problem 1 :

(a) pain person A & B
Ask whether A knows B
If answer is Yes, A is definitely not celebrity.
remove A from list
else
remove B from list.
pair the remaining person (celebrity candidate) with next person
repeat the above process until it reaches to the last person. n
For the one person left, ask everyone else if they know this guy
If they do,
this person is a celebrity
If they do not
No celebrity

(b) My Algorithm complexity is $O(n)$.

problem 2.

(a) For a unfavorable table (Not zero matrix), any move will change one digit in at least one column. Since any change to a digit in a column will change the sum of all digits in the column. And the sum will only change from even to odd for binary operation, ~~therefore~~, any move to a unfavorable table will make it favorable.

For a favorable table. find the leftmost (MSB) column that has odd # of ones. Chose one row from that column that has a digit one change that digit to 0 to make that column even. And also player has freedom to change remaining digits in that row to make remaining column even and ~~thus~~ make the table unfavorable.

Start from left-most column

if sum of digits in this column is even

continue to next column.

else if saved row index is 0

iterate from first row until it reaches to the row that it has a one

change that one to zero, update saved row index to this row index

save the new row value somewhere.

else if saved row index > 0

jump to the saved-row-index row

change the digit one in this column to zero

updated the new row value.

iterate to next column until it reaches column one.

Output saved_row_index and $(\underbrace{\text{original row value} - \text{new row value}}_{\text{Matches removed}})$

(b). Yes, I can tell whether there exist multiple ways to make the table unfavorable by counting the number of digits 1 in the left-most column that has odd sum. The # of digits 1 in that column represent the # of ways making the table unfavorable.

(c) The winning strategy of this game is simple: alway leave a unfavorable table to your opponent.

problem 3.

Let $n_1, n_2 \cdots n_{2k}$ be the set of all odd-degree Nodes. Pair these nodes in arbitrary fashion. For pair $i, = 0, \cdots k-1$, we add a new node W_i and two edges that create a new path connecting both vertices. For this new graph G', since every nodes has even degree, there exist an Eulerian circuit. Therefore there is a walk visits new node W_i exactly once via new edges incident with them.

This walk looks like:

$$V_1, N_1, V_2, W_1, V_3, N_2, V_4, W_2 \cdots$$

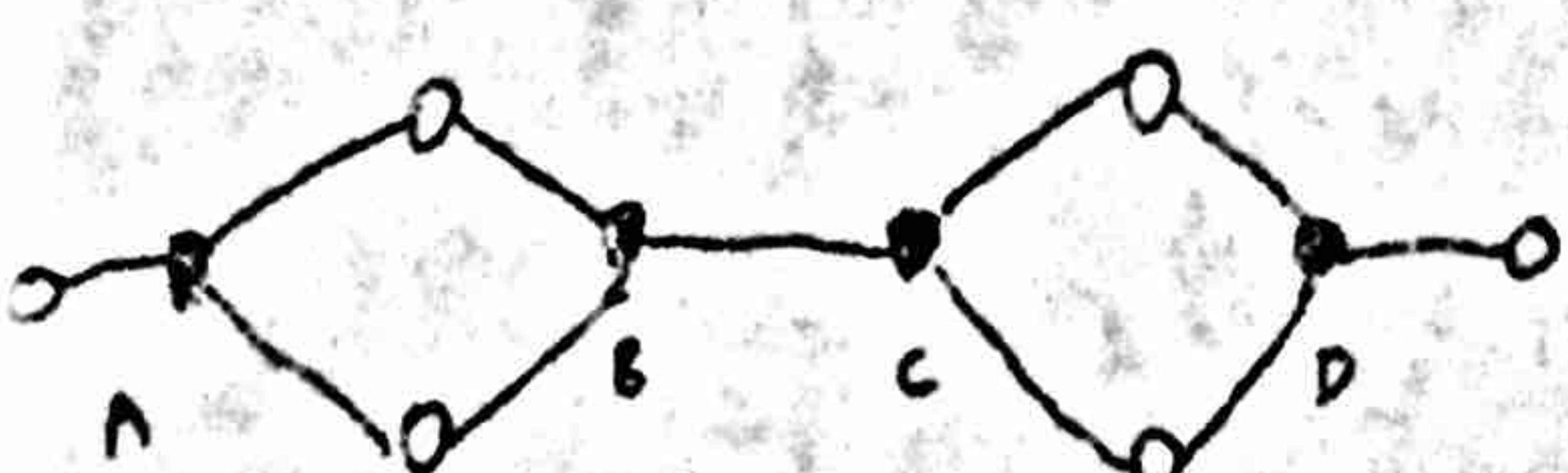
(V_i - new edges)
(N_i - new nodes)

