b] True

→ As (m,w) is a nice then if we break them appart then it will create instability

→ Hence for every stable matching (m,w) should be it's part.

a] False.

$m_1 \Rightarrow (w_2 \quad w_1)$ $\qquad$ $w_1 \Rightarrow (m_1, m_2)$

$m_2 \Rightarrow (w_1 \quad w_2)$ $\qquad$ $w_2 \Rightarrow (m_2, m_1)$

Pairing $\Rightarrow$ (m_1, w_2) (m_2, w_1)

It is stable but does't have nice couple.

c] True (assuming that there are n-mens and n-women)

→ If every one is a nice couple then, that means every one is paired with their first choice

→ If we try to alter any pair it will cause unstability

→ Hence only one stable matching is present.

d] False

Consider:

$m_1 (w_3, w_1, w_2)$ $w_1 (m_3, m_1, m_2)$

$m_2 (w_3, w_2, w_1)$ $w_2 (m_3, m_2, m_1)$

$m_3 (w_3, w_1, w_2)$ $w_3 (m_3, m_1, m_2)$

stable matching ⇒ $(m_1 w_1)$ $(m_2 w_2)$ $(m_3 w_3)$

This is the only stable matching possible. and all pairs are not nice couple.

Date

Q2) Lets consider there are 4 people

A B C and D

⇒ Their preferences

A ⇒ (B, C, D)
B ⇒ (C, A, D)
C ⇒ (A, B, D)
D ⇒ (A, B, C).

possible pairing ⇒ (1) A - B and C-D
B - C causes unstability
(2) A C and B - D
A - B causes unstability
(3) A - D and B - C
A - C causes unstability

Hence there is not stable option.

Date

Q3] let number of ships = n and S be the set of all ships let p = No. of port and P be the set of ships.

while (P ≠ empty)

{

→ Select any port from P let it be P_i

→ Check which ships comes last to that port P_i (In that month)

→ ~~Shedule the~~ let that ship be S_i

→ shedule the the truncate of S_i on port P_i

→ Remove S_i from S and P_i from P

// Note: when we check which chip comes last to

// P_i we have to check it from current

// S

}.

Q4] Part 1: Prove (Z)

Let's assume Z is unstable due to instability (m, w)

① → Since X and Y are stable (m, w) must have been part of X and Y

② → Now as Z selects one of from X or Y which ~~ever~~ is more prefered by m

③ → So let's consider without loss of generality that let us consider in X m is paired w_x^m and in Y m is paired with w_y^m

④ → Since in Z (m, w) is unstable. that means in Z w is paired with some m'

⑤ → that mean (m' w) is present in Z.

⑥ → But if (m', w) is present in Z it should have been present in X ~~and~~ or Y as well i.e $(m', w) \in X$ or $(m', w) \in Y$

⑦ → But X and Y contain (m, w) which is contradiction

∴ our asumption is wrong ∴ Z is stable

Page No.
Date

Q4] Part 2 : ~~Disprove~~ Prove (Z')

⇒ We will use similar approach like in the previous question (Part 1)

→ let's assume Z' is unstable
unstability (m, w)

→ Since X and Y are stable they must have (m, w)

→ As (m, w) is not present in Z it must be paird with some other man let it be m'

→ without lossing generality if (m', w) is in Z then it should be part of X or Y

→ But X and Y have (m, w) which is a contradition,

→ Hence our assumption is wrong

Z' is stable.

Date

Q5]

a] $\left(\frac{3}{2}\right)^{n} \leq 2^{\frac{n}{3}} \leq 3^{\frac{n}{2}}$

b] $\log(n) = \ln(n) = \log(n^2)$

c] $2^{\log(n)} \leq 2^{2\log(m)} = n^{\log 4}$

d] $\min(50n^2, n^3) \leq \max(50n^2, n^3) \leq 50n^2 + n^3$

e] $\left\lceil \frac{n^2}{20} \right\rceil = \left\lfloor \frac{n^2}{20} \right\rfloor = \frac{n^2}{20}$

Q6]

a] $\Rightarrow$ The outer two for loop make the run time $O(n^2)$

$\rightarrow$ The, the process of adding the entries inside these loop contribute $O(n)$

$\rightarrow$ The run time of the Algo $\Rightarrow O(n^3)$

$O(f(n)) = O(n^3)$

Total number of times the first loop run

b] $\Omega \Rightarrow$

for each value of i the the second loop will run for this number of time

No. of times the most inner loop run

$$\sum_{i=1}^{n} (n-i) \sum_{j=i}^{n} 1$$

$$\sum_{i=1}^{n} (n-i)\, n$$

$$n \sum_{i=1}^{n} (n-i)$$

$$n\left((n-1) + (n-2) + (n-3) \cdots + \cancel{} + 1 + 0\right)$$

$$\Omega \Rightarrow n\,(n-1)\,\frac{(n)}{2}$$

$\therefore \Omega \Rightarrow n^3$

$\therefore$ run time $\Rightarrow \Omega(n^3) = \Omega(f(n))$

$= \Omega(n^3)$

Page No.
Date

c] $f(n) = n^3$

$$\lim_{n\to\infty} \frac{g(n)}{n^3} = 0$$

$\therefore$ The possible value of. $g(n) = n^2$

Algo.

```
sum = 0
for ( i=1 ; i<=n ; i++)
    sum = A[i]
    for ( j= i+1 ; j<=n ; j++)
        sum = sum + A[j]
        B[i, j] = sum.
```

Time complexity = $O(n^2)$.