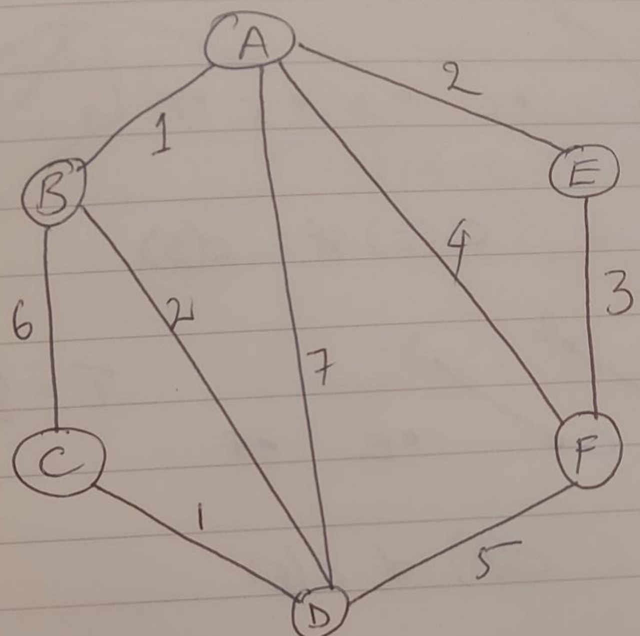
https://boazturya.github.io/cs435ALGO/BOAZ_LAB11B.pdf

Mo Tu We Th Fr Sa Su

LAB.11.B

Memo No.
Date / /

PROBLEM 1: KRUSKAL'S ALGORITHM..



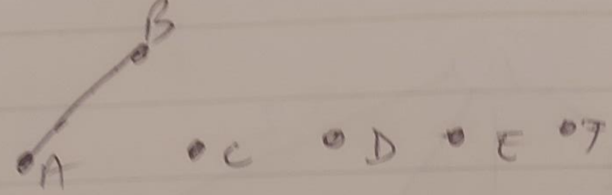
Sorted Edges = {AB, CD, AE, BD, EF, AF, DF, BC, AD}

T = {}

~~Stet~~ Step 1: Initialisations

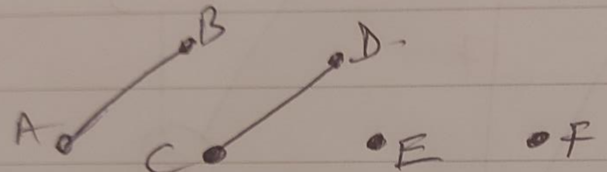
A B C D E F.

Step 2: $C[A) \neq C[B]$



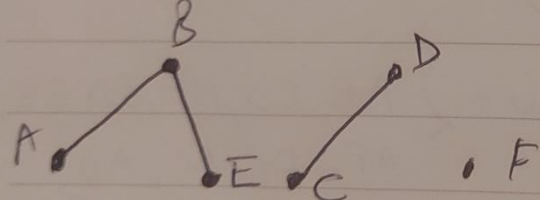
$T = \{AB\}$.

Step 3 : $C(C) \neq C(D)$



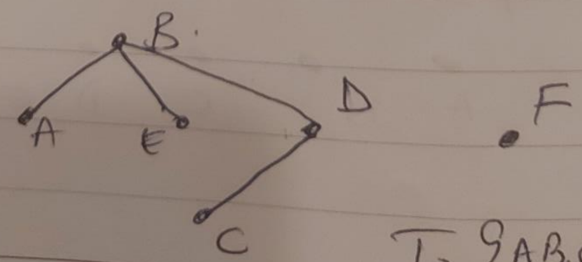
$T = \{AB, CD\}$

Step 4 : $C(A) \neq C(E)$



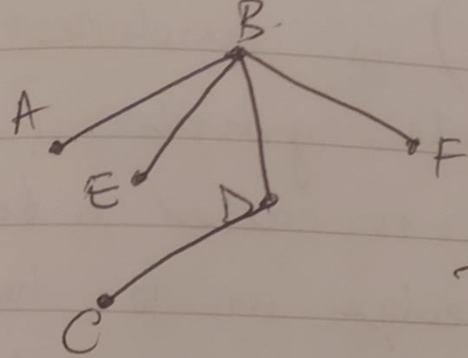
$T = \{AB, CD, AE\}$.

