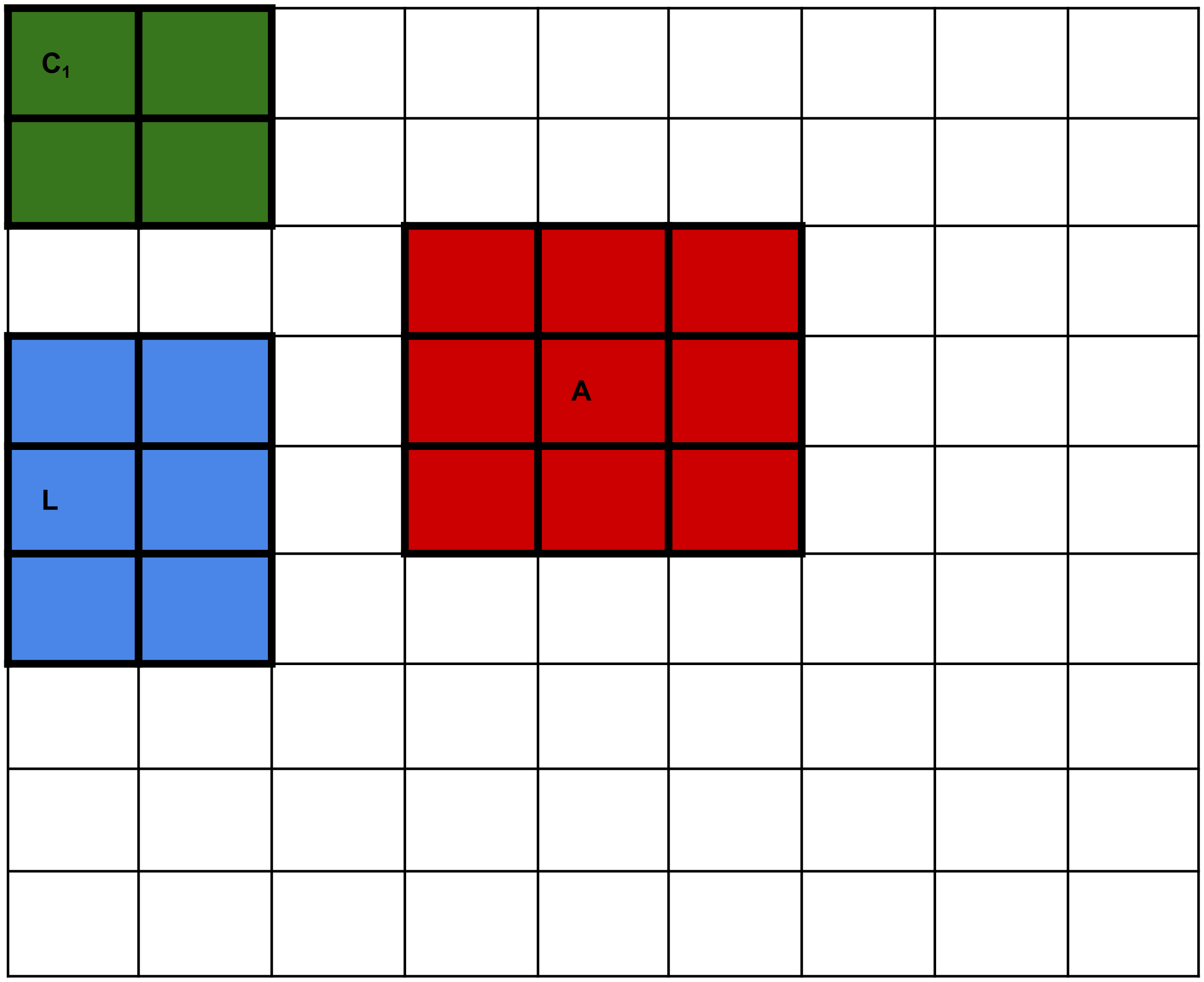
Problem Statement: Find Object in Random Matrix

