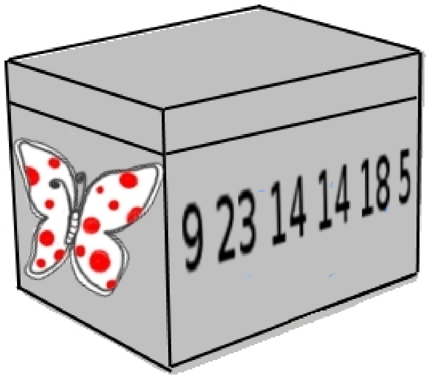
Magic Word

Question:

A magic word is needed to open a box. A secret code assigns each letter of the alphabet to a unique number. The code for the magic word is written on the outside of the box.



What is the magic word?

Possible Answers:

LOOSER
WINNER
LOTTOS
TICKET

Correct Answer:

WINNER

Explanation of Correct Answer:

The third and fourth letters of the word must be the same, since they have the same code value: thus TICKET and LOOSER cannot be correct.
The second and fifth letters must be different, since the code values are different: thus LOTTOS cannot be correct.
WINNER satisfies the mapping $W = 9$, $I = 23$, $N = 14$, $E = 18$, $R = 5$.

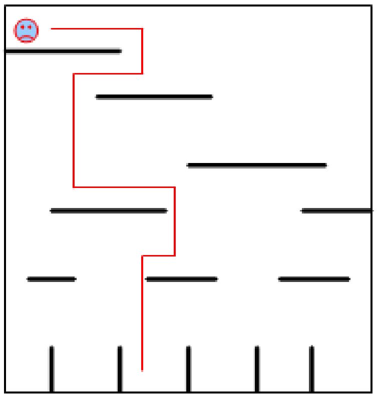
Connections to Computer Science:

The information that computers receive, analyze and transmit is stored using codes. The most common code for storing characters is ASCII: American Standard Code for Information Interchange. ASCII assigns a certain number to (almost) each possible character that can be typed on keyboards. For example, the character `A` has ASCII value 65, the character `a` has ASCII value 97, and the character `#` has value 35. In total, there are 128 different standard ASCII values.
Of course, there are lots of other non-English characters that also need to be transmitted such as Π (Greek uppercase letter "pi") and И (Cyrillic upper-case letter I). A newer encoding, called UTF-8, encodes virtually any character from any written language. For example, in UTF-8 encoding, Π has value 928, and И has value 1048. It is also worth noting that all the ASCII values have the same value in UTF-8 as they do in ASCII: for example, the character `A` has UTF-8 value 97.
With these standards, information can be stored and displayed consistently across all computer systems.

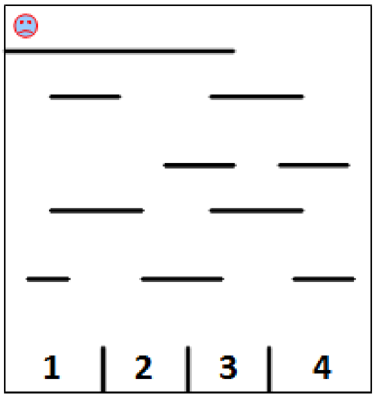
Falling Ball

Question:

A ball is placed at the top left corner of a maze. It falls down the maze from platform to platform until it reaches one of the slots at the bottom.
The ball always moves in the same way. It starts by going to the right. Every time it falls from one platform to another, it changes direction. Here is an example:



Which slot will the ball reach in the maze below?



Possible Answers

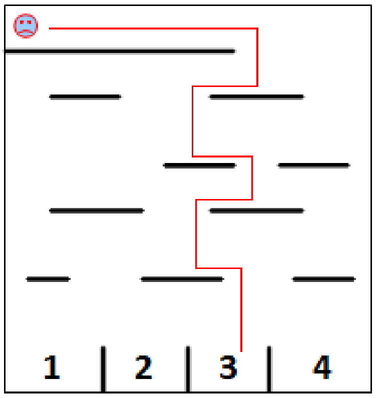
1
2
3
4

Correct Answer

3

Explanation of Correct Answer

Following the sequence of drops and “change direction”, the ball will take the path illustrated below to end up at position 3.



Connections to Computer Science:

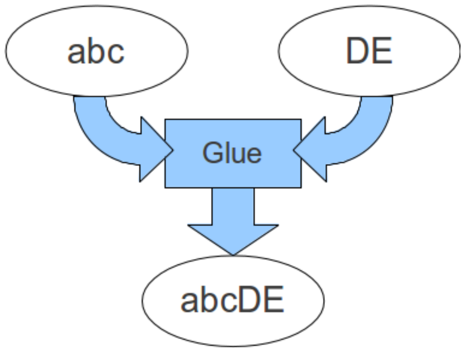
Following a procedure (as in “move until you drop, then change direction and repeat”) is a fundamental concept in computer science called an *algorithm*. You know lots of algorithms, even if you don’t know how to program: how to tie your shoes; how to brush your teeth; how to make a sandwich. All of these sequences of steps have similar properties:

- there is an order to the sequence of steps;
- the number of steps is finite (although the steps themselves may be repeated a finite number of times);
- the sequence of steps accomplishes some desired outcome.

