21k-3881, BSE-5A

DESIGN AND ANALYSIS OF ALGORITHM

ASSIGNMENT = 05

Q1) (a)

⇒ **P problem** = Problems that can be solved by deterministic turning machine in polynomial-time.

⇒ **NP problem** = Problems that can be solved by non-deterministic turning machine in polynomial-time.

⇒ **P=NP** :-

P=NP means whether an NP problem can belong to class P problem. In other words, whether every problem whose solution can be verified by a computer in polynomial time can also be solved by a computer in polynomial time.

(b)

Ans = If problem is NP-complete, there is very likely no polynomial-time algorithm to find an optimal solutions. The idea of approximation algorithms is to develop polynomial-time algorithm to find a near optimal solution.

(c)

X is NP-hard if every problem $Y \in NP$ reduces to X

$X \notin P$ unless $P = NP$

Subset Sum: given n integers $A = \{a_1, a_2, \ldots, a_n\}$ and target sum t, is there a subset $S \subseteq A$ such that

$$\Sigma S = \sum_{a_i \in S} a_i = t$$

→ weekly NP-Hard

(d)

3SAT = given Boolean formula of the form $(x_1 \vee x_3 \vee \bar{x}_6) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_7)$

You can set variables $x_1, x_2, \dots \rightarrow \{T, F\}$

Such that formula = T ↘ satisfying assignment

(e)

NP-complete Problem = Any problem is NP-complete if it is a part of both NP and NP-Hard Problem.

Prove X is NP-complete :-

(i) $X \in NP$ → non-Deterministic Algorithm
↳ Certificate + Verifier

(ii) Reduce from known NP-complete problem Y to X.

Today : NP-completeness

3SAT
- → Super Mario Bros
- 3-Dimensional matching
  - → Subset Sum
    - ↳ partition
      - ↳ Rectangle Packing
  - ↳ 4-Partition
    - ↳ Rectangle Packing
    - ↳ Jigsaw Puzzles

(f)

Reduction ⇒ from problem A → problem B
= polynomial - time algorithm converting
A - inputs → equivalent B inputs

(g)

Ans = It is NP hard problem as it
is not solvable ~~problem~~ and
verifiable ~~p~~ in polynomial time

OR

NP-Hard as validated in
$T(n) = 2^n$ times

(Q2)

Proof:

$\Rightarrow$ Assume a minimum vertex-cover is U^*

$\Rightarrow$ A vertex-cover produced by Approx-Vertex-Cover(G) is U.

$\Rightarrow$ The edges choosen in Approx-Vertex-Cover(G) is A.

$\Rightarrow$ A vertex in U^* can only cover 1 edge in A, so $|U^*| >= |A|$.