Task: Given two 2D Random matrices, **mat1** and **mat2**. An object moves horizontally to the left in **mat2** from its initial position in **mat1.** Following constraints hold on the movement of object

Constraint 1: Any point at index **p** in row **r** of matrix **mat1** will always match to a point which is also in row **r** of matrix **mat2**. since the object moves horizontally to the left in **mat2.**

Constraint 2: A point in **mat1** can match to no more than one point in **mat2.** To ensure this constraint holds, use a 3x3 window around a point (see **Red** window below aroun point **A**) and use the mean square error as distance metric between two windows in **mat1** and **mat2**. For corner points modify this window to size 2x2 (see **green** windows around point **C_1** in below example), whereas for points that exist on the extreme left or extreme right of the matrix use a 2x3 window (See **Blue** windows around point **L** below).

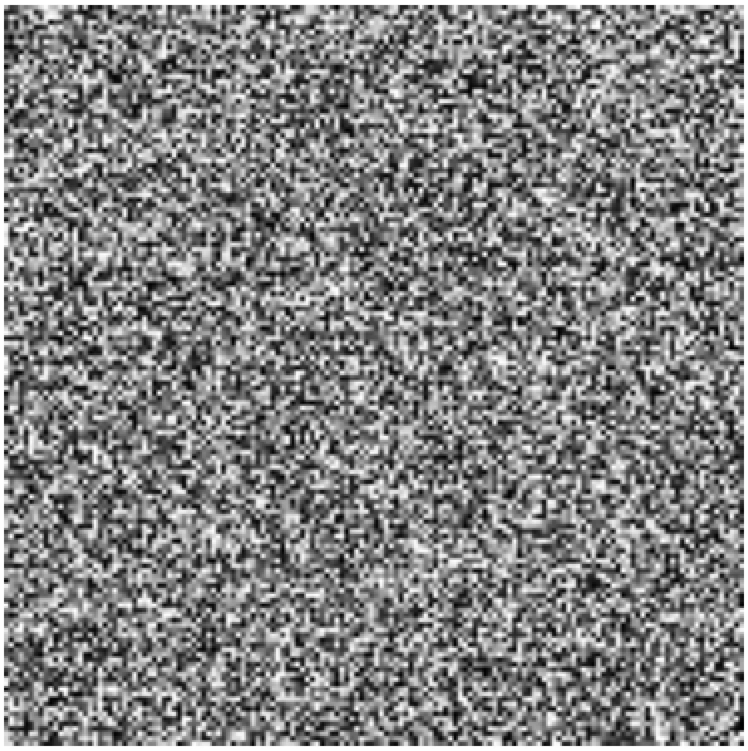


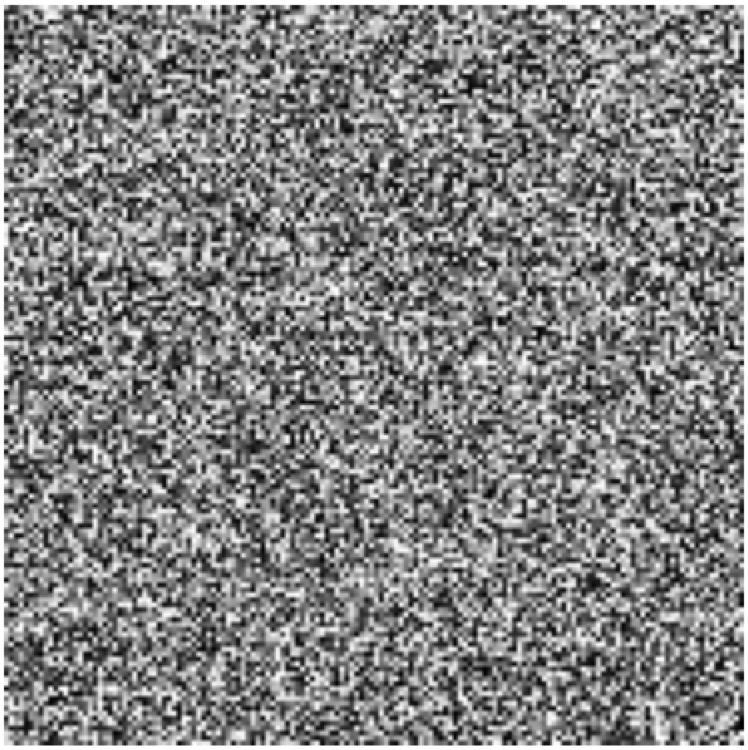
Constraint 3: If a point in **mat1** at position **p** matches to point in **mat2** at position **q,** then subsequent point in **mat1** at position **p+1** will find its match, if the match exists, in **mat2** at position **q+j** such that **j>0**

