Intuition

First, we need a clear way to measure improvement. We create a lambda function called calculateGain to compute how much the pass ratio of a class would increase if an extra student were added. This provides a consistent metric to evaluate and compare the potential impact on different classes.

Next, we build a max heap. Each class is represented by a tuple containing its negative gain (to simulate a max heap using Python's default min heap), along with its current number of passed and total students. This ensures that the class with the highest gain can always be retrieved efficiently.

We then distribute the extra students iteratively. At each step, we pop the class with the highest potential gain from the heap. We simulate the addition of one extra student to this class, updating its number of passed and total students. We then recalculate its gain and push the updated class back into the heap, allowing us to continuously adjust to the changing gains of each class as students are allocated.

After all extra students are distributed, we compute the final result. By popping all classes from the heap and summing their current pass ratios, we calculate the total pass ratio. Dividing this sum by the number of classes gives us the average pass ratio.