The algorithm used is based on max flow. The problem of finding if there is a valid combination is reduced down to seeing if there is students*required number of classes amount of flow from courses to a sink node. My algorithm heavily relies on hashmaps to map students to classes, classes to how many students they can hold, and classes and students to indexes and vice versa. I use an adjanency matrix if size student x classes that has students to point classes. I don't do a square because classes will never point back, so I save space. For every edge, I see if the class is full and if the student has reached the required number of courses yet. If not, then add one to flow from classes to sink and one from source to student. This is repeated until no more space in classes and all students have their needed number of courses. The resulting flow should equal students * require amount of class. The runtime of my algorithm is $\Theta(E * f)$ because we have to check all edges and push decide whether to push flow through each one.