$\Rightarrow$ For each edge in A, there are 2 veriables in U, so $|U^*| = 2|A|$

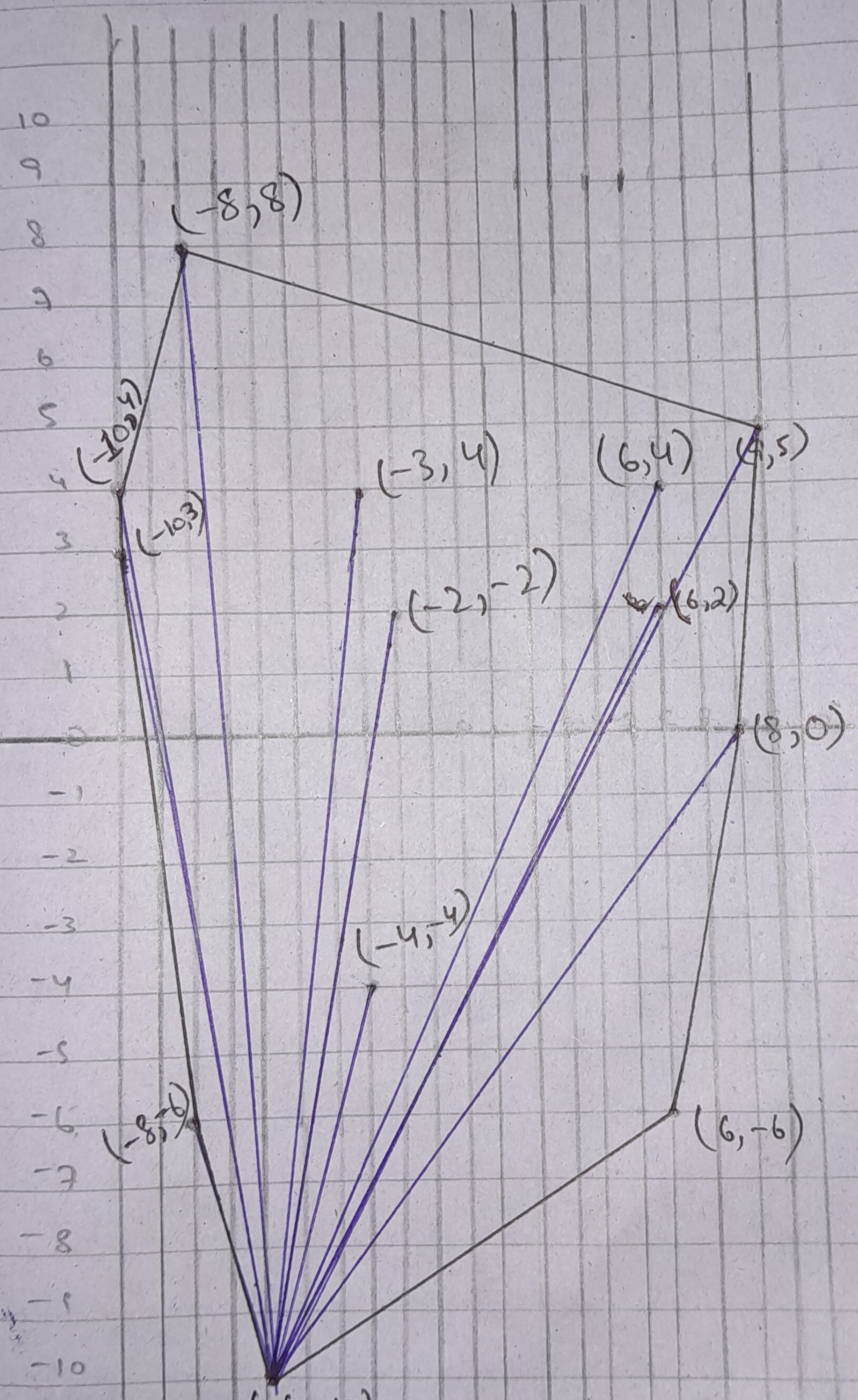
$\Rightarrow$ So $|U^*| >= |U|/2$

$\Rightarrow$ So $\frac{|U|}{|U^*|} \leq 2$

(Q3)

