Vinay Parikh
VHP286
EE360C

G-S Project 1

Pre-Loop

- Create Hashmap for each internships matching
- an annaylist to store engagement of each student
- Hashmap for the slots in each internship
- inverse preference list for internships for constant access time
- Arraylist to store the place of each student in their proposals

Loop:

```
outer -
    Sets the current student
    the loop runs till students are satisfied
Inner-
    get the internship from the place of his next proposal
    If the internship has empty spots :
            - Student engament to that internship
            - decrease remaining slots
            - update pairing for the internship
            - increase happy students by 1
            - decrease slots for that internship
        else
            - check if the student can replace another student
            - get lowest pref student of that internship
            - If switch needed switch
            - update all data structures
        do this for all proposals
        increment happy student if no remaining proposals left
    return
```

analysis:

The algorithm is not O(m·n) becuse when I want to switch students I check for the ~~min~~ lowest prefered of the current paired students. Beause of this In worst case I have to check all students + find the lowest prefered paired student. However this case is not common, generally the algorithm runs closer to O(m·n) becuse not ~~slots~~ all Students have 1 internship.