

Write your answers in the separate answer booklet.

You have 120 minutes (after you get the answer booklet) to answer five questions.
Please return this question sheet and your cheat sheet with your answers.

1. (a) Solve the following recurrences:

- $X(n) = 9X(n/3) + O(n^2)$
- $Y(n) = 4Y(n/3) + O(n)$
- $Z(n) = Z(n/3) + 3Z(n/6) + O(n)$

- (b) After making it most of the way through solving a complicated dynamic programming problem, you've come up with the following recurrence. Here, $A[1..n]$ is an array holding n arbitrary numbers.

$$BestBobble(i, k) = \begin{cases} 0 & \text{if } i \geq k \\ \min \left\{ \begin{array}{l} BestBobble(i, j-1) \\ + i \cdot j \cdot A[j] \\ + BestBobble(j, k) \end{array} \middle| i < j \leq k \right\} & \text{otherwise} \end{cases}$$

The solution to the problem can be obtained by evaluating $BestBobble(1, n)$. Describe an appropriate memoization structure and evaluation order for computing that function value, and state the running time of the resulting iterative algorithm.

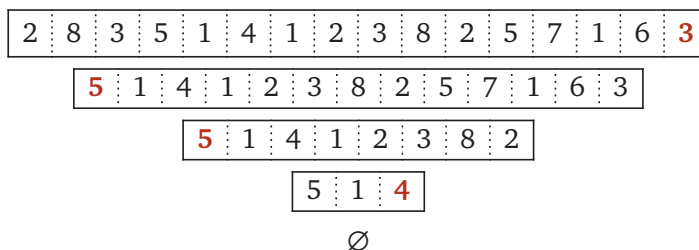
2. Suppose you are given a sorted array of n distinct numbers that has been rotated *forward* k steps for some **unknown** integer k between 1 and $n-1$. In other words, you are given an array $A[1..n]$ such that some prefix $A[1..k]$ is sorted in increasing order, the complementary suffix $A[k+1..n]$ is sorted in increasing order, and $A[n] < A[1]$.

Describe an algorithm to determine the unknown number k . For example, given the following 16-element array, your algorithm should return 10.

9	13	16	18	19	23	28	31	37	42		1	3	4	5	7	8
---	----	----	----	----	----	----	----	----	----	--	---	---	---	---	---	---

3. Consider the following solitaire (single player) game. The player is given a list of n positive integers (not necessarily distinct) ordered left to right. Integers are removed from either the left or right side of the list over a sequence of turns. Specifically, suppose the leftmost and rightmost remaining integers are x and y , respectively. The player has the choice to either remove the *rightmost* x integers from the list or to remove the *leftmost* y integers from the list. If the number of integers to be removed is larger than the number of integers remaining, the game ends.

The example below shows an instance of the game being completed after 4 turns.



Describe an algorithm that given a list of positive integers, computes the minimum number of turns that can be taken before completing the game. You may assume the input to your algorithm is an array $A[1..n]$ where $A[i]$ is the i th integer from the left in the list. Given the example above, your algorithm should return a value that is *at most* 4.

4. You're visiting your cousin in a new city with two overlapping tram systems XTremeTram and YourTram (or Systems X and Y for short) that share the same stations but offer different connections between them. Both systems charge money to use their connections but offer a discount to card-carrying members of that system.

Your cousin had offered to hook you up with, "a membership card for both tram systems," but when you first arrived at her home, she revealed that she really had a single blank card along with two card-sized stickers, one designed to make it look like a card for System X, and the other designed to make it look like a card for System Y. The effects of this scheme can be summarized as follows:

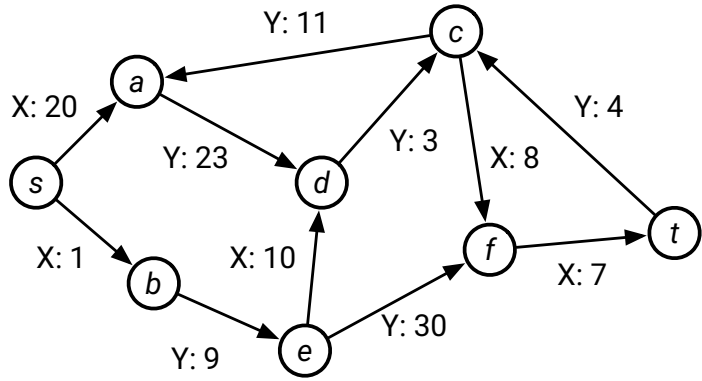
- You need to apply a sticker before you appear to be a member of either system.
- Once you apply the second sticker, you no longer appear to be a member of the first sticker's system.
- You'll pay **half** the cost of using a connection run by the system you currently appear to be a member of.