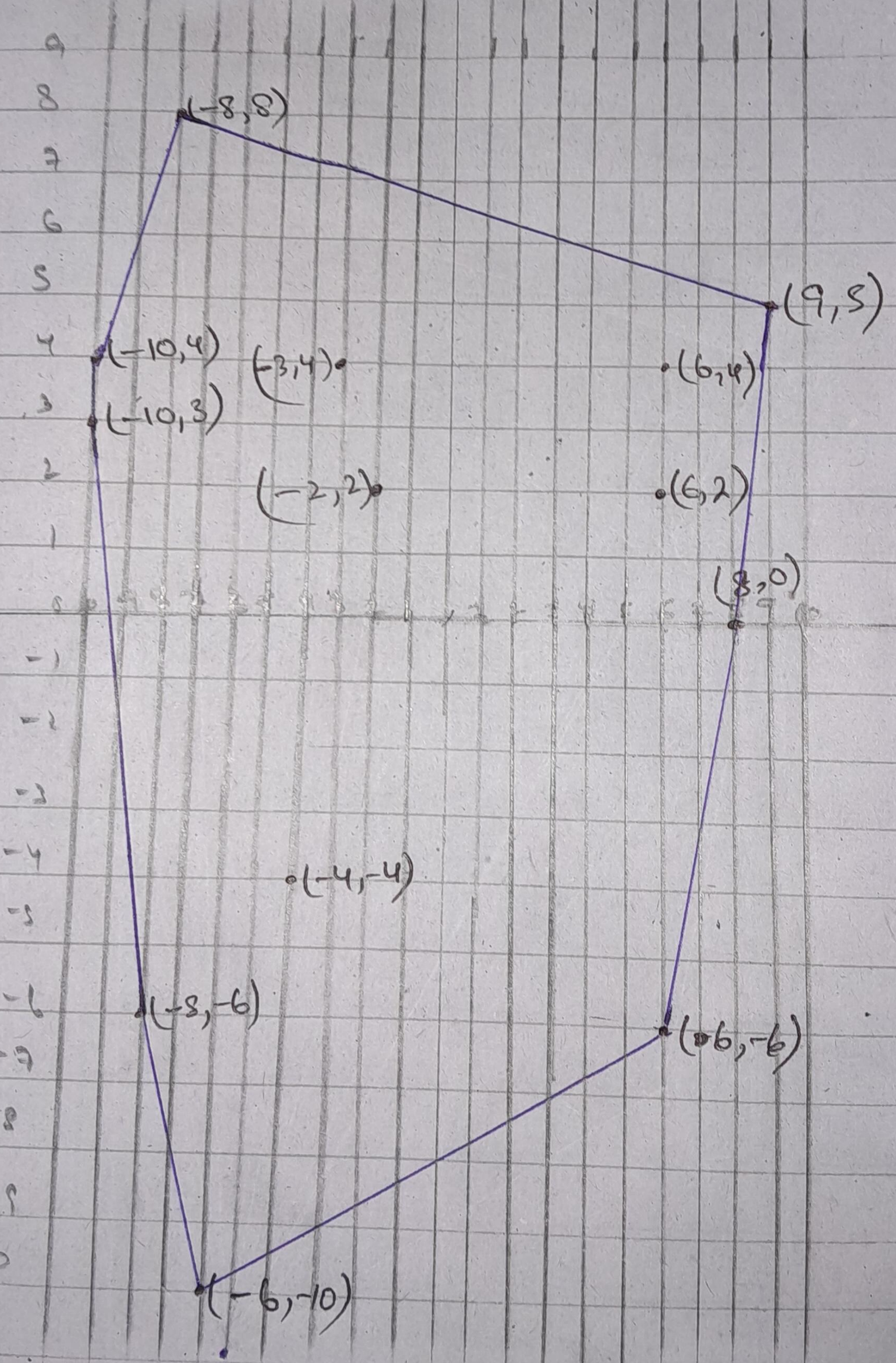
Ans = Since all the words have no repeated letters, the first word selected will be the one appears earliest on among those ~~the~~ with most letters, this is "thread". Now, we look among the words that are left, seeing how many letters aren't already covered that they contain. Since "lost" has four letters that haven't been mentioned yet, and it is first among those that do, that is the next one we select. The next one we pick is "drain" b/c it has two unmentioned letters. This only leave "shun" having any unmentioned letters, so we pick that, completing our set, so, the final set of words in our cover is {thread, lost, drain, shun}.

Q4) JARVIS MARCH



(−6,−10), (8,0), (4,5), (−8,8), (−10,4), (−10,3), (−8,−6)

By Graham Scan :-



(-8,-6)

Points included in Convex hull = {(-6,-10), (6,-6), (8,0), (9,5), (-8,8), (-10,4), (-10,3)}