# MP 1 Image Manipulation

Due: **Monday, September 11 at 11:59 PM**

⚠ **Solo MP**

This MP, as well as all other MPs in CS 225, are to be completed without a partner.

You are welcome to get help on the MP from course staff, via open lab hours, or Piazza!

## Ready, Set

Before starting this MP, make sure you have finished `lab_intro`.

- Copy your `HSLAPixel.cpp` and `HSLAPixel.h` files from `lab_intro` into `mp1`.
- Just like in `lab_intro`, these files both go into the `cs225` directory within the assignment folder.

## Lets Go!

This MP is the only one-week MP in CS 225 and is designed to get you set up for future MPs.

### Part 1: Getting the files

As with all assignemnts in CS 225, you can download the files by running the following command in your `cs225` svn directory:

```
svn up
```

If something goes wrong, our SVN Reference has common problems and fixes for you!

ⓘ **No SVN**

If you do not have an account on the svn server, you can still work on this MP by downloading the source files from mp1.zip.

### Part 2: Create a Makefile

In CS 225, we feel it’ s important you understand how a C++ program compiles.

Go through the Makefile Tutorial and create a `Makefile` for this assignemnt. You may find the `lab_intro` Makefile useful as a reference.

Your `Makefile` must compile together your own solution files, namely `main.cpp`, `mp1.cpp`, `mp1.h`, and files in the `cs225` directory. Do not have typos in your file names or your `Makefile`! For example, make sure your Makefile compiles a file named `main.cpp`, not a file named `main.C` or `test.cpp` or any other such thing.

Please make sure your `Makefile` does not compile extra files that are not part of this MP. For example, do not add in files from the `Makefile` tutorial by mistake; the only files the `Makefile` should be dealing with are the few listed in the paragraph above.

Your `Makefile` must produce an executable called `mp1` (all lowercase).

**Hint: Makefile for testing**

After creating a `Makefile` that builds `mp1`, the following two lines can be added to the end of your `Makefile` so that you can also build the test cases:

```
test : unit_tests.o mp1.o PNG.o HSLAPixel.o lodepng.o
	$(LD) unit_tests.o mp1.o PNG.o HSLAPixel.o lodepng.o $(LDFLAGS) -o test

unit_tests.o : tests/unit_tests.cpp tests/catch.hpp cs225/PNG.h cs225/HSLAPixel.h
	$(CXX) $(CXXFLAGS) tests/unit_tests.cpp
```

