UNIVERSITÄT
DES
SAARLANDES

Programmierung 2 - SoSe 23

Projekt 2 - C-Convert

Authors: Jerry Takou, Lucas Meyer, Felix Caglioti

10. Mai 2023

Saarland University

Overview

1. Organisation
2. Motivation
3. Image Format
4. Algorithm
5. Code Example

Organisation

Git Project Repository

You can get the project files with `git clone` using the following command:

```
git clone ssh://git@git.prog2.de:2222/project2/$NAME.git $FOLDER
```

$NAME = your username in the CMS / our course website

$FOLDER = folder in which you want to copy the project

Compiling & Testing

To compile your code you can run

```
make
```

To run our tests over your implementation you can run either `make check` or `./run_tests.py`

To run certain parts of your implementation you can use the appropriate command line arguments, e.g.

```
./bin/convert read-and-write > output.ppm < ./test/data/owl.ppm
```

Note: check the main function or the project description for all options

Motivation

Motivation

Broadway Tower is constructed primarily for decoration (folly) on Broadway Hill, near the large village of Broadway, in the English county of Worcestershire, at the second-highest point of the Cotswolds (after Cleeve Hill). Broadway Tower's base is 312 metres above sea level. The tower itself stands 20 metres high.

(Source: https://en.wikipedia.org/wiki/Broadway_Tower,_Worcestershire)

Motivation

(a) Original image

(b) Clockwise rotation

Motivation

(a) Original image

(b) Counterclockwise rotation

Motivation

(a) Original image

(b) Image mirrored along the vertical axis

Motivation

(a) Original image

(b) Image mirrored along the horizontal axis

Motivation

(a) Original image

(b) Image resized to 100×100

Motivation

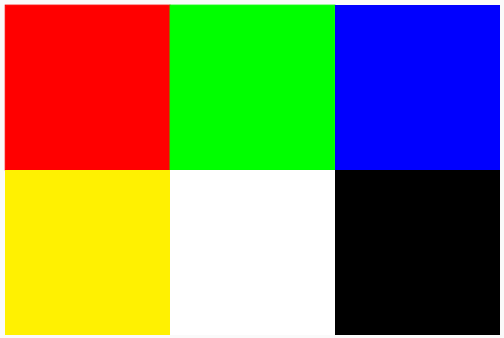
(a) Image of 100×100.

(b) Image resized to 500×500

Image Format

Image Format

```
P3
3 2
255
255   0   0   0 255   0   0   0 255
255 255   0 255 255 255   0   0   0
```



Algorithm

Algorithm overview

- Clockwise and counterclockwise rotation
- Mirroring images
- Resizing images
- Flood algorithm

Algorithm overview

- Clockwise and counterclockwise rotation
- Mirroring images
- Resizing images
- Flood algorithm

Clockwise and counterclockwise rotation

Representation of the matrix:

- case 1: The original image is of size n×n, here 5×5

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

rotation →

21	16	11	6	1
22	17	12	7	2
23	18	13	8	3
24	19	14	9	4
25	20	15	10	5

- The rotated image is of size n×n, here 5×5

Clockwise and counterclockwise rotation

Representation of the matrix:

- case 2: The original image is of size m×n, here 5×2

1	2
3	4
5	6
7	8
9	10

rotation →

9	7	5	3	1
10	8	6	4	2

- The rotated image is of size n×m, here 2×5

Clockwise and counterclockwise rotation

Clockwise rotation:

- considering (X,Y), where X = column position, Y= row position. Note: $0 \leq$ X,Y$<$n for an n$\times$n matrix
- indices (0,0), (1,0), (2,0) become (n-1,0), (n-1,1), (n-1,2)
- indices (0,1), (1,1), (2,1) become (n-2,0), (n-2,1), (n-2,2)

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

clockwise rotation

21	16	11	6	1
22	17	12	7	2
23	18	13	8	3
24	19	14	9	4
25	20	15	10	5

Clockwise and counterclockwise rotation

Clockwise Rotation:

- considering (X,Y), where X = column position, Y= row position. Note: $0 \leq$ X,Y, X$<$n and Y$<$m for an m$\times$n matrix
- indices (0,0), (1,0) become (m-1,0), (m-1,1) in the n$\times$m matrix
- indices (0,1), (1,1) become (m-2,0), (m-2,1) in the n$\times$m matrix

1	2
3	4
5	6
7	8
9	10

clockwise rotation $\longrightarrow$

9	7	5	3	1
10	8	6	4	2

Clockwise and counterclockwise rotation

Stop! Go back and have a look at how the indices change as they rotate. When you are done, let's move to the counterclockwise rotation.