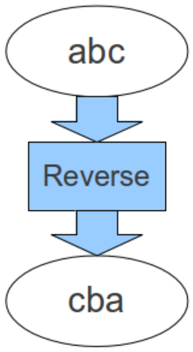
In general, all algorithms satisfy the above three points. Following and understanding algorithms is a core concept in computer science.

Text Machine

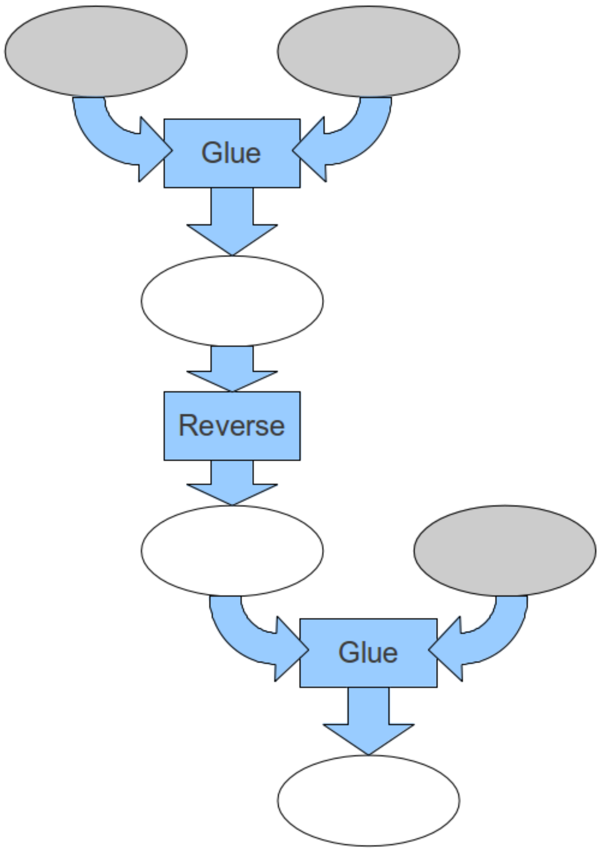
Question:
A *Glue* machine takes two pieces of text and puts one after the other. An example is shown below:



A *Reverse* machine takes one piece of text and puts the characters in reverse order. An example is shown below:



Two Glue machines and one Reverse machine are combined to create the *Combined* machine shown below. The *Combined* machine takes three pieces of text (in the grey ovals) and after processing them, gives one piece of final text (in the bottom-most oval).



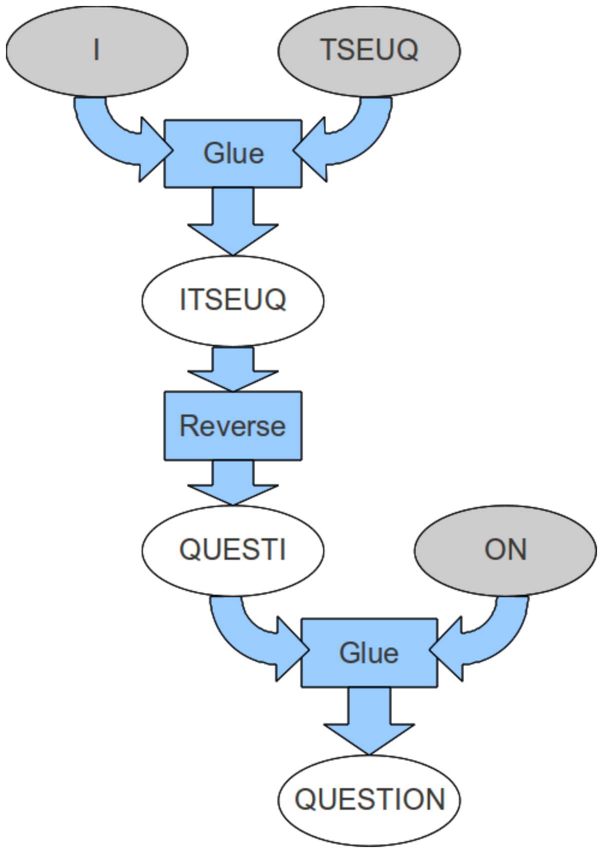
Which three pieces of text will produce the final text QUESTION when given to the Combined machine, in the order specified?

