CS102 – Algorithms and Programming II
Programming Assignment 6
Fall 2023

ATTENTION:

- Compress all of the Java program source files (.java) files into a single zip file.
- The name of the zip file should follow the below convention:
  CS102_Sec1_Asgn6_YourSurname_YourName.zip
- Replace the variables “Sec1”, “YourSurname” and “YourName” with your actual section, surname, and name.
- You may ask questions on Moodle and during your section’s lab.
- Upload the above zip file to Moodle by the deadline (if not, significant points will be taken off). You will get a chance to update and improve your solution by consulting with the TAs and tutors during your section’s lab.

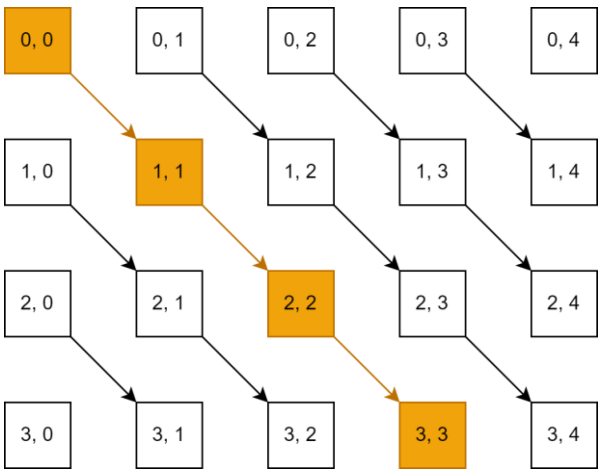
GRADING WARNING:

- Please read the grading criteria provided on Moodle. The work must be done individually. Code sharing is strictly forbidden. We are using sophisticated tools to check the code similarities. The Honor Code specifies what you can and cannot do. Breaking the rules will result in disciplinary action.

Animated Diagonal Selection Sort on Pixels

For this assignment, you will implement a Java application that sorts the pixels of images using selection sort. Selection sort finds the minimum of the unsorted portion at each turn and places it in the last position of the sorted part. You will animate the sort operation by displaying the current state of the image after each iteration of the sort.

Although a BufferedImage is stored as an array of RGB values, we can consider the 2D positions of pixels to assume different orderings. For this assignment, you will assume we have multiple arrays corresponding to the diagonal chains of pixels of the 2D image, shown in the following figure.

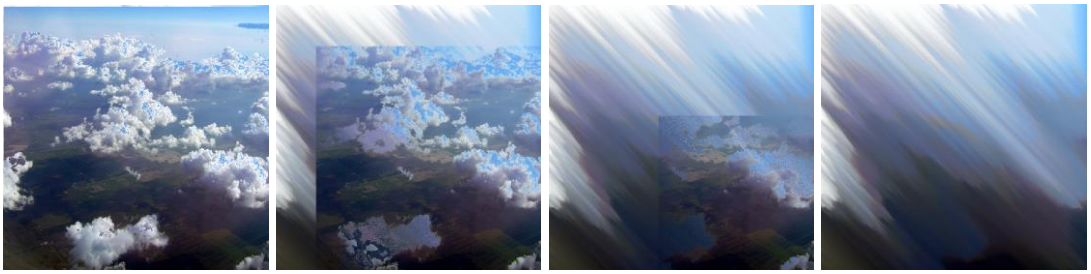


Note that the longest chain would start from the top left corner, and each neighboring chain can differ in length based on the width and height of the current image. The chains will be your arrays to sort. Keep an integer `iterationCount` to track the sort. At each turn of sort

and for each chain, the sorted part will contain `iterationCount` number of elements. Increment `iterationCount` by one after each iteration. You will use the brightness values of pixels to compare the pixel values, i.e., the pixel with the minimum value. You can calculate the brightness using the following luminance formula:

$$\text{Brightness}(R, G, B) = (0.2126 * R + 0.7152 * G + 0.0722 * B)$$

You will implement a reverse insertion sort where the pixels of higher brightness (the ones that are closer to white) are positioned towards the top left parts of the image.



Your applications should load at least three images at the beginning. You may use `ImageIO.read(new File("filename.png"))` method to load a BufferedImage from a file. By using the left and right arrow keys on the keyboard, switch the current image with the next or previous one in a circular manner. If we change into a different image and return to the previous image, the sort should continue from where it is left. You may keep the `iterationCount` for each image in a separate array to support this functionality. Check the following video as an example of this behavior: https://vimeo.com/888198920.

You should also resize the current image to fit either the current height or the width of the frame. So as not to distort the image, calculate one resize factor to multiply with the image's width and height. Suppose we want to resize a BufferedImage of size (width, height) by a factor f; you can draw the resized image on your Panel or Frame using the following method:

```
g.drawImage(image, 0, 0, f * width, f * height, null);
```

Check the following video as for the resize behavior: https://vimeo.com/888199099

Finally, using the R key on the keyboard should reset the current image, meaning you will load this image again and reset its `iterationCount` to start sorting from beginning.

Preliminary Submission: You will submit an early version of your solution before the final submission. This version should at least have the animated diagonal selection sort on one image functional.

You will have time to complete your solution after you submit your preliminary solution. You can consult the TAs and tutors during the lab. Do not forget to make your final submission at the end. Even if you finish the assignment in the preliminary submission, you should submit for the final submission on Moodle. **Not completing the preliminary submission on time results in a 50% reduction of this assignment's final grade.**