Question 1: I have the following directed graph:

- For every positive integer i, there is a node n_i.
- Yes, that means that there is an infinite number of nodes.
- There is a special node t.
- For every node n_i, there is an edge to n_{i+1}.
- If i is a prime number, then there is an edge from n_i to n_{5i}.
- If i is a prime number > 10000, then there is an edge from n_i to t.
- All edge weights are 1.

True or false -- this graph is acyclic.

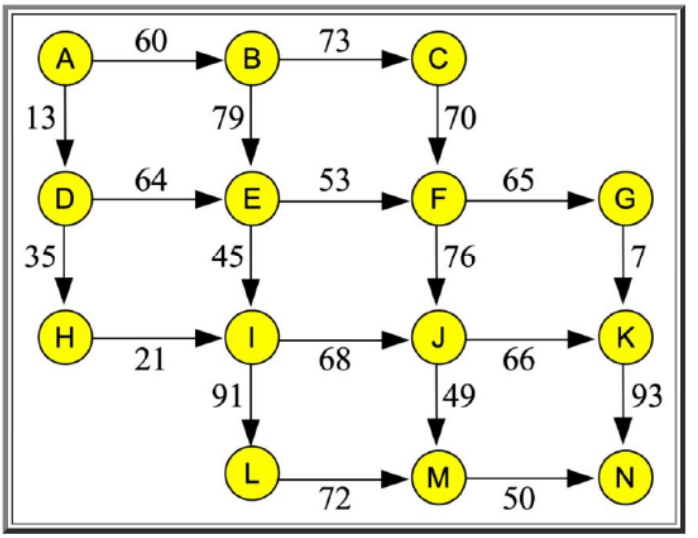
Question 2: In the graph above, suppose I want to find the path from 1 to t with the smallest overall weight of edges. Which algorithm will be the best? Just answer with the letter.

- **A.** Depth-first search.
- **B.** Breadth-first search
- **C.** Dijkstra's algorithm
- **D.** Network flow
- **E.** Topological sort
- **F.** It's impossible.

Question 3: True or false: Since the graph is of infinite size, I am not guaranteed to find the shortest path in a finite amount of time.

Question 4: In the graph above, suppose that the weights of the edges are random numbers, uniformly distributed from 1 to 50. Now, which is the best algorithm for finding the path from 1 to t with the smallest overall weight of edges? Use the same answers as Question 2.

Question 5: In the graph below, what is the flow of the path with maximum flow from A to N? Hint: Use topological sort (you may want to write in flow values for nodes B through N in that order).



Clicker Question Answers

Question 1: True. All edges either go to t, or they go from n_i to n_j, where $j > i$. Therefore, there's no way to have a cycle.

Question 2: B. Breadth-first search. The only contenders here are BFS, Dijkstra and Topological Sort. Unforunately, topological sort will never process t, because there is an infinite number of edges to t. The only way it can process t is if it processes all of the nodes with edges to t, so that will never happen.

BFS, on the other hand, keeps a queue of nodes with known shortest paths to t. The shortest path from 1 to t won't be very large -- it will keep going along edges from n_i to n_{5i}, when i is prime, and from n_i to n_{i+1}, when i is not prime. Eventually, it will get to n_i, where i is prime and > 10000, and it will go directly to t. So it will end.

Put another way, although there is an infinite number of paths from 1 to other nodes in the graph, there is a finite number of paths whose length is the minimum length. For that reason, the BFS will finish.

BFS is better than Dijkstra, because all of the edge weights are 1. The queue in BFS will perform better than the multimap of Dijkstra's algorithm.

Question 3: False. See above -- the algorithm will terminate.

Question 4: Since the edges have weights ≠ 1, we need to use Dijstra's algorithm, and it will terminate, again because there are a finite number of paths with the shortest weight.

Question 5: The answer is 60.

A valid Topological Sort is ABCDEFGHIJKLMN. So, let's just fill in the best flows to these nodes in order:

```
B - 60
C - 60
D - 13
E - 60 - max(13,60)
F - 60 - max(53,60)
G - 60
H - 13
I - 45 - max(13,45)
J - 60 - max(45,60)
K - 60 - max(60,7)
L - 45
M - 49 - max(45,49)
N - 60 - max(49,60)
```