LAB 10, IE 507

Ex 1

Part ① [M1, M2, M17, 61] → Red

[M5, 12, 50, 18, 21, 62] → Green

[M8, 60] → Blue

[M10, 16, 67] → Purple

[M13, 27, 68] → Pink

[M4, 37, 63] → Black

[M6] → Orange

→ No. of Colour used = 7.

Part ② ~~The~~ Python Code is also attached for performing this.

Step 1: Make diagonal entries as 0.

Step 2 Now in row of M1 routes under which 1 is present intersects with M1.

Hence, Say if possible we group those under which 1 is not written into single group and under which 1 is written into another group. say former group as L1 and latter as L2

L1 = [M1, M2, M4, M6, 16, M17, 18, 21, 27, 37, 60, 61, 62, 63, 67, 68]

L2 = [M5, M8, M10, 12, M13, 50]

Step 3) Now from the data remove rows and Columns of those routes under which 1 was not present in row 1.
So, we end up having data set / Matrix containing rows and Columns of routes in L2 only.

Step 4) As we did in step 2 again we can split L2 using 1's present in row corresponding to 1st entry of L2.
and we will get

L2 = [M5, 12, 50], L3 = [M8, M10, M13]

Step 5) Now we will L3 as we splited L2 in step 4
we will get

L3 = [M8] L4 = [M10, M13]

and again

L4 = [M10], [M13] = L5

CLASSTIME Pg. No.
Date / /

step 6 We can see that ~~L2~~ routes in L2, L3, L4, L5 have no interfere with others in that's ~~route~~ groups.

step 7 We will, one by one, take ~~them~~ routes from L1 and try to accomodate them in subsequent group (if allowed by datas of intersection).

for ex. take M2 from L1 and M4 from L1, ~~Now take L2~~ and check whether ~~if all of them do not intersect with~~ M2 & M4 intersect or not if they then we will ~~not~~ try to move M4 from L1 to others. For this take L2 and check if all of routes in L2 ~~it~~ do not intersect with M4, if they do not then put M4 in L2, else take L3 and check again. When M4 is alloted to any subsequent group, then start again with another element from L1 if M4 is not alloted to any group present then make another group of M4 only.

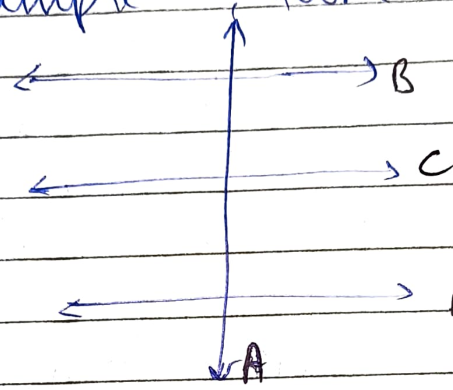
→ we can do this for all Li's in step 6 but it is not needed for L2, L3, L4 & L5 it is clear from data.

step 8 Stop when no one route in any group have intersection with others.

Part (4)

It is false.

for example take following network



here we have 4 routes, out of which route A interacts with other 3. (As per ques.)

But we can allign B, C, D same colour and A different to represent a map.

CLASSTIME Pg No.
Date / /

Ex (2)

Part (I) Say n items have weights W_i's, $i = 1$ to n
& Bin capacity $= B$.

Step (1) first arrange the items according to their decreasing order of their weights.

Step (2) Now take the left most item from list in step (1) and put that in a bin 1. cross that item from that list to represent it is fitted in a bin.

Step (3) Now again take the left most un crossed item from arranged list and try to fit that in left most bin possible i.e if it can be fitted in bin 1 then fit else fit it in Bin 2. (fitting means the sum of weights of items in a bin should not be more than B).
Again cross ~~item~~ item in arranged list.

Step (4) In general if at any stage we take a item then we should check, all those Bins which have items, from left to right. And fit that if possible.

Part (2) take items having size 9, 8, 9, 5, 4,
6, 4, 4, 10

and let Bin size as 20.

@

new array in decreasing order

~~10~~, ~~9~~, ~~9~~, ~~8~~, ~~6~~, ~~5~~, ~~4~~, ~~4~~, 4

10+9	9 + 8	6+5+4 +4	4

→ by FFD we are getting 4 as no. of bins

But clearly if we arrange in following way,

10+9	4+4+4 +8	9+6+ 5

→ no. of bins = 3.

Hence FFD is not giving optimal no. of bin for this example