TEAM PLACEMENT PROCEDURE

1. Work closely with Recruitment and Admissions managers to distribute survey, gather responses, and ensure up-to-date corps counts and survey data
2. Clean survey export to ensure headers match
3. Clean address column and compute commute times
4. Clean roommates column, ensure each roommate name matched to a full ACM name
5. Set manual placements. Things to consider:

* physical ability and school accessibility - make a survey item for this
* ACMs with children
* 2YACMS are manually placed in CHI
* Review distant ACMs and make manual placements to ensure reasonable commute (double check Span ability, HS eligibility, etc.)
* Review ACMs who have attended CY schools, make manual placements as necessary (double check Span ability, HS eligibility)

1. Run with only Spanish Speakers. Write placements to Manual Placements column.
2. Run with everyone else
3. Present to IDs and IMs, including paper profile sheet of each ACM in folders grouped by school assignment (with photos if possible)
4. Allow manual swaps where needed with awareness of all firms/soft constraints
5. Eval Team double-checks constraints and works with MDI to adjust team sizes as ACMs drop out and enroll, making placements as necessary
6. Present placements to corps

ALGORITHM IMPROVEMENTS AND ADJUSTMENTS

* + Spanish Speakers
    - We set firm targets by calculating the projected number of students at each school that had limited English ability (based on FY17 data provided by Chicago Public Schools on student body size, %hispanic, %Asian, and %Limited Enlgish - could be further improved by adjusting calcs to only the grades we serve for each school). We then distributed our number of Spanish speakers proportionally to each school’s need.
    - We reviewed these targets with Directors and made slight adjustments based on past experience.
    - In the first round of team placements, Spanish speakers were placed first, with only concern for commute time and firm constraints. Their placement was then set as a Manual.Placement when the remainder of the corps were run. This was done to improve swap efficiency.
    - One drawback of this approach was that some Spanish Speakers have long commutes, since there are only so many slots at each school and each Spanish Speaker must go to a school with Spanish Speaker need.
  + Eligibility Function - added function that calculates all ACMs' eligibility to serve in all schools. This approach helps ensure that more swaps are valid, and removes deep craters from the search space. This will be important if sites are only running the algorithm once. This was done in two functions:
* eligible\_placements\_func: calculates all ACMs’ eligibility to serve at all schools based on factors that are independent of team composition (HS eligibility, prior relationships with TL and IM, Spanish speakers set to only be eligible to serve at schools with Spanish Speaker target > 1, manual placements set to only be eligible to serve at manual placement school).
* elig\_plcmnt\_iter\_func: calculates all ACMs’ eligibility to serve at all schools based on factors that depend on team composition (roommate conflicts, prior relationship conflicts). This function runs for each iteration of the algorithm.
* Now when an ACM1 is selected for a swap from School1, the other School2 is selected based on AMC1's eligibility, and an ACM2 is selected that is eligible to serve at School1. This was done in order to minimize the number of swaps that trigger a "blown-up" score. Previously, if a Spanish speaker were randomly placed at a school, but their commute to that school was terrible, there was a very low chance that their placement would ever change. Either they would have to be selected at random AND have another Spanish speaker selected at random for a swap, or the algorithm would have to accept a score of 1e13 before finding a better placement. This is a solution that is also quite slow, unfortunately. It dropped speeds to 1,000 iterations per minute. However, more of those iterations were valid, and helped to push placements further with each swap.
  + Blown up scores still exist for Spanish and Gender.
  + Ethnicity Score - now seeks to minimize the proportion of the team that is the predominant ethnicity
  + acm\_id and sch\_id now calculated entirely within the script (except in the case of commutes). This was done to fix errors in matching when id's were handled manually. acm\_id must always match to the row index, and this was broken if an ACM dropped out and their row was deleted without re-setting acm\_id's.
  + School Input now .xlsx, becuase that version allows online collaboration, which was a useful way to gather info from staff

IJ PLACEMENTS

* + In Chicago, our partnerships were not confirmed in time to make school placements on day 1, so we elected to start in IJ teams. I made these teams by seeking to maximize commute times (and achieve ethnic/gender diversity, prevent roommates and prior relationships). The theory was that if each ACM is assigned to a IJ leader (TL) whose school is very far away, there is less chance that IJ-conflicts will arise in team placements.
  + IJ placements before school team placements is not advised and adds a potentially large constraint on where ACMs can be placed in schools
  + Ended up being less of a constraint than I thought, but it was an enormous pain. I ended up creating a soft score measure that helped minimize IJ conflicts in school team placements. We ended up with 7 conflicts.
  + To make IJ placements for new hires and to re-assign conflicted IJ placements, I set previous IJ placements as Manual.Placements, and I set school team placement TLs as prior relationships. I then ran the algorithm as normal. This approach assumes that TLs are the only folks who lead IJ spaces.

FUTURE IMPROVEMENTS

* + IJ placements more neatly integrated. Suggested avenues to build out:
* set IJ team sizes as a function of the number IJ leaders and corps size
* Are TLs the only folks who lead corps-level IJ spaces at all sites?
* "Do nothing with IJ Teams."
* "Create IJ Teams after making school teams."
* "Create IJ Teams Only."
* "IJ teams already exist. Prevent IJ teammates from serving on school teams."
  + Consider demographics of TLs
  + be very clear about deadlines for personal conflicts/roommates/addresses, and the process for gathering this information from Corps after the initial survey
  + improve speed (lapply is likely culprit, could also convert some things to datatable functions)
  + better way to distribute by gender (Esp. gender-nonconforming)?
  + way to distribute males early? - once male targets achieved, only allow male-male swaps?
  + May need to reset temperature schedule
  + review what acceptable placements looked like at each site to build a model which can produce comparable results in a single run
  + error handling, like:
    - "Number of Spanish Speakers is less than Number of Spanish Speakers Required"
  + survey to determine how accurate commutes were
  + Commutes
    - calculating commutes takes too long, should be run as a separate process from the PowerBI Algorithm
    - calculating commutes has an annoying random error. Could write some script features that mitigate this nuisance.
  + give slight favor for ethnicities that match school ethnicity