ALGORITHM II THOUGHTS

ROOM OBJECTS – has a name, capacity, and the time it opens and closes (24 hour clock).

EVENT OBJECTS – has a name, starting time and end time (decimal of 24 hour clock), and attendees

**How do we determine the ‘best’ change to make to fit in the most unassigned events?**

Find a way to find all combinations of events in rooms, whichever has the least number of events left over is the ‘best’:

**How can we get the best answer without it taking forever?**

If there are hundreds of rooms/events then this method will take forever because there are so many combinations. We will either need to come up with a way to simplify this or another plan.

Could we somehow do it recursively? Could we go through and start making schedules, having it add an event, go back and add another, etc. until it has a full schedule and then go back to the first event added and find a different second one to add and then find another schedule with that combo? The list/combo that is the longest is the one that fits the most events and therefore the ‘best’. It’s basically still finding every combination so it may take forever though. I don’t think the solution just described is recursive but maybe there is a way to come up with one????

loop(events, rooms, combos)

for every i in events #go through all events

for j room in rooms #go through all rooms

if event time fits in room time && room capacity >= attendees #if event fits in room

add (event, room) to start[x] #add tuple to list containing the start of combos

… (???)

return longest list in combos as ‘best’ schedule

recursive(events, rooms, combos = [[()]])

???