Clockwise and counterclockwise rotation

Counterclockwise rotation:

- considering (X,Y), where X = column position, Y= row position. Note: $0 \leq X,Y < n$ for an $n \times n$ matrix
- indices (0,0), (1,0), (2,0) become (0,n-1), (0,n-2), (0,n-3)
- indices (0,1), (1,1), (2,1) become (1,n-1), (1,n-2), (1,n-3)

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

counterclockwise rotation →

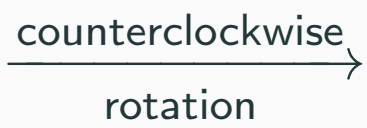
5	10	15	20	25
4	9	14	19	24
3	8	13	18	23
2	7	12	17	22
1	6	11	16	21

Clockwise and counterclockwise rotation

Counterclockwise rotation:

- considering (X,Y), where X = column position, Y= row position. Note: $0 \leq$ X,Y, X$<$n and Y$<$m for an m$\times$n matrix
- indices (0,0), (1,0) become (0,n-1), (0,n-2) in the n$\times$m matrix
- indices (0,1), (1,1) become (1,n-1), (1,n-2) in the n$\times$m matrix

1	2
3	4
5	6
7	8
9	10

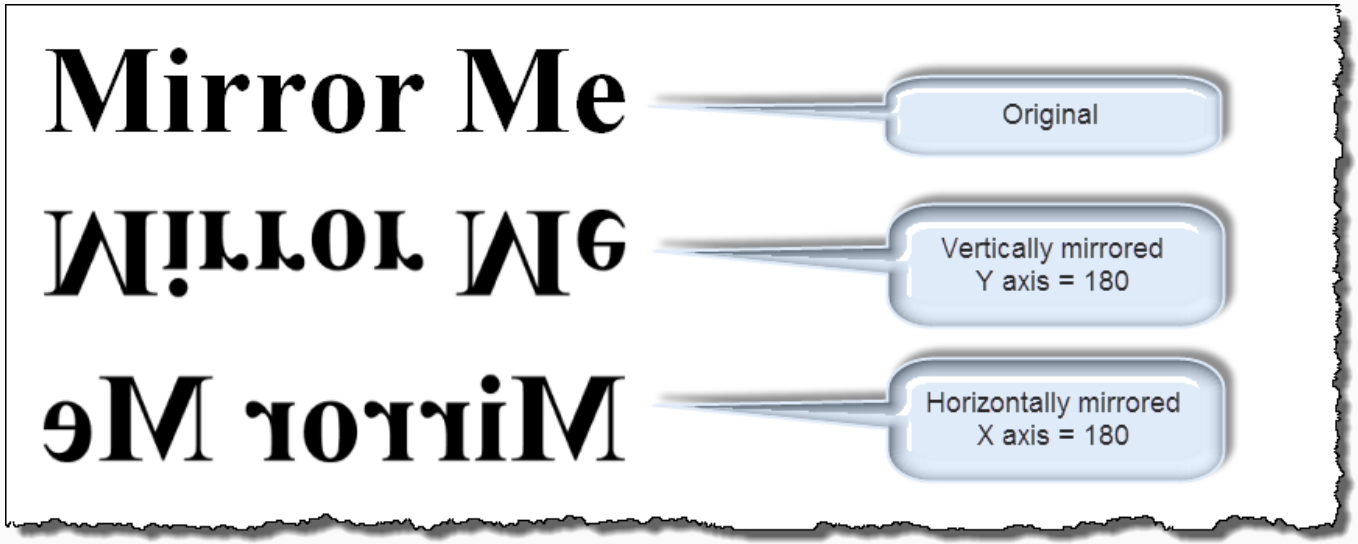


2	4	6	8	10
1	3	5	7	9

Algorithm overview

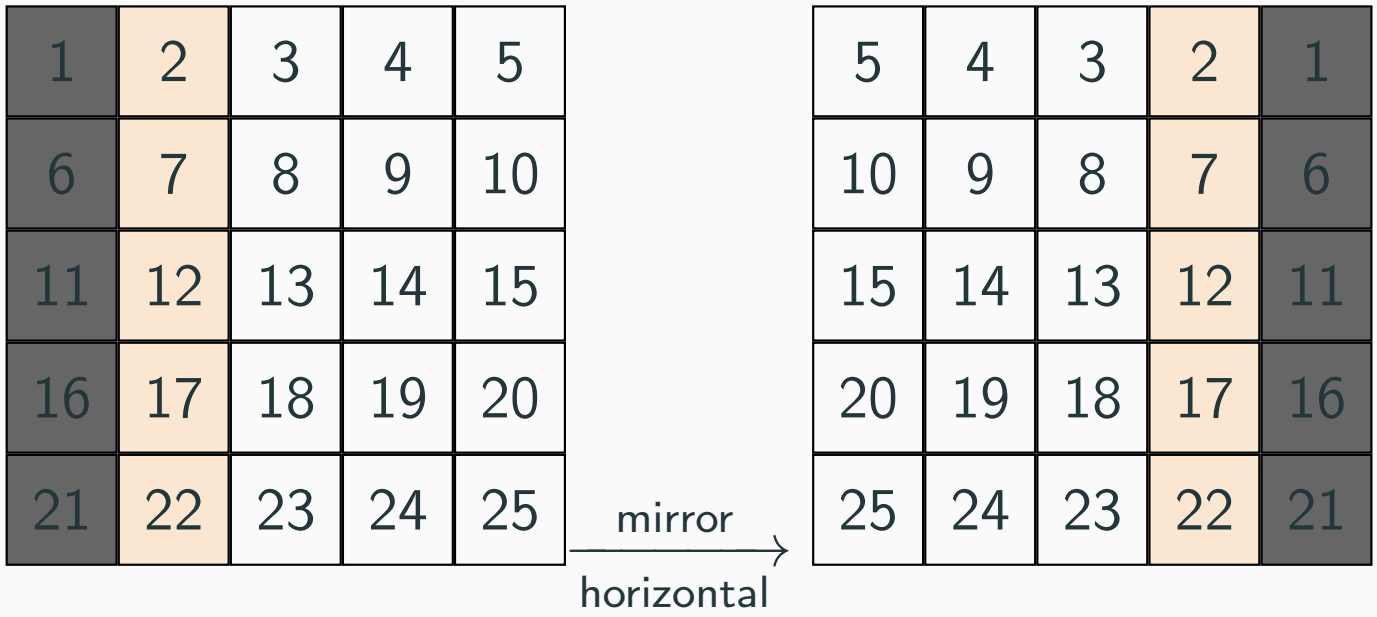
- Clockwise and counterclockwise rotation
- Mirroring images
- Resizing images
- Flood algorithm