Write a dynamic program to find the mask of the Object.

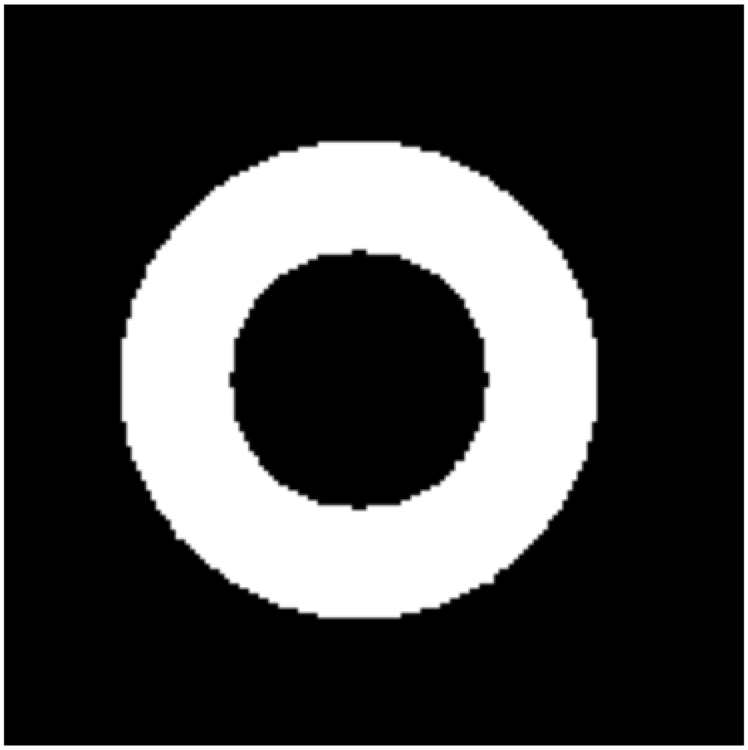
Examples:

1. Ring: Just a ring moves over a random 2D matrix. See ringGIF for more clarity and ring.py for code to generate random matrix