Possible Answers:

EUQ TS NOI
TSE UQ NOI
I TSEUQ ON
QU EST ION

Correct Answer:
I TSEUQ ON

Explanation of Correct Answer
We can try all possible inputs, but the only one which works is shown below:



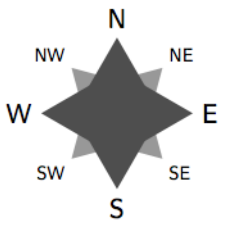
This can be reached by trial-and-error, or some elementary reasoning starting from either the top or bottom of the combined machine. For example, looking at the last Glue operation, the first two answers can be eliminated, since the ending of QUESTION cannot be NOI, TS, EQU, UQ or TSE.

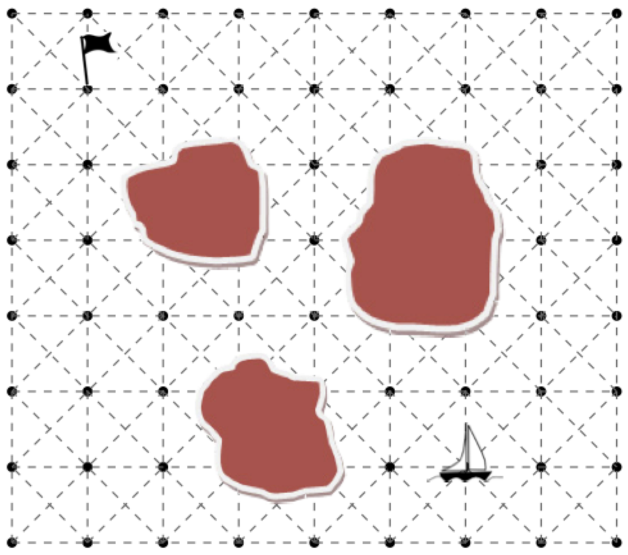
Connections to Computer Science:
Many computer processes can be thought of as "black-boxes": we know what the output is based on the given input, but we do not know *how* the "black-box" actually produces the output. This concept in computer science is called *abstraction* and it is a useful way of analyzing and understanding algorithms or processes: we focus on the only details of the problem that affect us, and ignore the other non-important aspects.
For this problem, we treat the Glue and Reverse boxes as abstractions, and then focus on their output on various inputs.

Beaver Navigation

Question:

A sailor takes her boat on the lake with islands shown below. Her aim is to sail to the flag by programming the boat's autopilot. The autopilot commands the boat to sail along the dashed lines. In one step, the boat moves from a point to the nearest point along a dashed line in one of eight different directions. For example, the command *1 N* means take 1 step in the northern direction, and the command *2 NE* means take 2 steps diagonally in the northeastern direction.
Each point is a small black circle on the map. The 8 different directions (N, NE, E, SE, S, SW, W, NW) are also shown below.





Which of the following routes to the flag avoids the islands using the smallest number of steps?

Possible Answers:

5 NW
2 NW, 2 W, 1 N, 1W, 2N
2 NW, 3 N, 3 W
2 NW, 2 W, 1 NW, 2 N

Correct Answer:
2 NW, 2 W, 1 NW, 2 N

Explanation of Correct Answer
The correct answer leads the boat to the flag safely and has 7 steps. Choices 5NW and (2NW, 2W, 1N, 1W, 2N) lead the boat to collision. Route (2NW, 3N, 3W) has 8 steps.

Connections to Computer Science:
Similar to the "Falling Ball" problem, this problem involves following an algorithm. However, to add a bit of complexity to this problem, it is not just a matter of following the movements of the boat, but also of doing some *analysis* in comparing the possible solutions.
Without any constraints, all of the possible answers would reach the destination. However, we care about solutions which satisfy two other criteria:

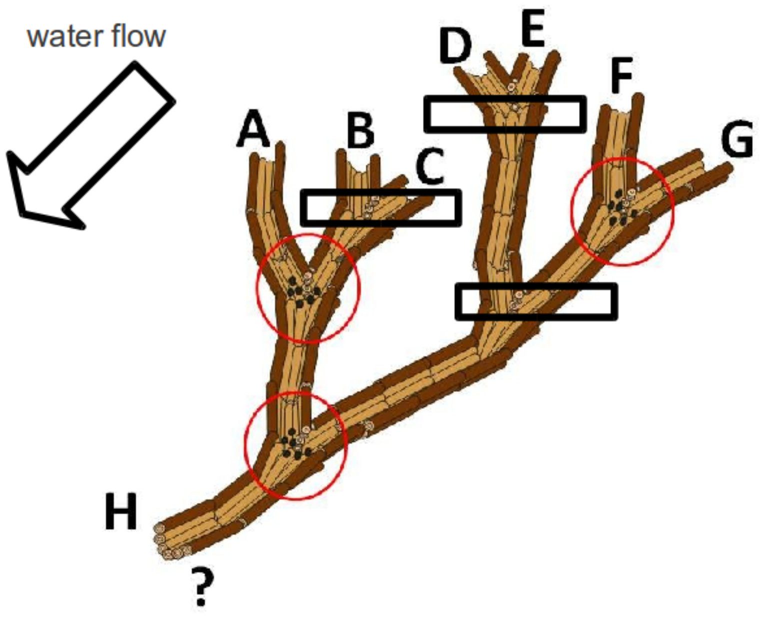
- no collisions with islands;
- fewest possible number of steps.