Mirroring images



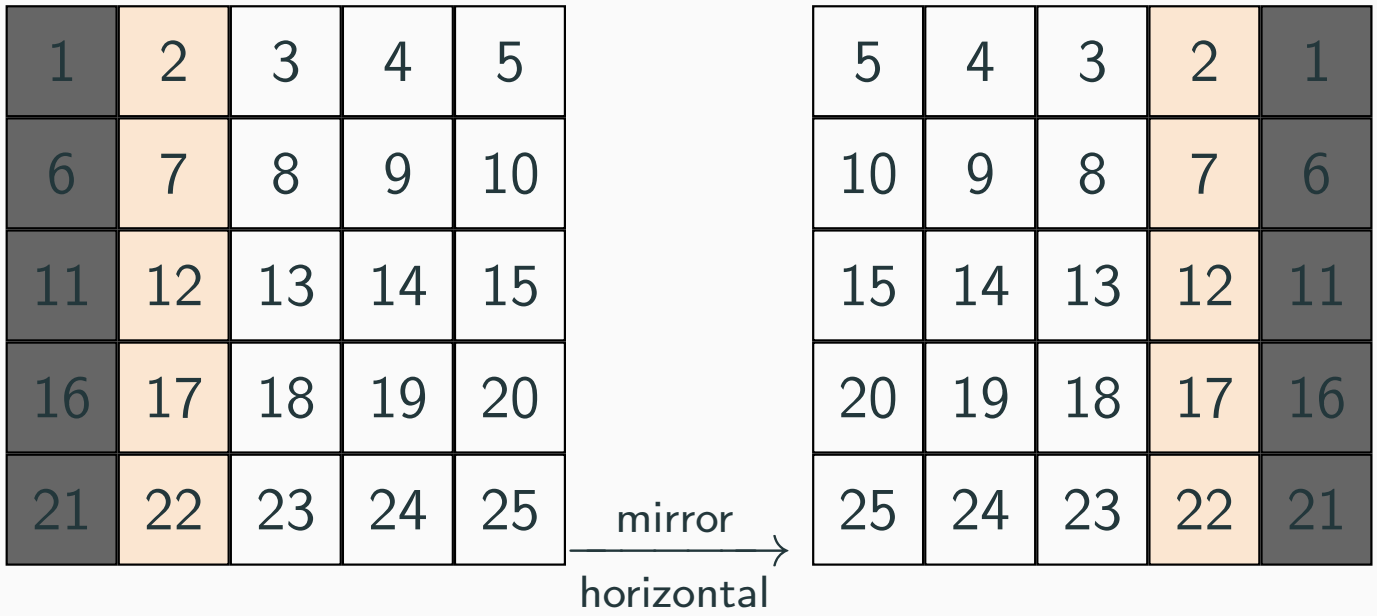
Mirroring images: horizontal

- The height and width of the matrix remain the same
- horizontal mirroring



Mirroring images: horizontal

- considering (X,Y), where X = column position, Y= row position. Note: $0 \leq$ X,Y, X$<$n and Y$<$m for an m$\times$n matrix
- The order of elements within each row is reversed
- (0,0), (1,0), (2,0) become (n-1,0),(n-2,0),(n-3,0)



Mirroring images: vertical

- The height and width of the matrix remain the same
- vertical mirroring

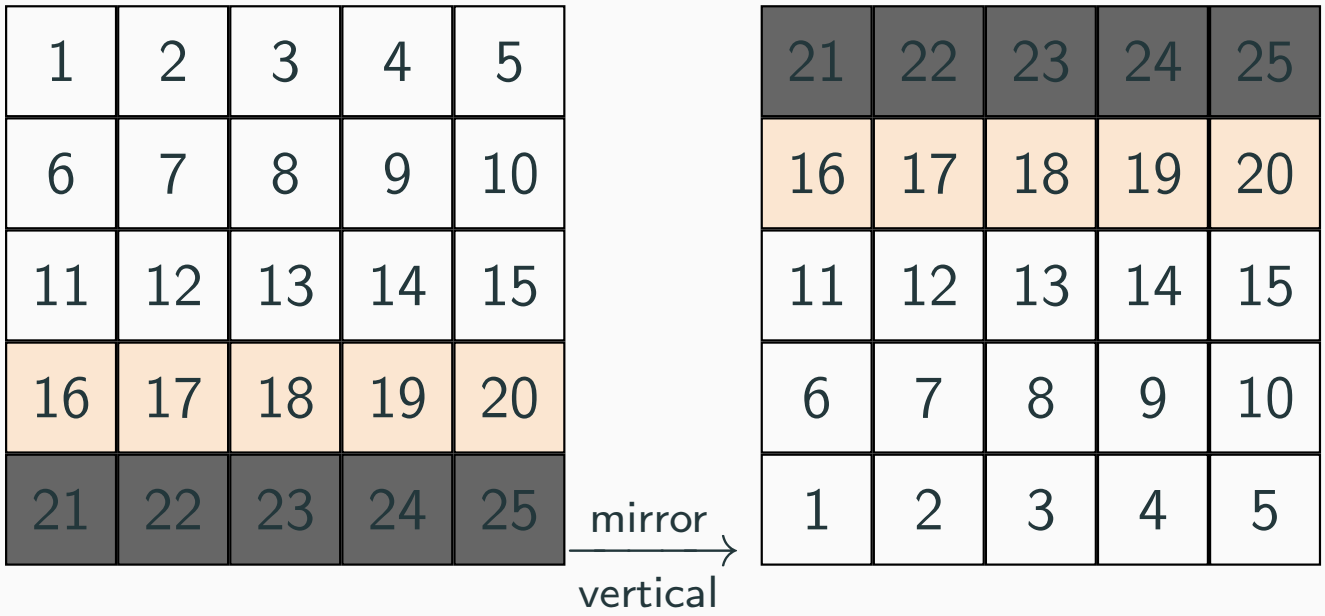
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

mirror vertical →

21	22	23	24	25
16	17	18	19	20
11	12	13	14	15
6	7	8	9	10
1	2	3	4	5

Mirroring images: vertical

- considering (X,Y), where X = column position, Y= row position. Note: $0 \leq$ X,Y, X$<$n and Y$<$m for an m$\times$n matrix
- The order of elements within each column is reversed
- (0,0), (1,0), (2,0) become (0,n-1), (1,n-1), (2,n-1)



Algorithm overview

- Clockwise and counterclockwise rotation
- Mirroring images
- Resizing images
- Flood algorithm

