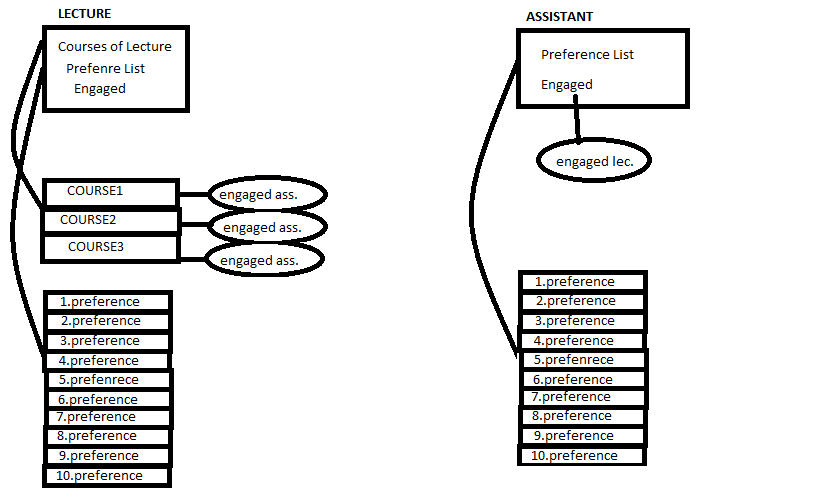
1. **Program Structure**

I created lecture object and assistant object. Firstly, my lecture object includes courses of lecture and preference assistant list of lecture. Assistant object has preference course list of assistant.



I created empty engaged place for keeping match result.

Firstly, I read reference lists and courses of lectures from text files. Then, I determined engaged = -1 (empty) for all courses and assistant and began Gale-Shapley algorithm.

**while** ∃ *free* man *m* who still has a woman w to propose to {

w = m's highest ranked woman to whom he has not yet proposed to

**if** w is *free*

(m, w) become *engaged*

**else** some pair (m', w) already exists

**if** w prefers m to m'

(m, w) become *engaged*

m' becomes *free*

**else**

(m', w) remain *engaged*

}

}

lecture\* readlectures(int\*,int\*,string);

this function reads courses of lectures from file and writes to lectures objects.

assistant\* readassistant(int,int\*,string);

this function reads assistant preference course list from file and writes to assistants objects.

void lecturelist(lecture\*\* lecture,int,int,string);

this function reads prference assistant list of lecture from text and writes to lectures objects.

void gale\_shapley(lecture\*\*, assistant\*\*,int,int,int);

this function matchs assistant and courses according to lecturer optimal solution.

void gale\_shapley2(lecture\*\*, assistant\*\*,int,int,int);

this function matchs assistant and courses according to assistant optimal solution.

void refreshengaged(lecture\*\*,assistant\*\*,int,int);

this function reset all matching datas (engaged).

1. **Complexity**

**while** ∃ *free* man *m* who still has a woman w to propose to {

w = m's highest ranked woman to whom he has not yet proposed to

**if** w is *free*

(m, w) become *engaged*

**else** some pair (m', w) already exists

**if** w prefers m to m'

(m, w) become *engaged*

m' becomes *free*

**else**

(m', w) remain *engaged*

}

}

We have while loop. So length of preference list is very important for complexity.

Length = n;

O(n^2)

1. **Explain**

When lecturer optimal, lectures wants assistants according to preference list. İf it doesn’t get it, it try next preference assistant from its list. So it always try worse preference. But you begin from best preference so it is your optimal. For example, you like 5 boys and liking level is different for each boy. You always want first boy of your list. But this boy has a girlfriend. Then, you try second boy of your list. This time boys accepts your offer. You are happy. But it is not long time. Another girl comes and get your boyfriend. Now you are alone again. What will you do? Of course, you try third boy of your list ☺ it continues likely. But you choose worse preference. You try next boy always and boy waits girls. They doesn’t go their first preferences. Finally, you select your optimal preference but your engaged is not lucky as you. Because he doesn’t select his optimal preference. So matching is optimal for who selects own engaged.

Weight[A]= (N – n)^2

N = number of your preference

n = order of your preference in your preference list

If you select best preference n =0. N is constant. So if you choose worse preference your weight decrease. If you have large weight, you choose more optimal preference for you.