CSC 225 - SUMMER 2019
ALGORITHMS AND DATA STRUCTURES I
WRITTEN ASSIGNMENT 4
UNIVERSITY OF VICTORIA

Due: Friday, July 26th, 2019 before noon. **Late assignments will not be accepted.**

Submit your answers on paper to the CSC 225 drop box on the second floor of ECS (in front of the elevators). You are expected to submit a typed solution (handwritten documents will not be marked). However, you are permitted to draw mathematical formulas or diagrams onto the typed copy by hand if that is more convenient than typesetting them.

Question 1: Free Trees [5 marks]
Recall that a **tree** is a connected acyclic graph. Earlier in the course we focused on **rooted trees** (such as binary trees). This question concerns **free trees**, which have no root node (and no notion of parents or children either). Prove by induction that, for all $n \geq 1$, any free tree on n vertices will have exactly $m = n - 1$ edges.

Question 2: Directed Graphs [5 marks]
In a directed graph, a vertex v is called a **minimum** vertex if v has no incoming edges. Similarly, a **maximum** vertex is any vertex with no outgoing edges. Note that there may be multiple minimum vertices (or none at all).

Prove that a directed graph with $n \geq 2$ vertices and

$$m = \frac{n(n-1)}{2} + 1$$

edges must contain a directed cycle. You may use the following theorem without proof (but you are not required to).

Theorem A: A directed graph with no **minimum** vertex must contain a cycle.

Hint: Use induction. In the induction step, consider a graph on $n + 1$ vertices and use two cases: one for graphs with a minimum vertex and one for graphs with no minimum vertex.

Question 3: Bipartite Graphs [5 marks]
A graph is bipartite if and only if it does not contain a cycle of odd length.

Design an algorithm TESTBIPARTITE to solve the following problem.

Input: A graph G with n vertices and m edges, provided as an adjacency list, where the vertices are numbered $v_1, v_2, \ldots, v_n$ and the set of neighbours of a vertex v_i is denoted NEIGHBOURS(v_i).

Output: A boolean value: `true` if G is bipartite and `false` otherwise.

Justify the correctness of your solution. For full marks, your algorithm must run in $O(n+m)$ time.

Question 4: Counting Problems [6 marks]

For each of the questions below, your answer must consist of a formula in terms of the number of vertices n, along with a brief but concise justification of its correctness. Answers without sufficient justification will receive no marks.

(a) We know that the maximum number of possible edges in a graph on n vertices is

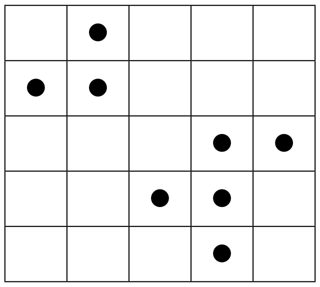
$$\binom{n}{2} = \frac{n(n-1)}{2}.$$

Derive a formula which counts the number of possible graphs on n vertices (with no restrictions whatsoever on the structure of the graph, besides that it has n vertices and is an undirected simple graph).

(b) Suppose G is a graph on $n \geq 4$ vertices with exactly four connected components. Derive a formula for the **maximum** number of edges in G.

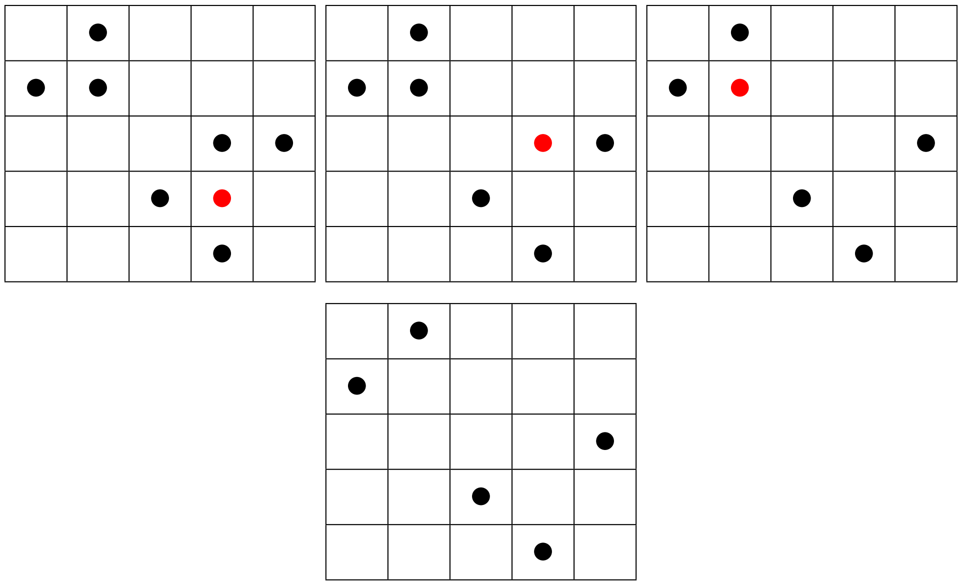
(c) Suppose G is a directed acyclic graph with exactly k minimum vertices. Derive a formula (in terms of n and k) for the **minimum** number of directed edges in G.