Resizing images

Resizing:

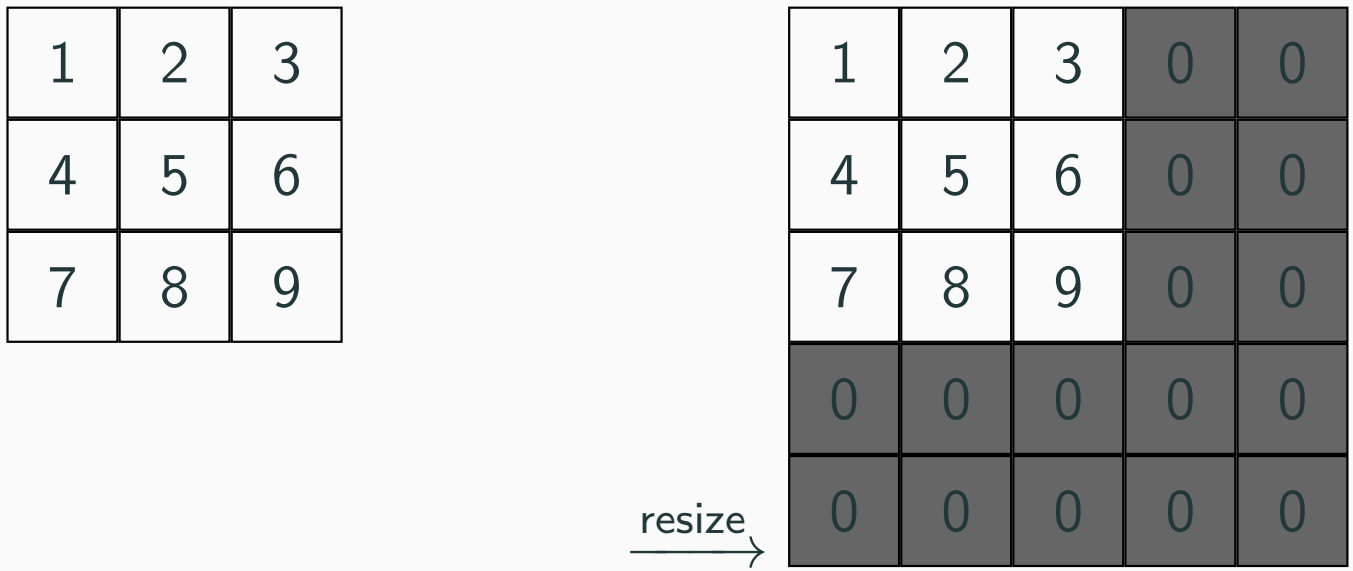
- case 1: The original image is of size m$\times$n, here (3$\times$3) and is to be resized to an image of size m'$\times$n', with $m' > m$ and $n' > n$ (5$\times$5)

1	2	3
4	5	6
7	8	9

Resizing images

Note:

- indices (0,0), (1,0) stay at (0,0), (1,0) in the new n×m matrix
- indices (3,y), (4,y), (x,3) and (x,4) were newly added



Resizing images

- case 2: The original image is of size m×n, here (5×5) and is to be resized to an image of size m'×n', with $m' < m$ and $n' < n$ (3×4)

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Resizing images

Note:

- indices (0,0), (1,0) stay at (0,0), (1,0) in the new n×m matrix
- indices (3,y), (4,y) and (x,4) were removed

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

resize →

1	2	3
6	7	8
11	12	13
16	17	18

Resizing images

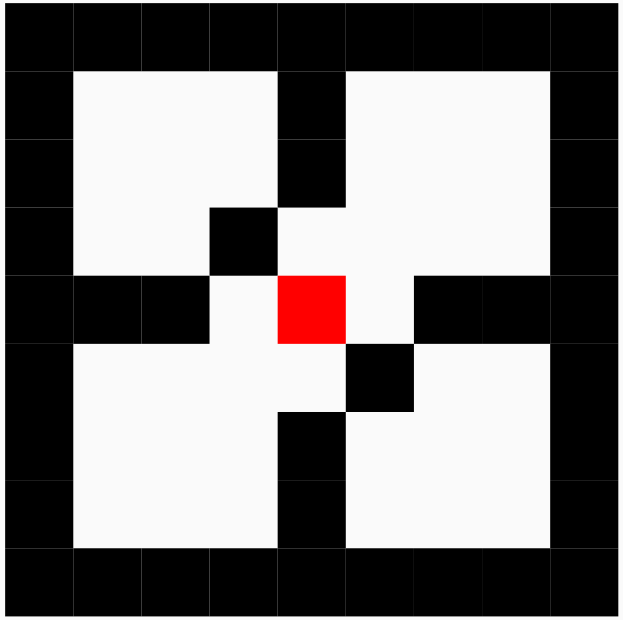
Warning!

