PRACTICAL – 4

PROGRAM -1

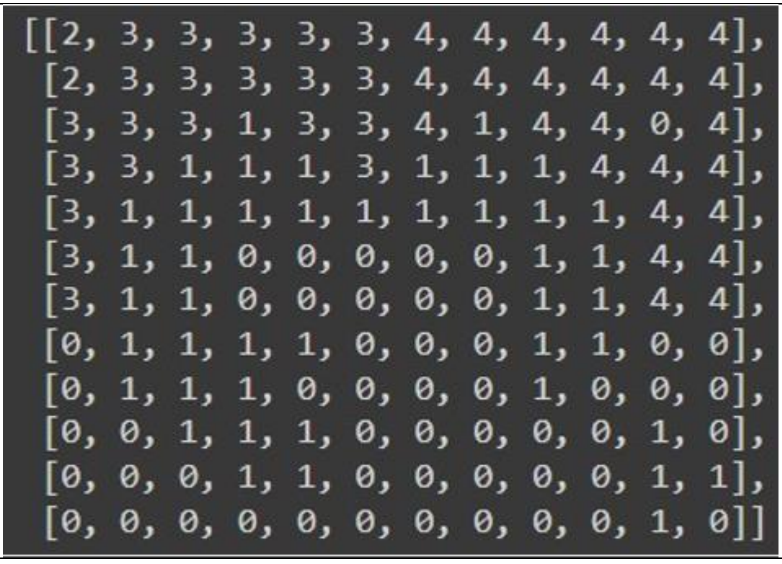
AIM- WAPP TO PERFORM FLOOD FILL ALGORITHM.

CODE

```
import random

m =[[2,3,3,3,3,3,4,4,4,4,4,4],
	[2,3,3,3,3,3,4,4,4,4,4,4],
	[3,3,3,1,3,3,4,1,4,4,0,4],
	[3,3,1,1,1,3,1,1,1,4,4,4],
	[3,1,1,1,1,1,1,1,1,1,4,4],
	[3,1,1,0,0,0,0,0,1,1,4,4],
	[3,1,1,0,0,0,0,0,1,1,4,4],
	[2,1,1,1,1,0,0,0,1,1,2,2],
	[2,1,1,1,0,0,0,0,1,2,2,2],
	[2,2,1,1,1,0,0,2,2,2,1,0],
	[2,2,2,1,1,0,0,2,2,2,1,1],
	[2,2,2,2,2,2,2,2,2,0,1,0]
	]


def flood_recursive(matrix):
	width = len(matrix)
	height = len(matrix[0])
	def ff(x,y,start_color,color_to_update):
		#if the square is not the same color as the starting point
		if matrix[x][y] != start_color:
			return
		#if the square is not the new color
		elif matrix[x][y] == color_to_update:
			return
		else:
			#update the color of the current square to the replacement color
			matrix[x][y] = color_to_update
			neighbors = [(x-1,y),(x+1,y),(x-1,y-1),(x+1,y+1),(x-1,y+1),(x+1,y-
1),(x,y-1),(x,y+1)]
			for n in neighbors:
				if 0 <= n[0] <= width-1 and 0 <= n[1] <= height-1:
					ff(n[0],n[1],start_color,color_to_update)
	start_x = random.randint(0,width-1)
	start_y = random.randint(0,height-1)
	start_color = matrix[start_x][start_y]
	ff(start_x,start_y,start_color,0)
```

```
        return matrix

flood_recursive(m)
```

OUTPUT



```
[[2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4],
 [2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4],
 [3, 3, 3, 1, 3, 3, 4, 1, 4, 4, 0, 4],
 [3, 3, 1, 1, 1, 3, 1, 1, 1, 4, 4, 4],
 [3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 4, 4],
 [3, 1, 1, 0, 0, 0, 0, 0, 1, 1, 4, 4],
 [3, 1, 1, 0, 0, 0, 0, 0, 1, 1, 4, 4],
 [0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0],
 [0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0],
 [0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0],
 [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]]
```