This sort of analysis highlights that quite often in computer science, we not only care about finding a solution, but finding a solution which satisfies other criteria. This kind of study is sometimes called *algorithmic analysis*, which usually covers items such as efficiency, constraint satisfaction, optimality, etc.

Power Generation

Question:

Beavers have built a system of wooden pipes (shown below) to supply a power plant at H. Water enters from some of the locations A to G and flows downward toward H.



There are two types of intersections where pipes meet:

1. **Those which allow water to flow without restriction** (highlighted with black rectangles): water flows through the intersection as long as at least one of the two incoming pipes contains water;
2. **Those which restrict water flow** (highlighted with red circles): water flows through the intersection only if both incoming pipes contain water.

For which of the following situations will water reach the power plant at H?

Possible Answers:

Water at A, B, C, F and no water at D, E, G
Water at A, B, G and no water at C, D, E, F
Water at A, C, D and no water at B, E, F, G
Water at B, C, E, G and no water at A, D, F

Correct Answer:
Water at A, C, D and no water at B, E, F, G

Explanation of Correct Answer
We will consider the possible answers in turn.
For the first possible answer, if there is no water at G, then there is no water flowing out of the intersection of F and G. If, in addition, there is no water flowing out of the intersection of D and E, there is no water from the intersection of D, E, F, and G, and thus, no water flowing to H in the first possible answer.
Similarly, for the second possible answer, if there is no water at F, there is no water at the intersection of F and G. If, in addition, there is no water flowing out of the intersection of D and E, there is no water from the intersection of D, E, F, and G, and thus, no water flowing to H in the second possible answer.
For the last possible answer, if there is no water A, there is no water at the intersection of A, B and C, and thus, no water into H.
The correct solution has water at A, C and D. Thus, since there is water at A and C, there is water at the intersection of B and C, and thus there is water at the intersection of A, B and C. Since there is water at D, there is water at the intersection of D, E, F and G. Thus, in total, there is water at the intersection of A, B, C and D, E, F, G, and thus there is water at H.

Connections to Computer Science:
In this problem, there is the application of boolean logic: values of *true* (i.e., water flowing) and *false* (i.e., no water flowing) are combined together using the boolean operators AND (i.e., an intersection has restrictions on water flow) and OR (i.e., an intersection has no restrictions on water flow). There is water at H if the following boolean expression is *satisfied* (i.e., if it evaluates to true):

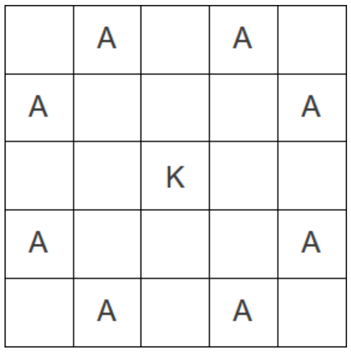
(A AND (B OR C)) AND ((D OR E) OR (F AND G))

where AND is true if and only if both of its arguments are true, and OR is true if and only if at least one of its arguments is true. You can verify that the correct answer is the only possible choice that does satisfy the above expression out of the four possibilities presented. Since computers use binary data (i.e., 0's and 1's) to store information, the actual machines use boolean logic, such as AND gates and OR gates, in the hardware of the machine. Also, many programs involve using boolean operators like AND and OR to make correct decisions based on data values (e.g., is the person using the system someone who is a female and who lives in either Alberta or Ontario?).

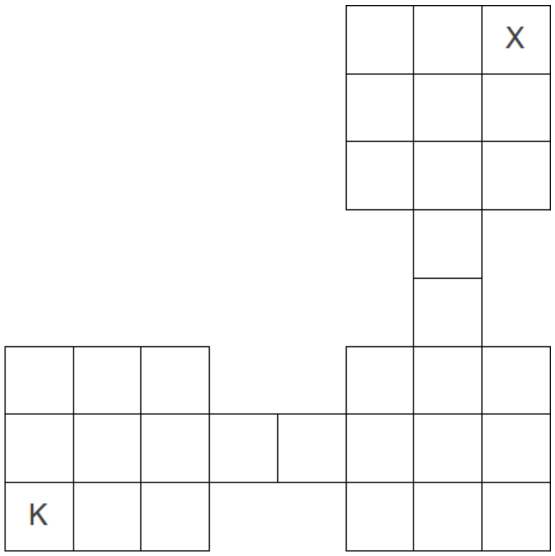
Knights Over Bridges

Question:

There are eight ways a knight can jump. If it starts at the position marked K on the grid below, in one move it can jump to any one of the eight positions marked A.



Suppose you have three 3-by-3 grids connected in the following way by two bridges of length two.



