Exercise: Python/C++

Write API (a function or a collection of classes - design decisions are up to you as long as the code is clean, readable, maintainable, and reasonably memory- and CPU-efficient) to count number of colored areas in an image. Then write an application that would use your API.

Input: a grey-scale image represented as a 2-dimensional array of unsigned bytes.

Output: array of 256 unsigned int numbers, each of them being a count of areas colored with the corresponding shade of grey.

For example, the image (download link: https://drive.google.com/file/d/1ocv4aGn-asPzP87i6cGohup2j7O872GR/view?usp=sharing) has 2 white areas, 3 black areas, and 2 grey areas (with value 200), i.e. the output array A will have all its elements set to 0, except A[0] that will be set to 3, A[255] will be 2, and A[200] will be 2.

The implementation should be in Python or C++ (C++ slightly more preferred). You can present two solutions: one in C++ and one in Python, but this is very optional.

You can use common libraries (e.g. numpy or STL) for input, output, and generic operations on containers for common operations, but you need to implement the algorithm itself on your own.

We should be able to run your code as following, with text output on stdout:

```
count-areas <input-filename> --shape <height>,<width>
```

Where input file is a binary representation of 2D unsigned char array and the output to stdout is 256 numbers corresponding to the area counts for pixel values 0, 1, ..., 255

For example, for the sample image "sample.bin" (Download Link: https://drive.google.com/file/d/115BQAu5XQk9059MTbF7bVkQUWhQq-2IM/view?usp=sharing):

```
> count-areas sample.bin --shape 256,256
3
0
...
2
...
0
2
```

If your solution is in C++, please send us compilation scripts or build instructions. The submissions should be able to be built on a linux system.