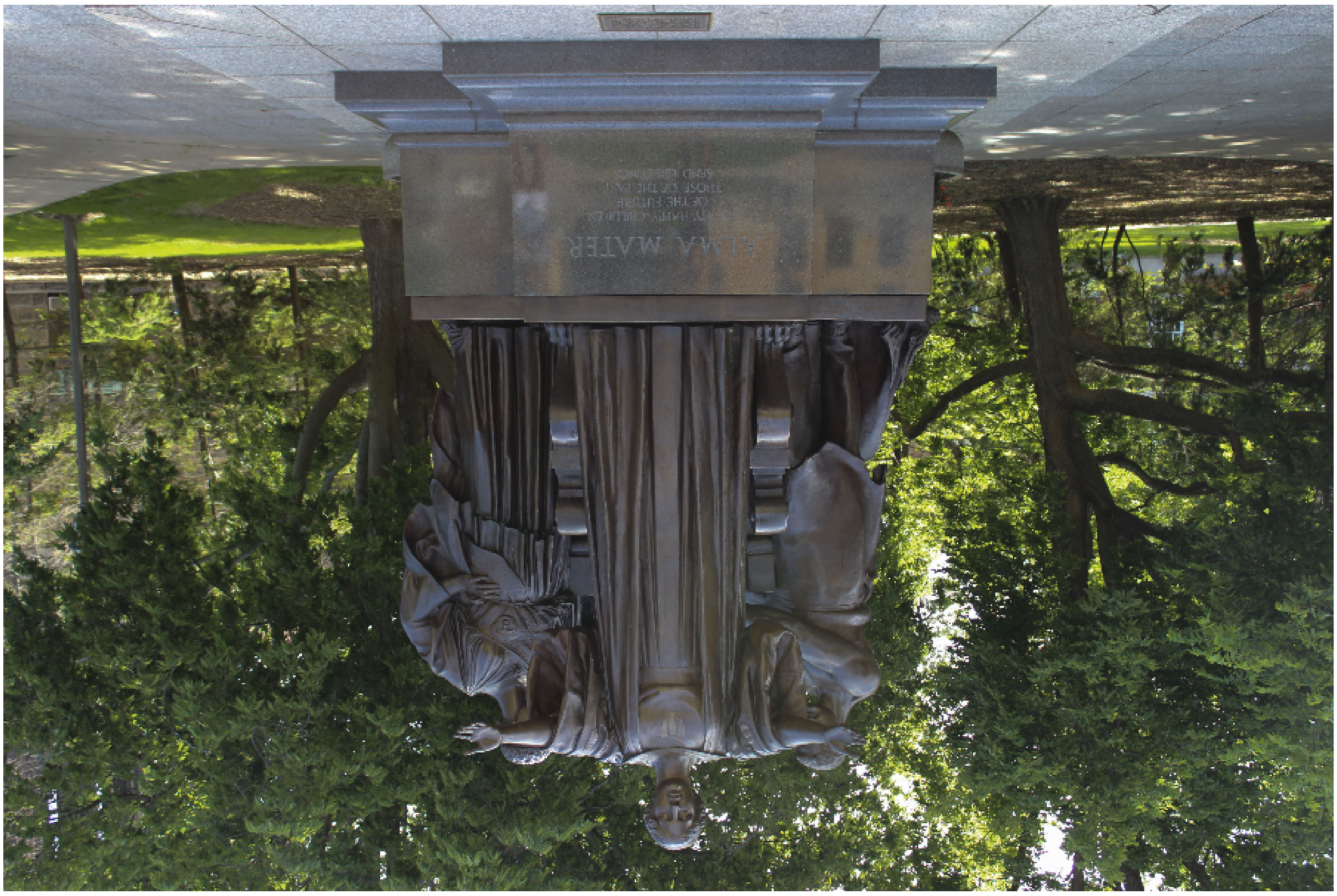
## Part 3: Rotate an Image

Open `mp1.cpp` and complete the `rotate` function. This function must:

- Read in `inputFile`,
- Rotate the image 180 degrees,
- Write the rorated image out as `outputFile`

Here’ s `alma.png` rotated 180 degrees:

in. png


out.png

In order to complete this, you will need to make use of the CS 225’ s PNG class (make sure to add `#includes` as needed). The CS 225 PNG class is documented here.

**Hint: How Do I Rotate an Image?**

Take a piece of paper and draw a 5x5 grid on it. Mark the box at (0, 1). Rotate that paper 180 degrees and note where the marked box is now located. Repeat for other squares if necessary.

Can you find the pattern/formula for how the pixel moves?

## Testing

The `Makefile` you created must produce an `mp1` executable (all lower case `mp1`). The following command must make `mp1`:

```
make
```

A `main` has been provided for you that will call your `rotate` to read in `in.png` and output `out.png`. With that, you may use the given files `in_01.png`, `out_01.png`, `in_02.png`, `out_02.png`, `in_03.png`, and `out_03.png` to test your program. First, copy `in_01.png` to `in.png` by typing

```
cp in_01.png in.png
```

at the command line. Once your program has run, type

```
diff out.png out_01.png
```

to compare your program’ s output to `out_01.png`, which is the output file produced by the solution program written by the course staff.

You are encouraged to test your program on additional images of your own, looking at each output image to see whether it is a perfect 180 degree rotation of the corresponding input image.

### Autograder Testing

You can run a subset of the test cases that will be used in the autograder with the following commands:

```
make test
./test
```

## Grading and Submission

We have provided your output format in the MP specification above, and you need to match that **exactly**. Testing will be done by comparing your output PNG image file to our own (using the `diff` utility described in the previous section). If your image file does not match ours, that is considered incorrect output; you do not get partial credit for getting “close”.

When you’re ready to commit a solution, make sure to add all of the files you created:

```
svn add Makefile
svn add cs225/HSLAPixel.cpp
svn add cs225/HSLAPixel.h
```

To commit your changes to the repository type:

```
svn ci -m "mp1 submission"
```