What is the fewest number of moves needed to move the knight from position K to position X without ever leaving the grids or bridges?

Possible Answers:

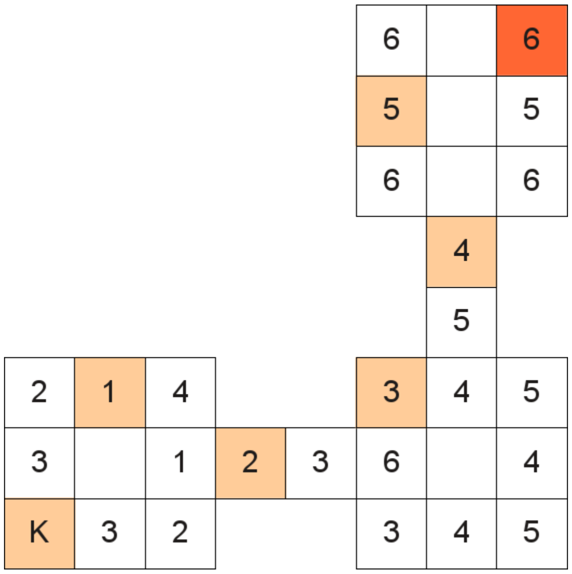
6
7
8
9

Correct Answer:

6

Explanation of Correct Answer

Label each position starting from K based on the fewest number of knight moves to get to that position. The essential elements of the grid are shown below:



Notice that the bridges act as "bottlenecks," which limit the number of possible paths. You could completely fill in the "shortest path length" to all positions, but this is not necessary for the solution.

Connections to Computer Science:

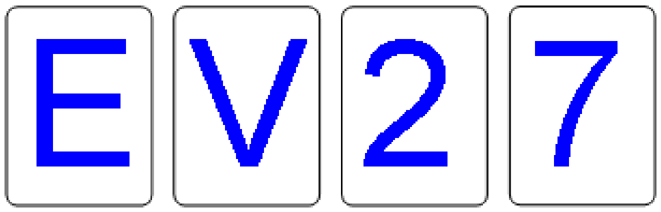
The solution to this problem uses one of the fundamental algorithms in computer science called *breadth-first search*. The idea of breadth-first search is to start at a set of possible positions and examine the next step for all these positions, before repeating this process. In this problem, we start with the starting position of the knight, and notice there are two positions that the knight can move in one turn. We then examine both of those positions to see what new positions we would end up at (that is, ignoring any positions we have already visited, such as the knights starting position). We repeat this process until we reach the destination location on the grid.

Internet traffic, electrical system and computer opponents in video games often use the breadth-first search algorithm to find the best or shortest path to reach the intended goal.

Turn the Cards

Question:

Cards have a letter on one side and a number on the other side. A beaver shows you the four cards below.



The beaver says:

"If there is a vowel on one side of a card, then there is always an even number on the opposite side of the same card."

A vowel is one of A, E, I, O or U.
You only see one side of each card so you do not know if the beaver is telling the truth.
Which cards must be flipped over to determine if the beaver is telling the truth?

Possible Answers:

All of them
E and 7
E and 2
E, V and 2

Correct Answer:
E and 7

Explanation of Correct Answer
You need to flip the E in order to check if there is an even number on the other side.
You don't need to flip the V. There are no statements concerning cards with a consonant.
You don't need to flip the 2. It is not forbidden to have a consonant on the other side.
You need to flip the 7. If the other side shows a vowel then the statement is false.

Connections to Computer Science:
To solve this problem, clearly understanding the specification of the problem is crucial, and quite often, misunderstood. The key phrase says that if there is a vowel on one side of the card, then there is always an even number on the opposite side of the same card.
In logic, we call this an *implication*: if X is true, then, by implication, it must be that Y is also true. Here, X is the statement that "there is a vowel on one side of the card", and Y is the statement that "there is an even number on the opposite side of the same card."
In logic, we would write this as $X \Rightarrow Y$.
Using a boolean logic table, we have the following definition:

X	Y	$X \Rightarrow Y$
false	false	true
false	true	true
true	false	false
true	true	true

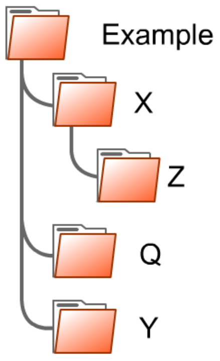
Notice that if X is false, the value of $X \Rightarrow Y$ is always true. In other words, if the card has a non-vowel on it, we do not care what is on the other side. Thus, we do not need to worry about the card with V on one side. Similarly, if Y is true, the value of $X \Rightarrow Y$ is always true. So, we need not worry about the card with 2 on one side.
Implication is often misunderstood by reversing the implication. For example, if $X \Rightarrow Y$, people incorrectly assume that if Y is in fact true, then X must have been true. A classic example is "If it rains, the road will be wet." Notice that this implication does not reverse itself: just because the road is wet does not imply that it must have rained. Understanding specifications of problems is an important aspect of computer science and software engineering.

