**SOLUTION**

class Solution {

public:

Solution(){

ios::sync\_with\_stdio(false);

std::cin.tie(nullptr);

std::cout.tie(nullptr);

}

vector<vector<int>> floodFill(vector<vector<int>>& image, int r, int c, int newColor) {

if(image[r][c]== newColor)

return image;

dfs(image,r,c,newColor,image[r][c]);

return image;

}

void dfs(vector<vector<int>>& image, int r, int c, int newColor, int oldColor){

if(r<0 || r>=image.size() || c<0 || c>=image[0].size() || oldColor!=image[r][c])

return;

image[r][c]=newColor;

dfs(image,r+1,c,newColor,oldColor);

dfs(image,r-1,c,newColor,oldColor);

dfs(image,r,c+1,newColor,oldColor);

dfs(image,r,c-1,newColor,oldColor);

}

};

**TIME COMPLEXITY: O(N)**

**SPACE COMPLEXITY: O(N)**