Question 5: Marbles [5 marks]
This question requires solving an optimization problem based on a simple game. Consider the 5×5 grid below, where some grid cells contain a marble.

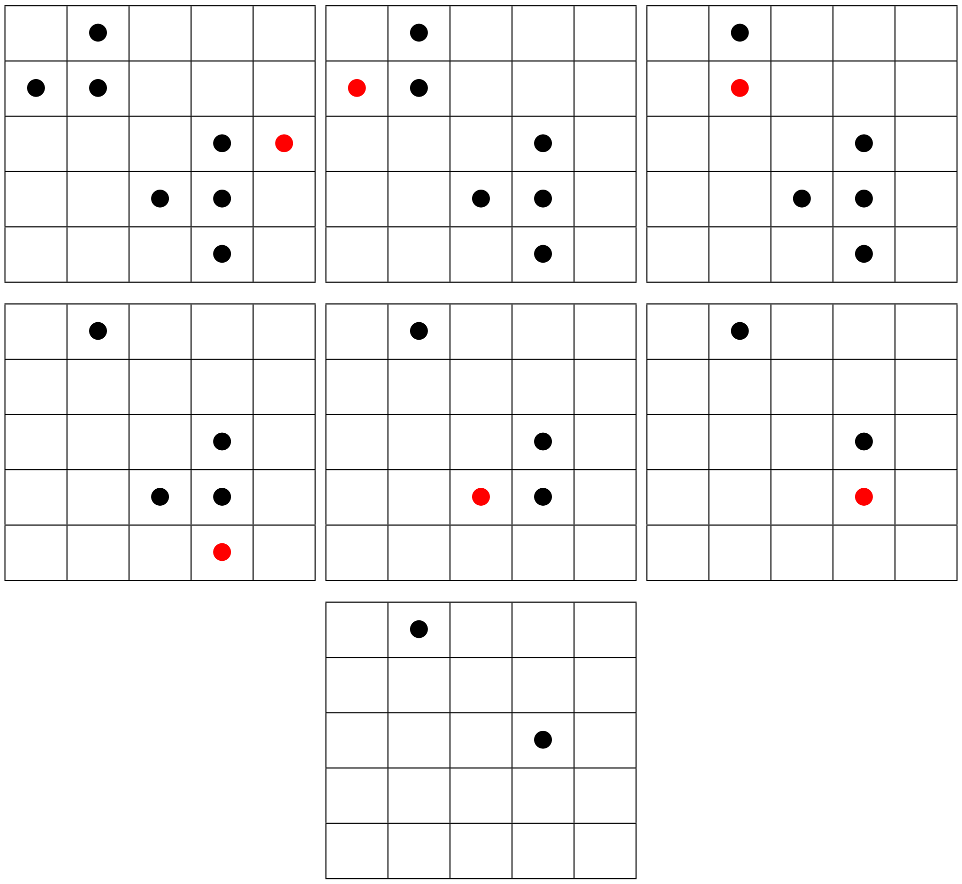


Starting from an initial grid (which can be represented by a 2d array of boolean values in a program), the goal of the game is to remove as many marbles as possible, until no further removals are possible. A marble can be removed from the grid if and only if there is at least one other marble in the same row or column of the marble to be removed.

The order in which marbles are removed can affect how many marbles can be removed in total. For example, starting from the grid above, the following sequence of moves can remove 3 marbles (where the marble to be removed is highlighted in red at each step).



However, with the sequence of moves below, it is possible to remove six marbles.



Design an algorithm Marbles to solve the following problem.

Input:	An integer n along with an $n \times n$ array A of boolean values, where $A[i][j] =$ `true` if there is a marble in row i, column j.
Output:	The maximum number k of marbles that can be removed from the grid under the rules above.

Notice that your algorithm **does not** have to find the sequence of marbles to remove, just report the maximum number of marbles that can be removed (which should make the problem easier). It is sufficient to provide a high level description of the algorithm instead of pseudocode (e.g. describe how a graph could be created and which algorithms to use on the graph, instead of explicitly giving code for those steps). You must include a justification of the correctness of your algorithm. For full marks, your algorithm must run in $O(n^2)$ time in the worst case.