Hierarchical Structure

Question:

When sharing files over the internet, programs often encode how files are organized in folders (also known as directories).

Suppose we have the folder Example with various folders underneath it, as illustrated below:



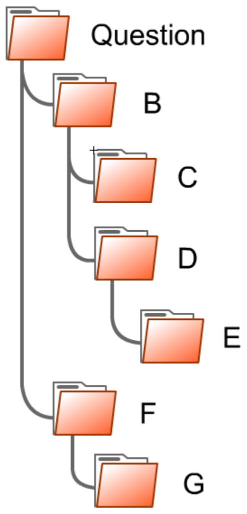
We can encode a folder and all of the folders which it contains using the following rule: each folder is encoded by its name followed by (followed by the encoding of any folders contained within it, followed by).

The encoding for the Example folder would be:

```
Example ( X ( Z ( ) )  Q ( )  Y ( ) )
```

Notice that folders which have no folders within them, such as Z in the example above, are denoted as ( ).

Which of the following is the encoding of the Question folder shown below?



Possible Answers:

```
Question ( B ( C ( ) D ( E ( ) ) ) F ( G ( ) ) )
Question ( B ( C D ( E ) ) F ( G ) )
Question ( B ( C ( D ( E ( ) ) ) ) F ( G ( ) ) )
Question ( B ( C ( D ( ) E ( ) ) ) F ( G ( ) ) )
```

Correct Answer:

`Question ( B ( C ( ) D ( E ( ) ) ) F ( G ( ) ) )`

Explanation of Correct Answer

In the second possible answer, the folder `C` needs to indicate that it does not have any folders underneath it, by saying `C ( )` instead of just `C`.
In the third and fourth possible answers, `D` is not a subfolder of `C` in the picture, but we have `C ( D ( ...) )` in this answer.
Verifying the first possible answer by matching the parentheses and ensuring the correct folders are subfolders demonstrates that the first answer is the correct answer.

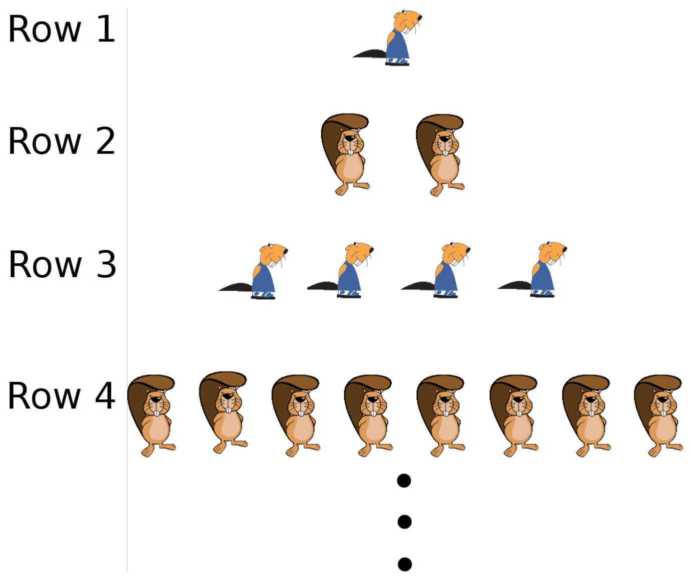
Connections to Computer Science:

In nearly all computer languages the *syntax* (structure) of the language has elements that must be balanced or matched-up correctly. In mathematical expressions, for example, parentheses must be matched up: the mathematical statement $(3+2)*(4+2)$ makes mathematical sense, but $(3+2(*(4+2)$ does not, due to a mismatched parenthesis. For example, web browsers know how to display web pages because they are marked up using the language HTML (HyperText Markup Language). In HTML, each region should have *tags* that begin and end that region: for instance, `<b>` marks the beginning of a bolded text region, and `</b>` marks the end of a bolded text region in HTML.
This problem also involves hierarchical structures, which in computer science are called *trees*: directory trees (as in this problem), search trees (in a graph), arithmetic expression trees and parse trees involve some hierarchical structure, and are used often in computer science.

Beaver Pyramid

Question:

Beavers arrange themselves in rows. There is one beaver in Row 1. Each row after Row 1 contains twice as many beavers as the previous row. The odd-numbered rows contain beavers facing sideways. The even-numbered rows contain beavers facing forwards.



In the first 4 rows, there are a total of 15 beavers.
If the pattern continues until there are a total of 511 beavers, how many beavers are facing sideways?

Possible Answers:

161
255
341
363

Correct Answer:

341

Explanation of Correct Answer

One approach is to simply count yielding $1 + 4 + 16 + 64 + 256 = 341$.
Alternatively, one can notice that there will be 9 rows and that every row of sideways-facing beavers after the first has twice as many beavers as the row just before it. This means that $\frac{2}{3}$ of the 510 beavers not in the first row are facing sideways. This is a total of $340 + 1 = 341$ beavers facing sideways.

