Name: _____

This assignment is a simple getting-started exercise and much easier than ones that follow. The focus is on figuring out various details such as packaging, submission, coding style, testing, timing, etc. that, while not specific to algorithms or this course, are nevertheless essential for completing other work. Please use the class discussion forum for clarifications and discussions.

The main task: blurring an image We wish to write a program that blurs an optical image (picture), reading input from *standard input* and writing output to *standard output*, along with optional diagnostics to *standard error*. While there are many interesting variations of this problem, we will use a very simple one because our focus is on the other aspects noted above. Briefly, this simple variation replaces each pixel of an image with the mean (rounded up to the nearest integer) of the pixels in a box centered on that pixel. In cases where some of the box falls outside the array (for pixels near the edges), the part of the box that contains no valid pixels is simply ignored.

In more detail, consider the input composed of a $w \times h$ (width times height) input array P of nonnegative integers (the pixels) and another nonnegative integer b (the blurring strength). Using the textbook's convention of array indices starting at 1, define the *neighbors* centered at (c, r) as:

$$N_{cr} = \{P_{xy} \mid c - b \leq x \leq c + b,\ r - b \leq y \leq r + b,\ 1 \leq x \leq w,\ 1 \leq y \leq h\}$$

Using the above notion of neighbors, the output may be defined as a $w \times h$ array Q with $Q_{cr} = mean(N_{cr})$ using the conventional arithmetic mean.

Conceptually, the **input** consists of a two-dimensional array of nonnegative integers (*pixel values*) and another nonnegative integer that is the *blur strength*. In more detail, the input is a sequence of at least three nonnegative integers. The first is the blur strength, while the second and third are the width and height of the image (number of rows and columns of the array). The remaining integers specify the elements of this two dimensional array in row-major order. Any missing values are treated as 0.

Conceptually, **output** is an array whose dimensions are identical to those of the input array and whose elements are the blurred pixel values. In more detail, each row of the array corresponds to a single line (newline-terminated) in the output, in array order. Each line consists of the integers in the cells within that row, in array order, with adjacent integers separated by a single spaces.

Questions

1. (1 pt.) Write your name in the space provided above.

2. (9 pts.) Provide a shortest valid input. Explain both why it is valid and why it is shortest.

3. (10 pts.) Provide output corresponding to the input of Question 2. Is the output unique? Explain your answer.

4. (10 pts.) Describe an algorithm that implements the blurring process described above. Describe your algorithm in English as precisely as possible.

5. (10 pts.) Explain why your algorithm is correct.

6. (10 pts.) Provide pseudocode, using the textbook's style as a guide, for your algorithm. Include explanatory comments and outline a proof of its correctness.

7. (10 pts.) State and justify the running time of your algorithm as a function of the blurring strength b and the number of cells $n = w \times h$.

8. (10 pts.) Repeat Question 7 for working space (memory use) instead of running time.

[additional space for earlier material]

9. (20 pts.) *On the class discussion forum*, post messages with sample inputs and outputs for this assignment. Your inputs (and corresponding outputs) should be designed to illustrate the problem as well as to test difficult cases (e.g., boundary cases or other tricky situations). Include good explanations with your samples.

10. (100 pts.) Implement your algorithm. Test and document your work carefully and submit your packaged source code and supporting documentation.

11. (15 pts.) Conduct a brief experimental study of your implementation, measuring the running time for a suitable collection of inputs. Include your test code in your electronic submission, with suitable documentation.

12. (15 pts.) Summarize your experimental results by making effective use of charts and tables. Comment on how well the experimental results match the predictions based on your answer to Question 7. Highlight any significant differences and explain them the best you can. Include these results, comments, and explanations as a single PDF file in your submission.

Submission: Submission consists of (1) a hardcopy of this assignment with answers filled in neatly and (2) a well-packaged electronic submission with answers to the programming components implemented in standard Python 3 with minimal library dependencies. Refer to the syllabus for submission instructions. Name the electronic submission using the template

`cos454-hw01-`*`lastname`*`-`*`firstname`*`-`*`pqrs`*`.tgz`

where *`lastname`* and *`firstname`* are replaced by the obvious and *`pqrs`* is replaced by a 4-digit string of your choosing. The submission should be designed so that the command `tar zxf cos454-hw01-`*`lastname`*`-`*`firstname`*`-`*`pqrs`*`.tgz` results in the creation of a directory `cos454-hw01-`*`lastname`*`-`*`firstname`*`-`*`pqrs`. In that directory should be all the source code (organized in further sub-directories as needed) as well as a README file with the usual semantics. *Do not submit any kind of non-source files* (e.g., `.pyc` files or other artifacts).