Registration Letter Group Organization Algorithm

...

Christian Lussier, Ben Watto, Mikey Spurr

Overview of Project

- Taking the current registration system and improving it's process.
- Randomizing the groups initially.
  - Dividing the groups into halves, then further into sub-halves (quarters), then divided into sub-halves again (eighths).
  - Going through each letter group and sorting them based off of GPA.
- Each letter group has the potential to register within the first half.
  - The eight groups are divided in half, at the beginning of each semester, the half that registered first will register in the second half, and vice versa.
  - Within each of the sub-halves the highest avg GPA with go first within those sub-halves.
- Producing a fair and more efficient way to make sure every letter group will register within the first half, as well as the potential to be the first group to register.

Project Motivation & The Problem

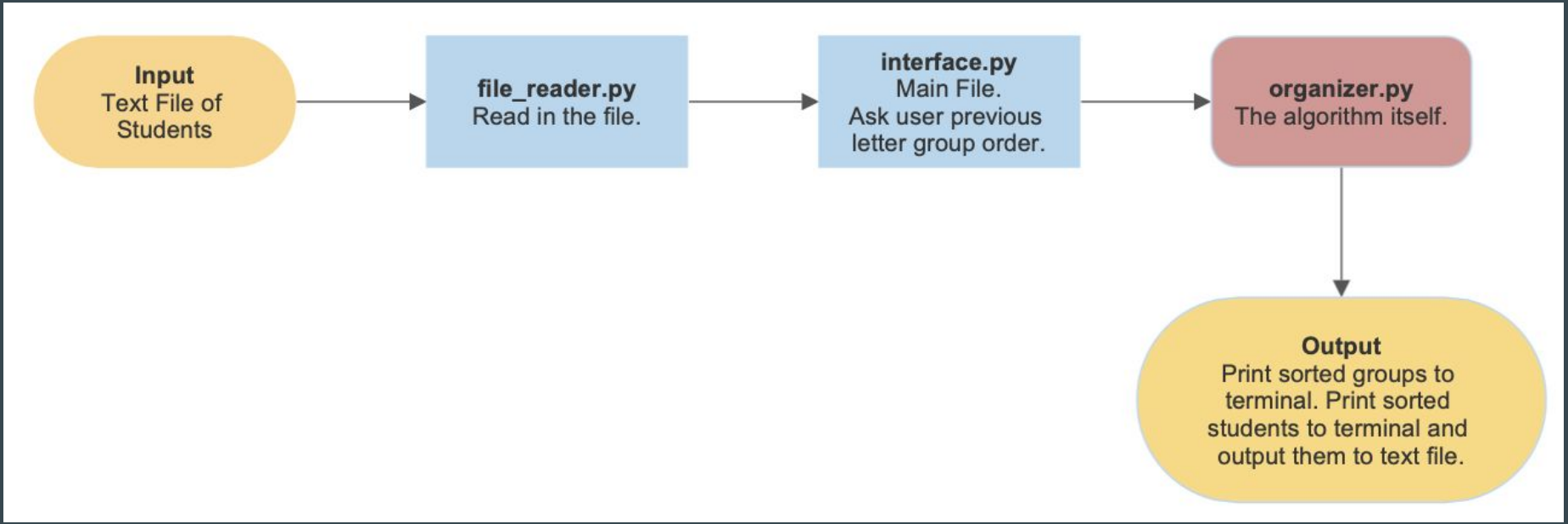
- Track 2 seemed interesting.
- Students register by class and their assigned letter group within each class.
- We feel the way letter group order is determined is flawed and unfair.
  - Letter group order randomly determined.
  - If are in the second half of letter groups last semester, you could be there again.
  - You may never register as one of the first one or two letter groups.
- Real-World Impact:
  - We all participate in this system every semester to register.
- Motivated to make a prospective algorithm that would make letter group ordering process more fair.

Date	Starting time	Registration Class	Letter Group
Monday, 4/8	7:30 AM	SR	B
	1:00 PM	SR	A (and B)
Tuesday, 4/9	7:30 AM	SR	D (and B, A)
	1:00 PM	SR	C (and B, A, D)
Wednesday, 4/10	7:30 AM	SR	F (and B, A, D, C)
	1:00 PM	SR	E (and B, A, D, C, F)
Thursday, 4/11	7:30 AM	SR	H (and B, A, D, C, F, E)

Approaches

- To start it off, we were bouncing ideas off of each other and were drawing diagrams for how we thought that we should implement this idea.
- We then came up with some pseudocode that helped us determine how we were going to implement our idea.
- After we created a file of sample students and code to read it in.
- Decided to use Python for implementation language.
- We then created a main file to run the algorithm.
- We looked over previous algorithms from class to base our current algorithm off of and then we created our own sorting algorithm for this project.

Project Flow Diagram



Challenges

- Getting the algorithm to work properly.
  - Lots of trial and error.
- Issues correctly importing student files.
  - Added a space before content the algorithm couldn't handle.
- Failed to implement a student priority functionality.
  - Undermined our algorithm.
- Analyzing the algorithm.

Analysis

$O(n^3)$ = cubic

Input Size	Running Time	Ratio
10	0.00016283988952636 72	0
20	0.00035595893859863 28	2.19
40	0.00057387351989746 09	1.62
80	0.00087904930114746 09	1.52
160	0.00162601470947265 62	1.83

Results

- Letter groups are properly sorted.
- Successfully sorts students based on new letter group order.
- Overall, our algorithm & program function as intended.
- Calculated the tentative worst-case time complexity - O(n^3).
  - **Also ran experimental analysis.**
- Still some minor improvements that could be made.

Concluding Remarks & Possible Future Works

- In the future, we will allow users to enter the previous letter group order.
- If we had more time, we could consider adding in a priority ranking setting.
  - This priority setting would take into account that certain students have deadlines that they need to meet in regards to taking certain courses.
- Overall we think that this algorithm could be quite beneficial to our school's current registration system.
- With our algorithm, every letter group has the potential to register first. Unlike our school's current system where it is completely randomized.

Demo Time!
Any Questions?