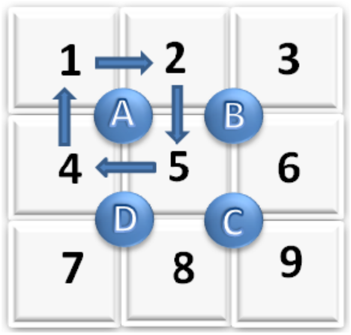
Connections to Computer Science:

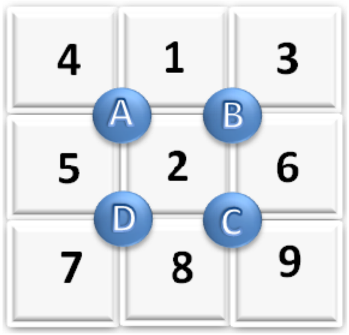
The beavers form a *binary tree* which is a fundamental data structure. A binary tree has the property that each element has at most two children (i.e., elements underneath it) in the tree: in this problem, each element has exactly two children. Binary trees are used in a variety of circumstances in computer science: decision trees (since many decisions are binary: either the decision is true or it is false), arithmetic trees and text-compression (using Huffman trees). Also, the use of powers of 2 is common in the mathematics of computing, due to the binary nature of the 0 and 1 values that computers use to store information.

Rotating Puzzle

Question:

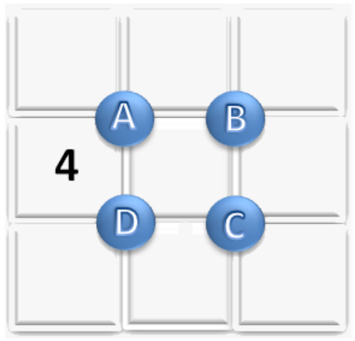
Henry Beaver plays a new game. If he presses one of the buttons A, B, C or D, the four numbers adjacent to the button will be rotated clockwise as shown in the picture on the left. The result of pressing the button A is shown in the picture on the right.

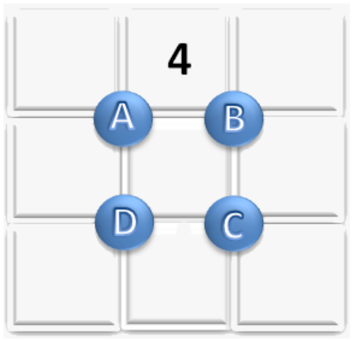


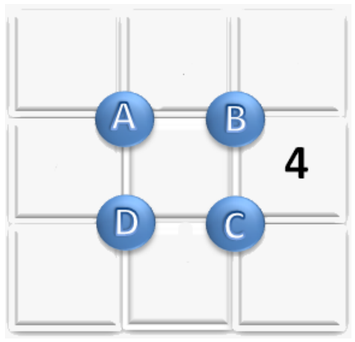


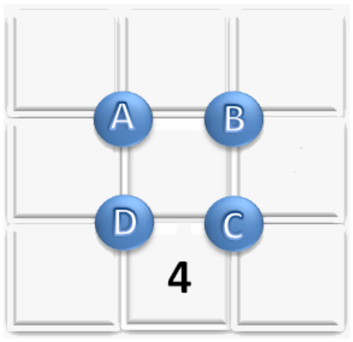
Starting from the picture above on the left, Henry Beaver pressed four buttons in the order of D, C, B, B.
Where is the number 4 after Henry pressed the buttons?

Possible Answers:

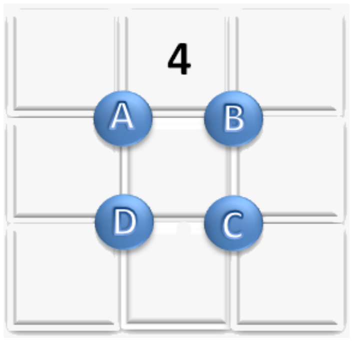




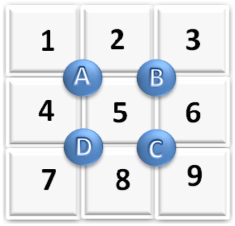




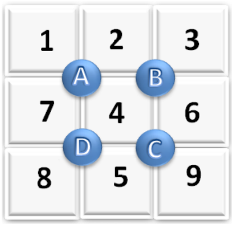
Correct Answer:



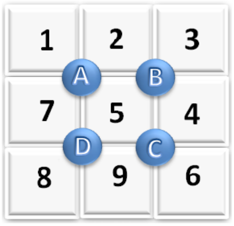
Explanation of Correct Answer



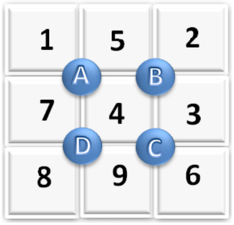
⇒ Press D ⇒



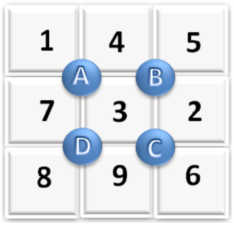
⇒ Press C ⇒



⇒ Press B ⇒



⇒ Press B ⇒



Connections to Computer Science:

The problem is solved by following the algorithm described (i.e., rotating the four-squares around the particular letter), but also keeping track of the *state* of the board after each step. In many computer applications, the state of the system (e.g., word processor, computer game or internet browser) changes via simple button pushes, and the computer application must keep track of what the current state is.

