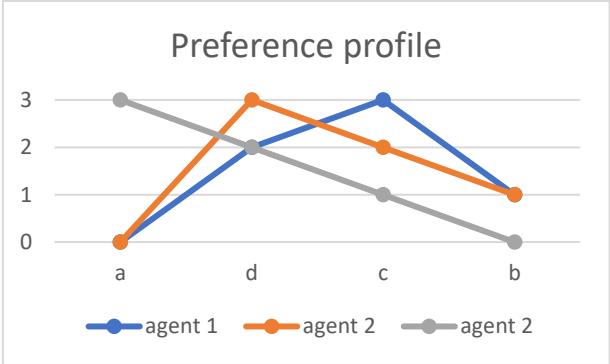
Ass 2

Question 1

$$1 : c \succ d \succ b \succ a$$
$$2 : d \succ c \succ b \succ a$$
$$3 : a \succ d \succ c \succ b$$

1. Prove or disprove that the preference profile is single-peaked with respect to some order of alternatives.



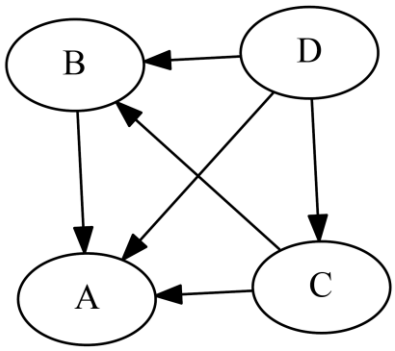
This profile is single-peaked with order a>d>c>b.

2. Prove or disprove that a Condorcet winner exists for the preference profile.

The Condorcet winner is d.

It wins two times comparing to a and c, and wins three times comparing to b.

3. Compute the pairwise majority graph for the preference profile.



4. Compute the Top Cycle set for the preference profile.

The top cycle set is $\{d\}$.

Question 2

Consider a resource allocation setting in which n agents have positive additive utilities over m > n indivisible items. Prove or disprove the following.

1. The allocation that maximizes utilitarian welfare is Pareto optimal.

First, assume that the allocation that maximizes utilitarian welfare is not Pareto optimal.

This means that there exists an allocation where j items can provide a higher utility while not affecting the total utility provided by other items.

Hence there exists an allocation with higher additive utility.

Thus, the utilitarian welfare of the original allocation is not maximized.

2. If an allocation is Pareto optimal, it is envy-free.

	o_1	o_2
1	5	1
2	4	2

Consider the example above where the pareto optimal allocation is:

$$X_1 = \{o_1\}, X_2 = \{o_4\}.$$

However, $u_2(X_1) = 4 > u_2(X_2) = 2$. So it's not envy free.

3. If n = 2, envy-freeness and proportionality are equivalent.

Assume that an allocation X is envy-free.

Then for each i, we have $u_i(X_i) > u_i(X_j)$ for all j.

Because there are only two agents, we have $u_i(X_i) > \frac{\Sigma^{j=1,2}(u_i(X_j))}{2}$. So, it implies proportionality.

Also, assume that an allocation X is proportional.

Then for each i, we have

$$u_i(X_i) > \frac{\Sigma^{j=1,2}\left(u_i(X_j)\right)}{2} \Rightarrow \frac{u_i(X_i)}{2} > \frac{u_i(X_j)}{2}(i \neq j) \Rightarrow u_i(X_i) > u_i(X_j)(i \neq j)$$

So, proportionality implies envy free in this case.

Therefore, we can conclude that envy-freeness and proportionality are equivalent.

4. The sequential allocation algorithm, in which agents arrive in order (1,2,3,...,n) ∗ and are given a most preferred unallocated item, is strategyproof.

Let $n = 2, m = 4$, and the preference profile are listed below:

	o_1	o_2	o_3	o_4
1	5	3	4	1
2	4	1	**2**	3

Original preference profile

	o_1	o_2	o_3	o_4
1	5	3	4	1
2	4	1	**5**	3

"fake" preference profile

The original allocation for this case is:

$$X_1 = \{o_1, o_3\}, X_2 = \{o_4, o_2\}.$$

The "fake" allocation is:

$$X_1' = \{o_1, o_2\}, X_2' = \{o_3, o_4\}.$$

Where $u(X_2') > u(X_2)$.

Therefore, the sequential allocation algorithm is not strategyproof.

Question 3

$$1 : e, b, a, c, d \qquad a : 2, 4, 3, 5, 1$$
$$2 : b, a, c, d, e \qquad b : 3, 2, 4, 5, 1$$
$$3 : a, b, c, d, e \qquad c : 3, 2, 4, 5, 1$$
$$4 : a, b, c, d, e \qquad d : 5, 2, 4, 3, 1$$
$$5 : d, b, c, a, e \qquad e : 1, 2, 3, 4, 5$$

