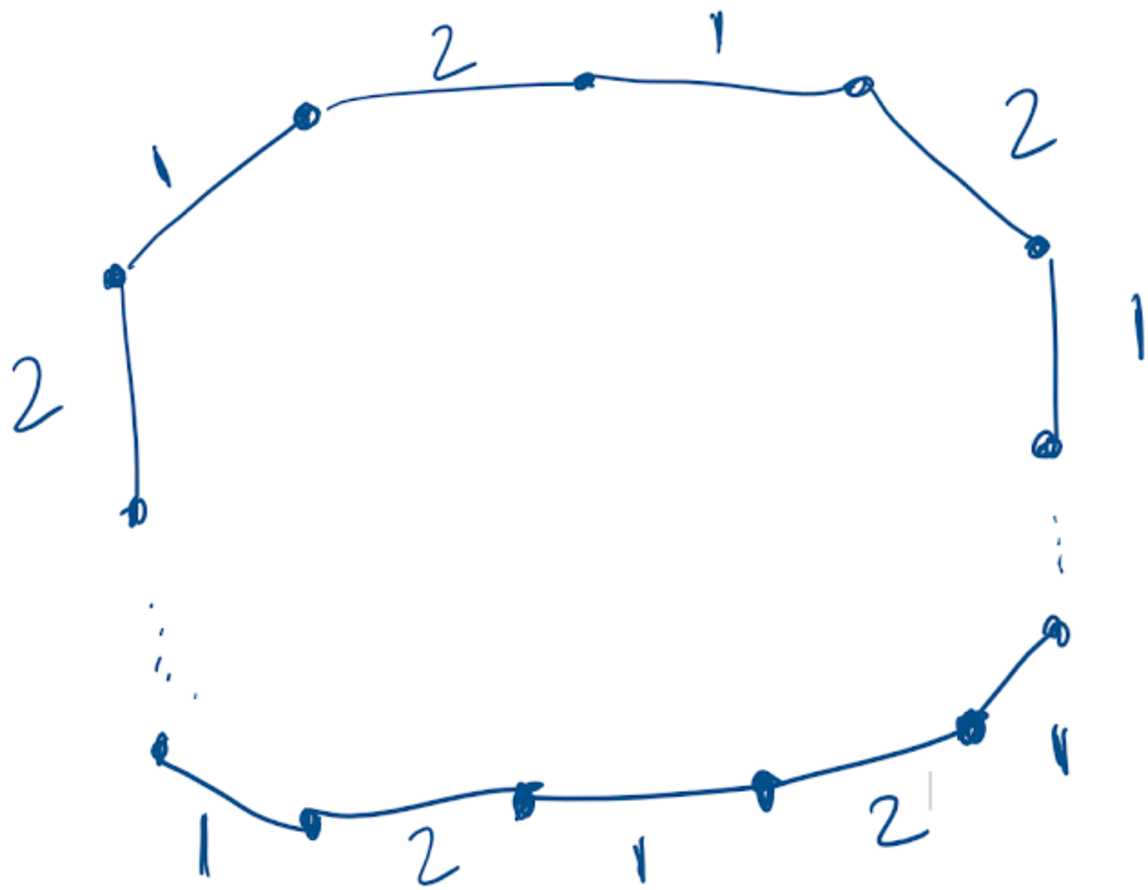
Consider a cycle on 100 vertices, where the edges have weights 1 and 2 in an alternating fashion (as shown in the figure below). How many distinct minimum spanning trees does this graph have? Provide the answer in numeric form.



50

Let P be a path graph on n vertices. That is, it contains n vertices v1, v2, v3, ... vn such that there is an edge from v1 to v2, v2 to v3, v3 to v4 and so on till v(n-1) to vn. Suppose we perform BFS on P starting at an arbitrary vertex. What is the maximum number of vertices that can be present in the queue while BFS is being executed?

- ○ 1
- ◉ 2
- ○ 3
- ○ Depends on n

Which of the following best describes the time complexity of BFS if the graph is provided as an adjacency matrix?

- ○ O(|V|)
- ○ O(|V| + |E|)
- ◉ O(|V|^2) (O of |V| squared)
- ○ O(|E|^2) (O of |E| squared)

There is a graph on n vertices and 10n edges, for some large integer n. Which of the following is the most accurate statement? (in the following n^2 stands for "n squared")

- ( ) The adjacency list and adjacency matrix uses O(n) space each
- (•) The adjacency list uses O(n) space, but the adjacency matrix needs O(n^2) space
- ( ) The adjacency matrix uses O(n) space, but the adjacency list needs O(n^2) space
- ( ) Both the adjacency list and adjacency matrix needs O(n^2) space

Let G be a tree (connected and acyclic graph). Suppose each edge of G has a nonnegative weight assigned to it. Which of the following is the best upper bound for the time complexity of Dijkstra's algorithm if we implement it using adjacency list and binary heap (for the priority queue)?

- ( ) O(|V|)
- ( ) O(|V|^2)
- (•) O(|V| log |V|)
- ( ) O(|V|^2 log |V|)

Assume you have a min-priority queue with running times t1 for Extract-Min and t2 for Decrease-Key. Using such a priority queue, the worst case running time of the Dijkstra's algorithm on an adjacency list of a graph with n vertices and m edges is:

- (selected) O(n t1 + m t2)
- O(n t2 + m t1)
- O(n (t1 + t2))
- O(m (t1 + t2))

When implementing a Graph on n vertices and m edges as an adjacency list, which among the following, is the tightest worst case running time of checking if an edge exists?

- (selected) O(n)
- O(m)
- O(n+m)
- O(1)

Consider the linked list implementation of Disjoint Set with the union by rank heuristic. Assume a node N holds an element x that is inside a set of size s. If a union operation changed the metadata pointer of N , then which of the following could be the size of the set that contains x after the union?

- [ ] s/2
- [ ] s
- [x] 2s
- [x] 4s