Personally, I would say the first part of this assignment went smoothly. I was able to create however many schedules that were as random as python’s random library could make them. The issues started to rise once I got to the crossbreeding section of the code. At first, I wasn’t quite sure the best way to cross breed. If I should pick randomly or pick from the upper ten or so. I wanted to generate the best offspring, but I also wanted to avoid the possibility of inbreeding. Also, mutations were messing my program up a lot. This was because my mutation rate was way too high, and I didn’t realize that till late in development. Ultimately, I ended up decaying my mutation rate exponentially over time, so mutations were far more likely to happen early on since it seems most mutations are lethal and provided little to no positive change.

I am personally very happy with the schedules that my program is producing. Based off the requirements in the initial document and my fitness function most if not all requirements seem to be getting met when I generate a schedule. So far, I have yet to generate a schedule where one professor is needed at one time slot, a room is too small or far too big, facilitators rarely travel from beach to roman right after the previous class. As it seems no two classes are in the same room at the same time. I think overall the important aspects of a schedule such as a professor being at multiple places at one time, or room double booking, which would make a schedule impossible, have yet to occur and my program does a consistent job at making sure schedules like that are penalized and are not generated. I believe there are always exceptions, and it is entirely possible that a poor schedule could be generated if say a mutation happens very late in life despite it being unlikely.

When it comes to improving this program, I think it works great for this problem on a small scale, but if we were to turn this loose on a bigger problem with a much bigger data sample, it would take far too long. Some ways to improve time could be how we keep the best parents and children between generations, maybe instead of keeping the best upper half we have the upper half compete amongst themselves in a tournament style of staying alive, where only those who “live” can stay in the gene pool. Another option would be parallelization. The fitness function is the largest section and is made up of the most nested for loops out of any section in my code. We could speed this process up quite a bit if parts ran in parallel. If we utilized multi-threading to run say three different parts at once that could be beneficial. We have one section calculating the fitness of professors, one section calculating the fitness of time, and one section calculating the fitness of the room. Having these three sections run at the same time and then converge to calculate one fitness score could produce an answer much quicker than waiting for the one thread to get through the whole section and do everything sequentially.

There is also one aspect of fitness that I think should be added. Dr. Tyler works on the school board, my schedule doesn’t penalize him for taking more than 2 classes at the moment, but if he is on the board I feel as if it should because if he is assigned the acceptable max number of classes, that being four, he would have a lot of time taken out of his day that I think could conflict with his other responsibilities. So, adding a penalization for having more than two or three classes a day for Dr. Tyler I think could be an interesting check to add.

The last thing I would like to add about this project is that overall, it was a very fun project to learn how to implement. I see how this could be turned loose on a lot of other problems that have constraints. I was even thinking about trying to figure out if this could be turned loose on the Einstein problem, however I think that would almost require too much mutation for this program, as every aspect of the word bank is needed. But overall, it has been my favorite program and by far has been one of the more interesting parts of this AI course for me.