1. Find the outcome matching. Prove or disprove that the resultant matching is Pareto optimal for the students.

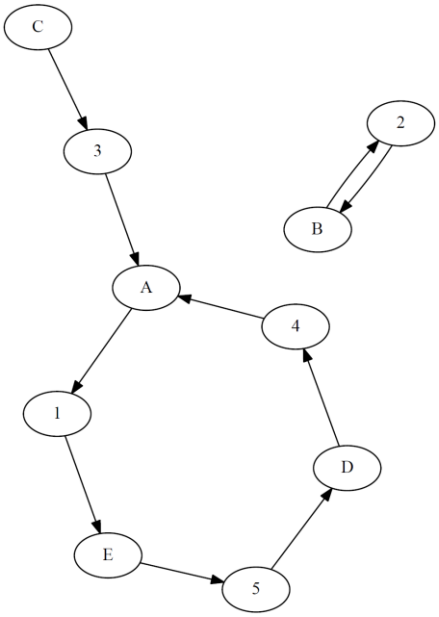
i) 1 apply to e, accepted.
ii) 5 apply to d, accepted.
iii) 2 apply to b, 3 and 4 apply to a.
iv) a rejects 3 in favour of 4.
v) 3 apply to b, accepted.
vi) 2 is then rejected and apply to a, accepted.
vii) 4 rejected by a and b, apply to c, accepted.

Result: {{1, e}, {2, a}, {3, b}, {4, c}, {5, d}}

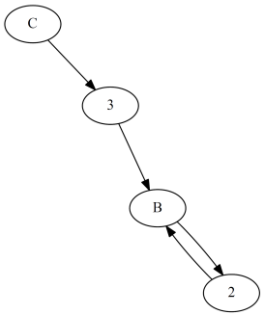
The matching is not pareto optimal for students, because if we send student 2 to school b and 3 to school a, they will achieve a more overall utility without harming the utility of other three students.

2. Apply the Top Trading Cycles (TTC) Algorithm.

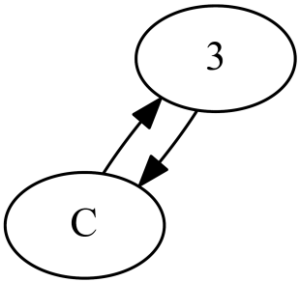
Initially, the graph is shown below.



Remove cycle A->1->E->5->D->4->A. Allocate 1 to E, 5 to D, 4 to A.



Remove cycle B->2->B. Allocate 2 to B.



Allocate 3 to C.

The result is: {{1, e}, {2, b}, {3, c}, {4, a}, {5, d}}.

3. Give three reasons, why the deferred acceptance algorithm is preferred.

The first reason is that deferred acceptance algorithm considers both the preference profile of school and students. The result is not fair to schools. This problem is two-sided matching.

The second reason is that the assumption of student 1 is arranged to a, 2 to b … 5 to e is not reasonable. And different arrangements may lead to different results.

Justified-envy is eliminated in deferred acceptance algorithm.

The last reason is that the DA algorithm is relatively space costless.

3. Give one reason, why TTC is preferred.

TTC is Pareto efficient on students' side, if we want to satisfy student requirement as much as possible, TTC is a better choice.

Question 4

1. Write down the ASP encoding for the preference profile in Question 1.

```
rank(1, c, 1).    rank(2, d, 1).    rank(3, a, 1).
rank(1, d, 2).    rank(2, c, 2).    rank(3, d, 2).
rank(1, b, 3).    rank(2, b, 3).    rank(3, c, 3).
rank(1, a, 4).    rank(2, a, 4).    rank(3, b, 4).
```

2. Write an ASP program condorcet.lp that takes in a preference profile as input, and outputs the Condorcet winner in the predicate condorcetWinner/1.

```
% count number of times candidate X wins head-to-head against Y.
win(X, Y, C) :- C = #count {I: rank(I, X, C1), rank(I, Y, C2), C1<C2},
rank(IX, X, CX), rank(IY, Y, CY), X!=Y.
% Y is dominated when win fewer times than X.
dominated(Y) :- win(X, Y, C1), win(Y, X, C2), C1 > C2.
% condercet winner is the one who never been dominated
condorcetWinner(W) :- not dominated(W), rank(V, W, K).
```

3. Write an ASP program condorcetborda.lp.

```
% count number of times candidate X wins head-to-head against Y.
win(X, Y, C) :- C = #count {I: rank(I, X, C1), rank(I, Y, C2), C1<C2},
rank(IX, X, CX), rank(IY, Y, CY), X!=Y.
% Y is dominated when win fewer times than X.
dominated(Y) :- win(X, Y, C1), win(Y, X, C2), C1 >= C2.
% condercet winner is the one who never been dominated
condorcetWinner(W) :- not dominated(W), rank(V, W, K).

calborda(0) :- condorcetWinner(X).
calborda(1) :- not calborda(0).

% Calculate score for each rank
score(I, X, N-C) :- rank(I, X, C), N = #count {X' : rank(I, X', C)}.
% Calculate total score for each candidate
total(X, T) :- T = #sum {S : score(I, X, S)}, score(IX, X, SX).
% Get highest score
highest(S) :- S = #max {Score : total(X, Score)}.
% List the candidate with highest score
satisfied(X) :- total(X, H), highest(H).
% Set the name with first alphabetical order winner
bordawinner(X) :- calborda(1), X = #min {Name : satisfied(Name)}.
#show bordawinner/1.
#show condorcetWinner/1.
```

4. Write an ASP program cbmanipulation.lp.

Please see the attached file.

Output: original/3, fake/3.

Shows all possible results.