These two cases are not the only ones!!!

Algorithm overview

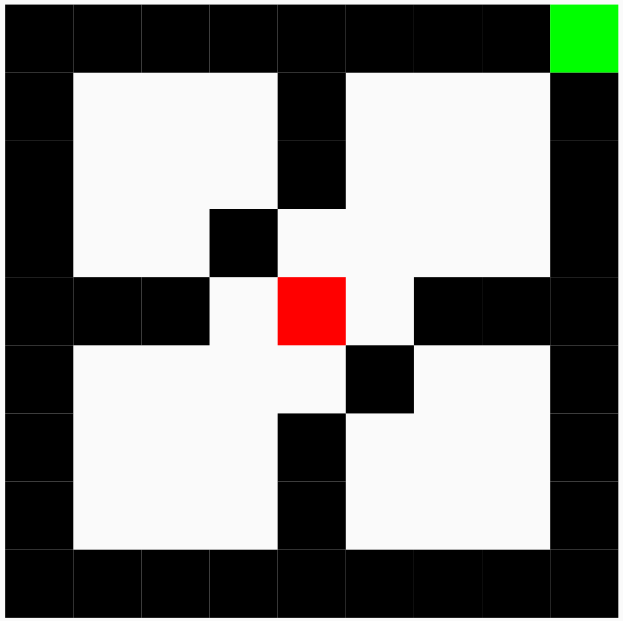
- Clockwise and counterclockwise rotation
- Mirroring images
- Resizing images
- Flood algorithm

Flood algorithm



Flood algorithm

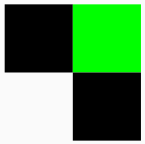
Begin the flood algorithm at the given position (x, y), here $(8, 0)$:



Flood algorithm

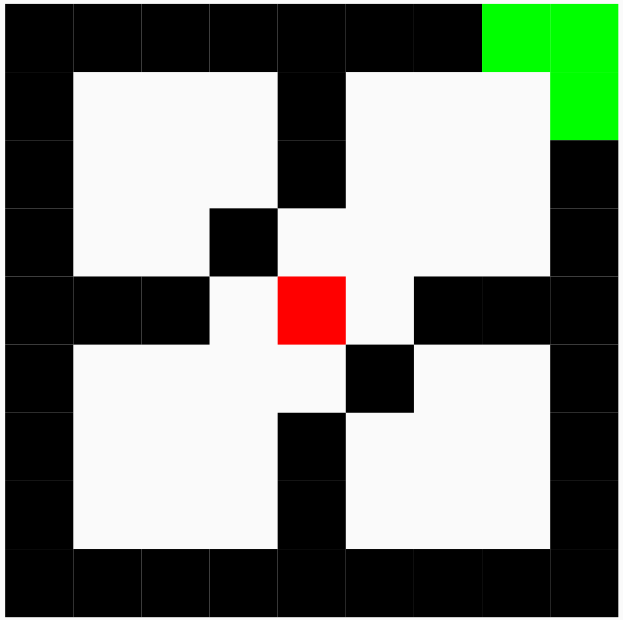
Check if neighbouring pixels, excluding the diagonal ones, have the same colour as the starting pixel.

$\rightarrow$ *Note:* The pixels must be inside the image!