Step 5; $C(B) \neq C(D)$.



$T = \{AB, CD, AE, BD\}$

Step 6: $C(E) \neq C(F)$



$T = \{AB, CD, AE, BD, EF\}$

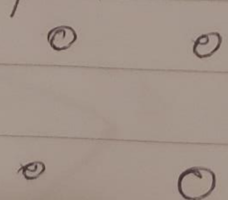
PROBLEM 2

Suppose $G=(V,E)$ is undirected (un-weighted) simple graph.
A subset U of V is called a base for G if every edge in G has at least one endpoint in U.

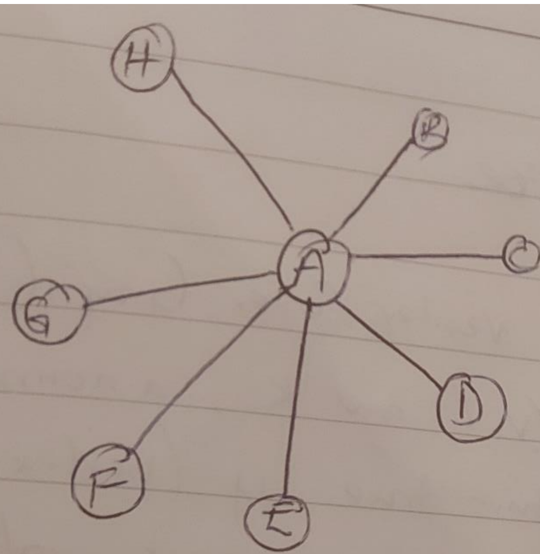
a) Given $G=(V,E)$

Yes, by definition of E, every e in E has an endpoint in V.

b) Yes any graph with one or more vertices and no edges is an example of this.



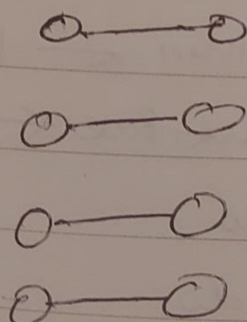
c)



If $U = \{A\}$

then every Edge in G has one end point in U.

D).



If n of the given edges are in U, then each ~~node~~ edge in G has one end point in the graph U.

~~E)~~

E) Brute Force

ALGORITHM vertex Cover Graph(G, k)

INPUT G = (V, E) and k, a nonnegative integer

OUTPUT return true if (Vertex Cover) base of U has size k, otherwise false.

$P \leftarrow$ Powerset of V

~~d ← V~~ currentMin $\leftarrow |V|$

~~SC ← Powo~~ currentBase $\leftarrow V$

for U in P do

for e in E do,

a ← left endpoint of e

b ← right endpoint of e

if a in U or b in U, then

currentMin $\leftarrow |U|$

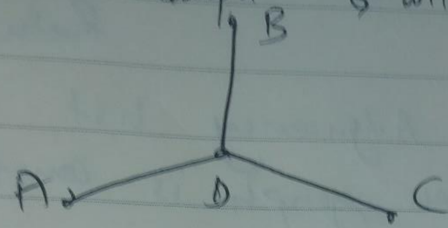
currentBase $\leftarrow U$

return U

Running time $O(m2^n)$.

PROBLEM 3

a) Imagine Graph G with $V = \{A, B, C, D\}$



The graph above has no Spaning cycle.

b) ALGORITHM spanningCycleDecision$(G(V,E))$.
INPUT Graph G whith V vertices, E edges
OUTPUT True, if G contains a spanning cycle, otherwise False.

Perform BFS starting anywhere and keep a counter for number of components by incrementint it after the single component loop if there are still more unvisited vertices in the graph.

If graph has more than one component, then it is not connected, then
Return False.

Get the Adjacency list which is formed when graph is constructed.

For Each key V in the adjacency list, get size S of it's list of adjacent vertices
if $S \neq 2$, ~~then~~
Return False.

Return true

Running time
BFS $\longrightarrow O(n+m)$.
checking adjacency List $\longrightarrow O(n)$.

Total time $O(n+m)$