The stickers were produced with extremely good adhesive, so neither of them can be removed once they are applied. In particular, the second sticker you apply **permanently** covers up the first. However, nobody at the tram stations will be watching you very closely, so you can apply either sticker at your friend's house and the second sticker at any other stop. **Your goal is to get from your friend's house to the Transit Authority as cheaply as possible**, so you can buy yourself some *proper* membership cards.

Formally, you are given a directed graph $G = (V, E)$ whose n vertices represent tram stations and whose m edges represent direct connections between pairs of stations. Every edge e is given a label $system(e) \in \{X, Y\}$ saying whether that connection is run by System X or System Y, respectively, along with a positive cost $\$(e)$ for **non-members** to use that connection (again, you'll pay half that cost if you appear to be a member of the connection's system.)

Describe an algorithm to compute the minimum cost to get from your friend's house at vertex s to the Transit Authority at vertex t . For example, given the graph

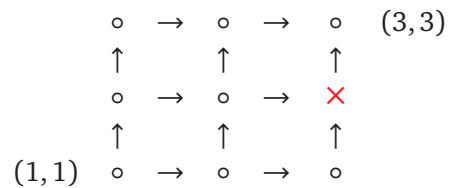
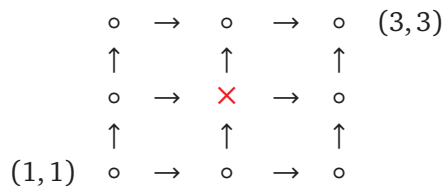
below with each edge labeled by its system and cost, your algorithm should return 21 for the plan of applying the System Y sticker, traveling along $s \rightarrow b \rightarrow e$ while appearing to be a member of System Y at cost $1 + 9/2 = 5.5$, applying the System X sticker, and traveling along $e \rightarrow d \rightarrow c \rightarrow f \rightarrow t$ while appearing to be a member of System X at cost $10/2 + 3 + 8/2 + 7/2 = 15.5$.



- Suppose we are given an $n \times n$ grid containing the points $\{(i, j) \mid 1 \leq i \leq n, 1 \leq j \leq n\}$, where some grid nodes are marked as dangerous. $DANGEROUS(i, j)$ returns TRUE if grid point (i, j) is dangerous, and it returns FALSE otherwise.

From every grid point (i, j) , there is an edge to the right and above the grid point, namely to $(i + 1, j)$ and $(i, j + 1)$. A path is called *safe* if it contains no dangerous nodes.

In the following 3×3 examples, \times indicates dangerous grid points. In the left example with point $(2, 2)$ being dangerous, the number of safe paths from $(1, 1)$ to $(3, 3)$ is 2. While in the right example with $(3, 2)$ point being dangerous, the number of safe paths is 3. For each of the examples, its paths are listed under the example.



- $(1, 1) \rightarrow (2, 1) \rightarrow (3, 1) \rightarrow (3, 2) \rightarrow (3, 3)$
- $(1, 1) \rightarrow (1, 2) \rightarrow (1, 3) \rightarrow (2, 3) \rightarrow (3, 3)$

- $(1, 1) \rightarrow (2, 1) \rightarrow (3, 1) \rightarrow (3, 2) \rightarrow (3, 3)$
- $(1, 1) \rightarrow (1, 2) \rightarrow (2, 2) \rightarrow (2, 3) \rightarrow (3, 3)$
- $(1, 1) \rightarrow (2, 1) \rightarrow (2, 2) \rightarrow (2, 3) \rightarrow (3, 3)$

Describe an algorithm to find the number of safe paths from node $(1, 1)$ to node (n, n) . Given the left and right examples depicted above as input, your algorithm should return 2 and 3, respectively.