Remove all new edges and nodes in G' and leave W_i in original graph, we found K disjoint walks.

$$W_1, W_2 \cdots W_k.$$

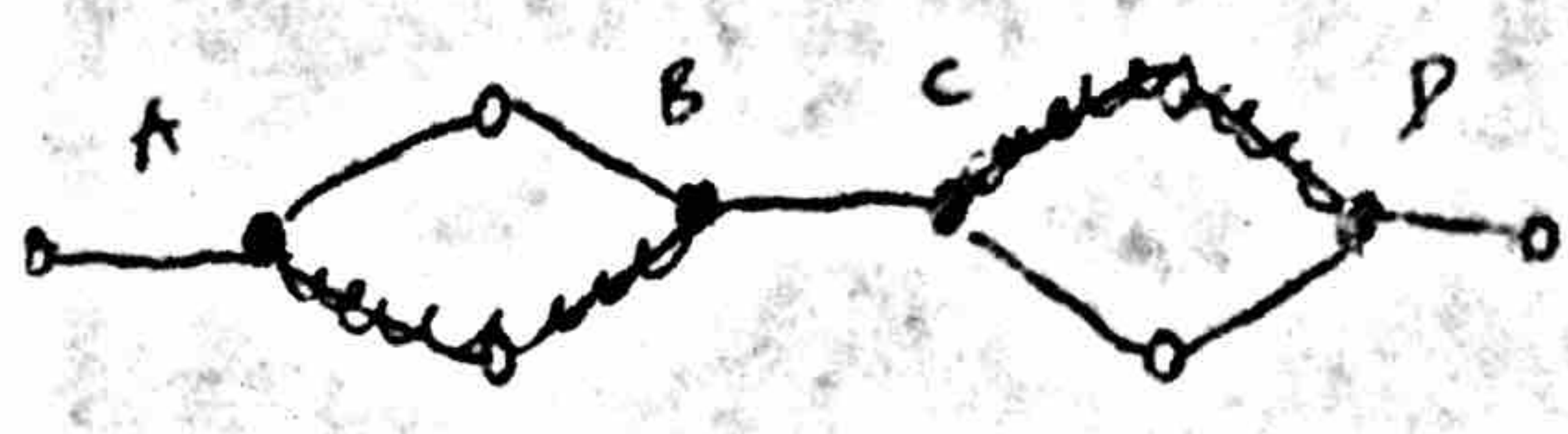
Found Each pair of odd nodes W_i associate with. And we find out set of nodes that create edge-disjoint paths.

E.g.



create [A, C], [B, D]

Add new vertex and edges and find eulirean circuits.



remove new vertex and edges, leave paths

Found new pair:

[A, B], [C, D].

problem 4:

(a). DAG is a finite directed graph with no directed circles. That is, it consists of vertices and edges with each edge directed from one to another such that there is no way to start at any vertices V and loops back to V again.

If every vertices in a graph has outgoing edges for finite vertices, the graph will be no longer acyclic as now it is possible to start from a vertice and loop back to it.

Therefore, for DAG, there must be one sink node to ensure it being acyclic.

(b). Any case is worst case for this strategy.
This strategy will always result in a infinite loop and will never terminate! As (a) proves, there always exist a sink node in a DAG. For example, in the following example, no matter how you reverse edges, sink node will not be terminated. Therefore, it will never terminate.