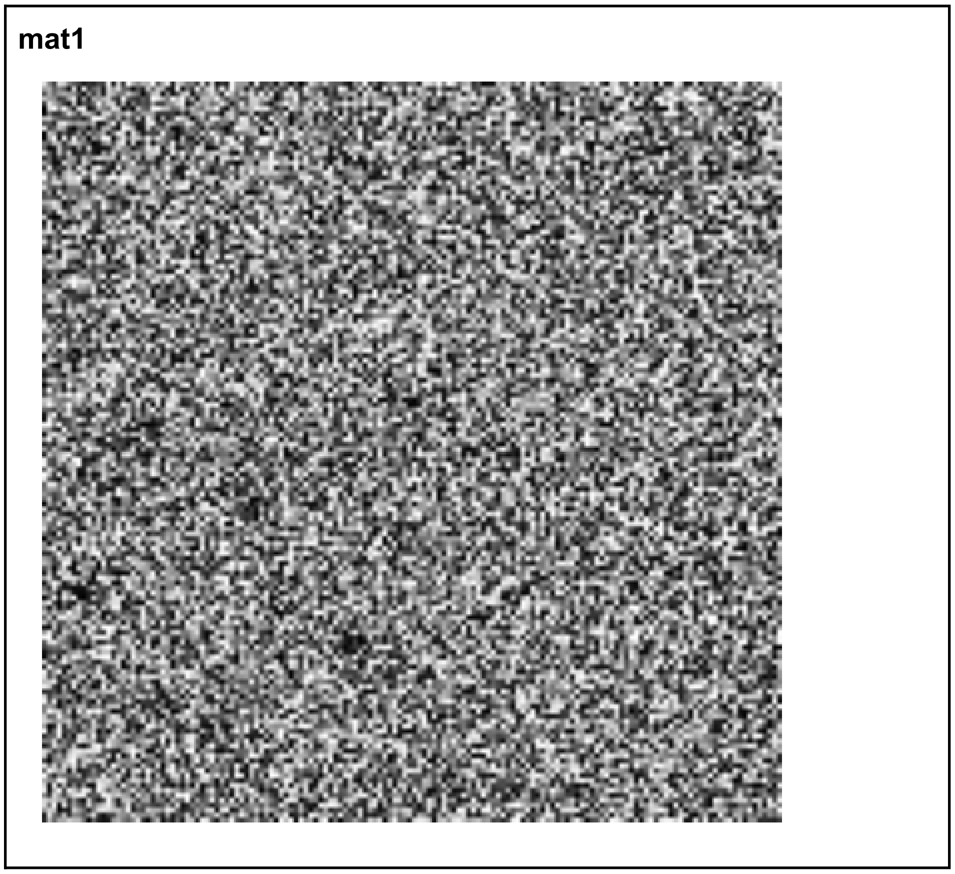
mat1 
mat2

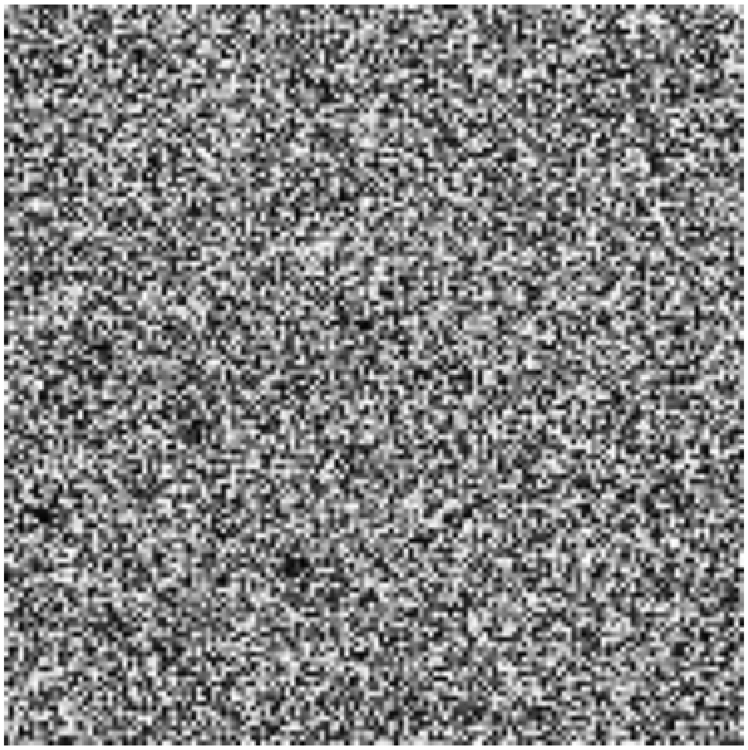


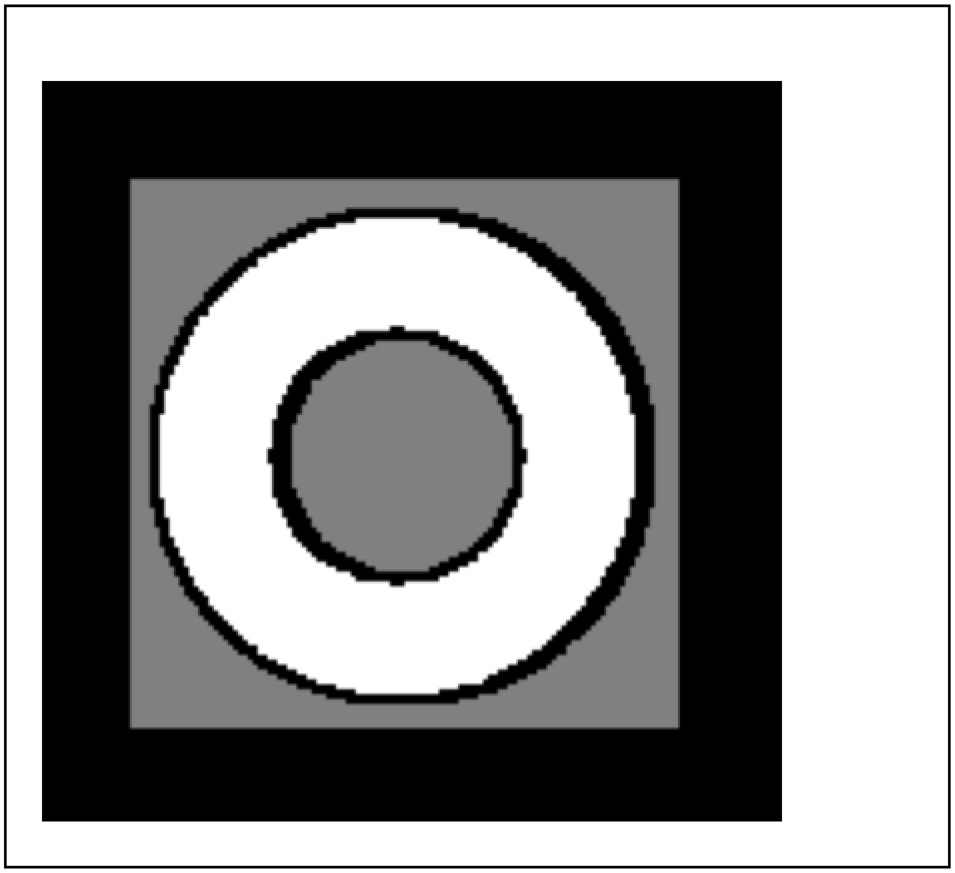
Result



2. RingoverSquare: Ring is placed over a square, Ring movies faster than the square. Note that square only moves by 2 points whereas Ring moves by 4 points, that’s why square is Gray and Ring is white.



mat2

Result



Constraints:

1. Program must be written,run in python over Google colab. After finishing the code, download python notebook as .ipynb (File>Download>.ipynb) and save it as <Rollnumber>.ipynb. (example 20EC91J02.ipynb). Also share it with editor rights with following email IDs from the share option in top right with following people
    - **Tamal Hansda:** tamalhansda16@gmail.com
    - **Aditya Purohit:** adityapurohit234@gmail.com

   Email to the aforementioned TAs and include a share link to your colab notebook. This must be done from the very first day you create the file in colab.

2. Provide a hand written formulation of the optimal solution used. Make your hypothesis clear and state the existence of a solution as is done in Dynamic Programming.
3. Use Numpy and Matplotlib to load the image or save the result, create a numpy array and display an image. A sample code has been provided to load, create a numpy array and plot an Image. **Apart from the functions listed in sample code, You are not allowed to use any built-in function from these or any other libraries.**
4. Do not Hardcode. Your code should automatically accommodate any size of matrices and should work for all possible shapes and configuration of shape that holds the constraint.

5. Solution should be Optimal, Brute force solution will attract penalties