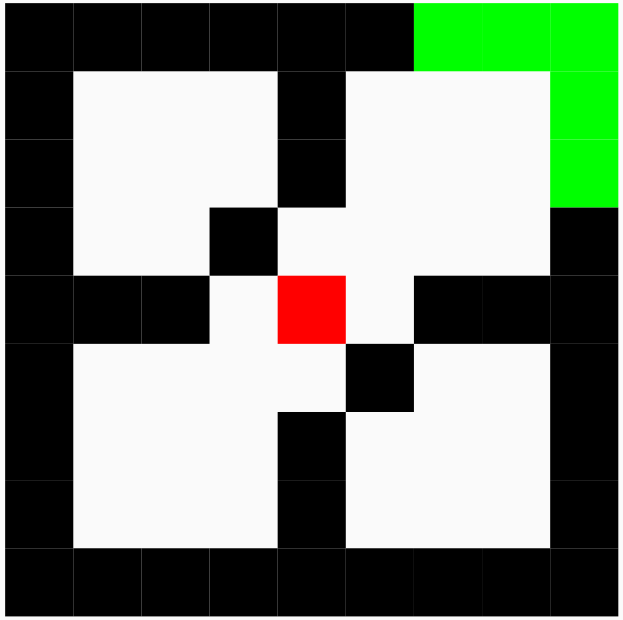


Flood algorithm

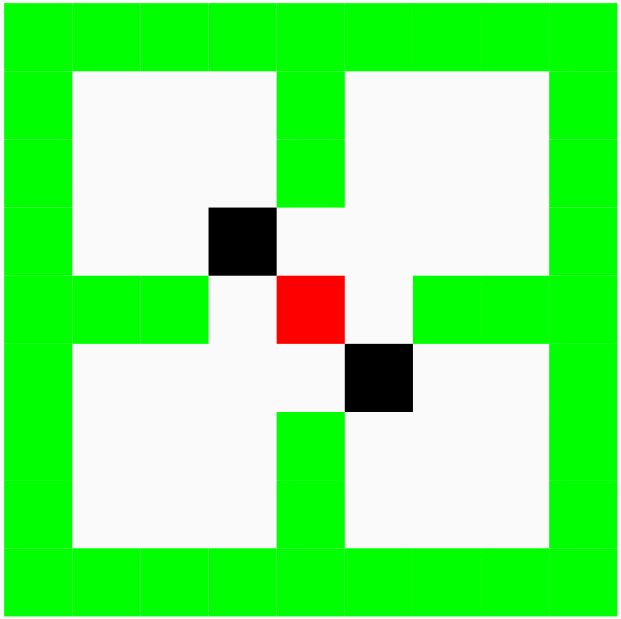
Continue recursively until no neighbouring pixels match:



Flood algorithm

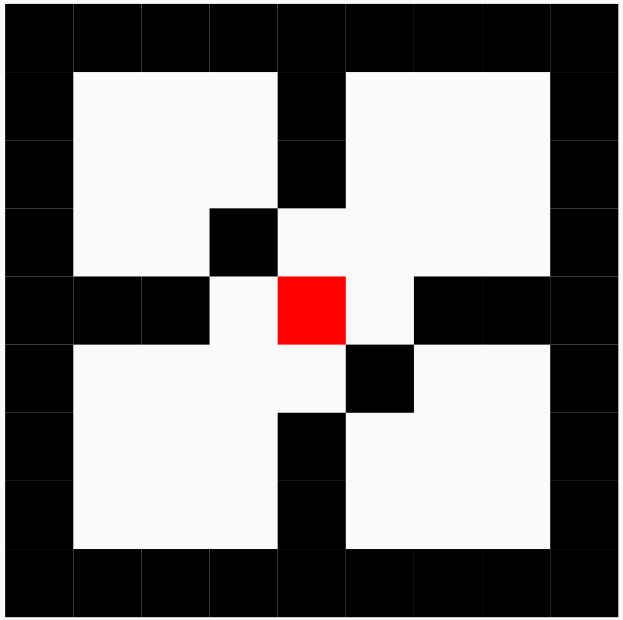


Flood algorithm



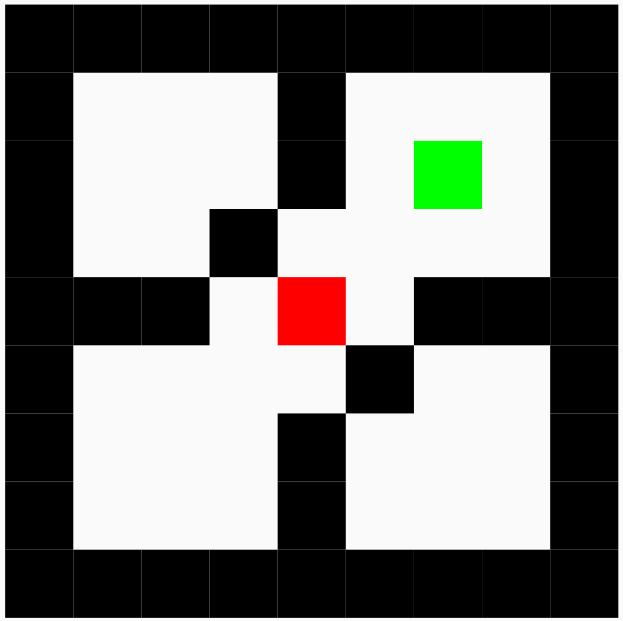
Flood algorithm

Another example:



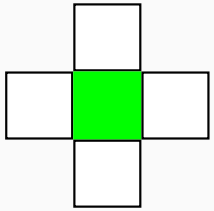
Flood algorithm

Begin the flood algorithm at the given position (x, y), here $(6, 2)$:

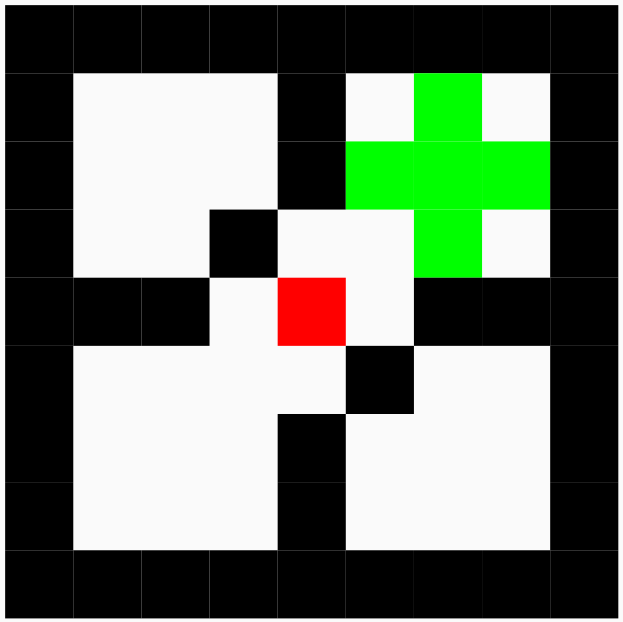


Flood algorithm

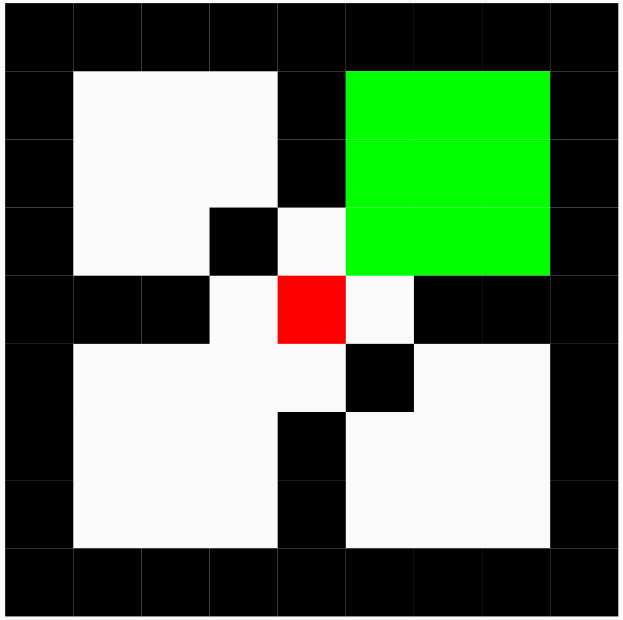
Here we have to check all four neighbours since none are outside the image.



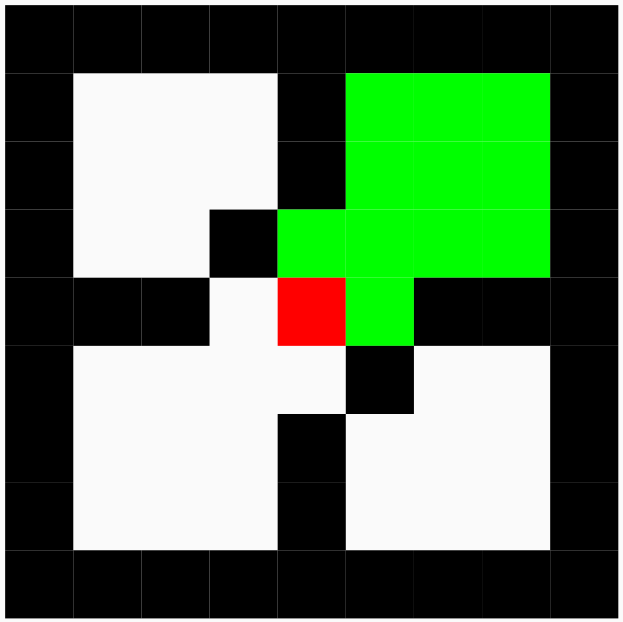
Flood algorithm



Flood algorithm



Flood algorithm



Questions?

Code Example