[19sp]:

* (Placeholder for todos added from Slack)

[18fa]:

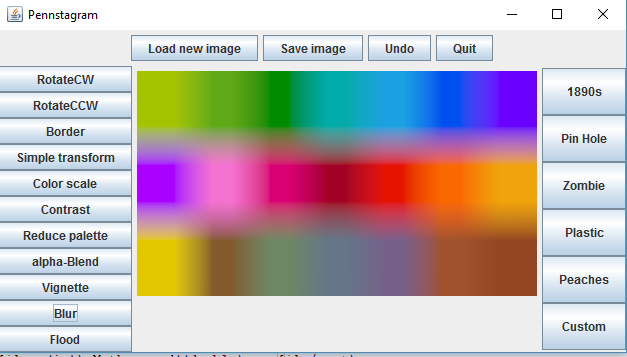
* (xuyuan): update hw HTML instructions to have a task 0 with the link to download the HW and also a link to eclipse instructions (some people had trouble finding these?)
* (jennche): its interesting that the submission test for small blur looks for a diff of 6, but the same exact test we give in manipulateTest looks for a diff of 12. That seems to be giving false positives as in <https://piazza.com/class/jktwqkemhyf5za?cid=2167>. This person also passed the larger blur submission tests even though it looks like his blur loop wasn't quite right -- that is, he didn’t go to the edge of the picture -- row < width - 1 vs row < width. Perhaps need to add a test for that.
* (jennche): maybe add to FAQs (or into the homework instructions/resources) some of the commonly seen eclipse errors -- for example, updating the location path of mytest should probably go into the instructions, making sure build automatically is turned on, cleaning the project, etc.
* (jennche): also people don't know how to run the tests/run the GUI in Eclipse -- maybe add to the instructions somewhere.
* (jennche): add to FAQs/somewhere -- alphaBlend test for MyTest depends on grayScaleAverage. If grayScaleAverage is failing, then even if alphaBlend is right, the test in MyTest for alphaBlend will also fail.
* (jennche): strongly encourage students to look at ManipulateTest first for test cases before moving onto MyTest. ManipulateTest is much easier to debug.
* (sankhe): The main problem in office hours today was that for blur, people were using the original 2-D array without considering what would happen if they changed the pixel values in place. Maybe a hint next time could make this easier.
* (jennche): A student was able to pass all submission test cases even though his distance function was this: `return Math.abs(red - px.red) + Math.abs(green - px.green) + Math.abs(blue - px.blue);`  
  which \_should\_ cause a Null Pointer Exception when px (the Pixel passed in) is null.
* (xuyuan): HW 6: Students said the ColorMap was not well explained. I think including an example above the put method would be helpful along with what default constructors are. For example, in AdvancedManipulations.reducePalette instructions line 35, we can add a few sentences like:  
    
  ```You will need to make use of the ColorMap class to generate a map from Pixels of a certain color to the frequency with which identically-colored pixels appear in the image. For example, if the Pixel p shows up 3543 times in your image, you would have an entry in your map from p to 3543. Take a brief detour to the ColorMap.java class right now to check out some methods you will use for this task. You may notice there is no defined constructor. In those cases, Java provides a default constructor, which you can call with no arguments: ColorMap m = new ColorMap();```  
    
  And in ColorMap above put, we can say:  
    
  ``` If we start out with an empty map m, calling m.put(Pixel.RED, 1) would create a new entry <Pixel.RED, 1>. If we then call m.put(Pixel.RED, 10), this would replace the old entry with <Pixel.RED, 10>. m.size() would still be 1 in this case.```
* (Jane): update hw HTML instructions to have a task 0 with the link to download the HW and also a link to eclipse instructions (some people had trouble finding these?)

-- before--

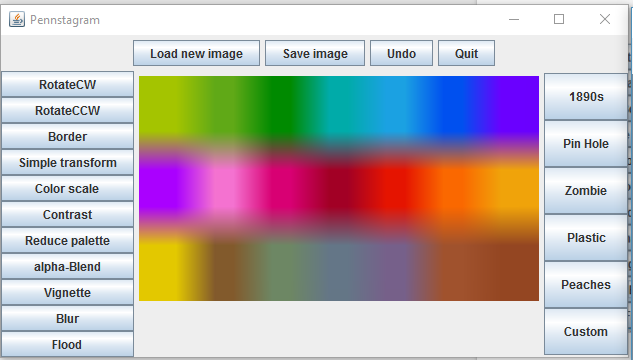
Consider this problem: <https://www.youtube.com/watch?v=LKnqECcg6Gw>

Should we add to the instructions/update our solutions/update our expected solutions/tests to account for this to be even more realistic? (could be for just blur, or for all methods).

Top image: **blur** with squaring



Bottom Image: **blur** without squaring (our solution/expected solution).



TODO Spring 2018

* ~~(MC) add `@param` description for `numColors` in `reducePalette`~~
* ~~(BD from 17fa) make sure given junit test names are consistent with java naming convention~~
* ~~(JC) update the instructions for the kudos problem (doesn't use IntQueue anymore)~~
* ~~if you mess up .equals so it always returns true you’ll pass a lot of tests even though you shouldn’t~~
* Add to FAQ: if you’re failing every test, check your equals method!!

TODO Fall 2018

* ~~(JX) be clearer about what ColorMap put does—like maybe include the fact that if it overwrites previous entries.~~

TODO Fall 2018

* ~~JC: Check on the old Pixel issue (equals, constructor?)~~
  + Talk to Jenny/Jane/Emma for the Slack channel messages regarding this issue (from 18sp, March 21, pre post mortem channel) if needed, relevant quotes posted below
  + “Equals always returns true”
  + Matt quote: “so the way to get a 100 on this assignment is to just implement getRed, getGreen, and getBlue to return constant values”
  + Palmer quote: “So the issue essentially comes down to the tests that the students write that use the broken Pixel class. We can easily fix the tests that load from images and we can easily fix all the private tests. But we want them to use the Pixel class for the tests they write because otherwise it will make them question the utility of having a Pixel class.”
  + From what I remember, solutions that were talked about:
    - (for the student to notice this issue): Provide straightforward tests to the student to make sure they aren’t doing constant values for rgb
    - Compare BufferedImage directly
      * Easy change, except we would have to load everything in from a file instead of constructing from Pixel which makes it hard?
    - Rewrite test suite to use *our* Pixel class when constructing Pixel pictures
      * Pro: maybe the point of testing, Con: Lots of refactoring
    - Add some kind of equals in our tests that makes sure you get the right RBG values
      * Pro: easy fix, Con: very hacky

Pixel p = new Pixel({20, 30, 40});

assertEquals(20, p.getRed());

assertEquals(30, p.getGreen());

assertEquals(40, p.getBlue());

* + - * ^^ this as an if statement
    - :(