Programmed Robot

Question:

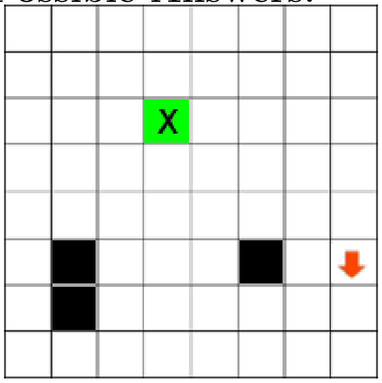
A robot is programmed to find a target (the green field marked with X) on a map of square fields. The robot has its movements programmed as follows:

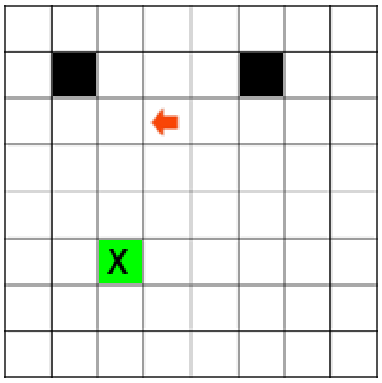
- The robot moves straight forward until it reaches an obstacle (black field) or the edge of the map.
- When reaching an obstacle or the edge of the map, the robot turns right by 90°.
- When the robot moves out of a field, the field becomes a black obstacle.

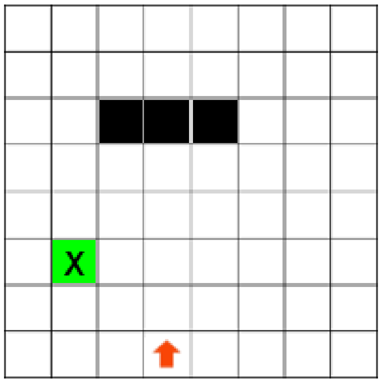
The arrows on the maps below show the starting position as well as the starting direction of the robot.

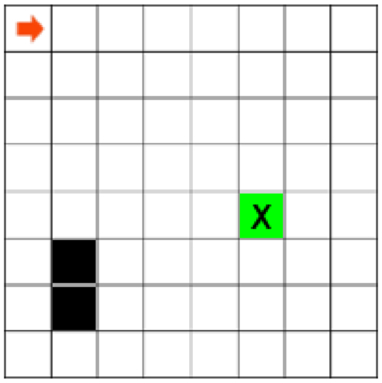
On which map does the robot NOT eventually reach the target (green field marked with X)?

Possible Answers:

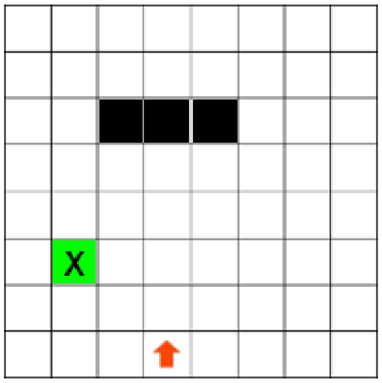






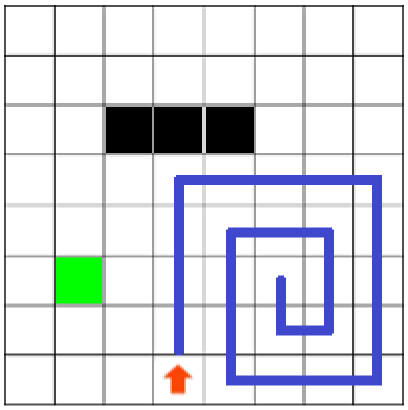


Correct Answer:

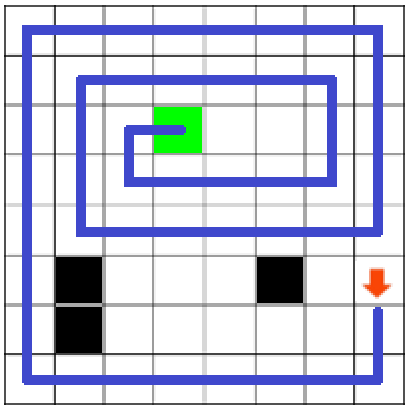


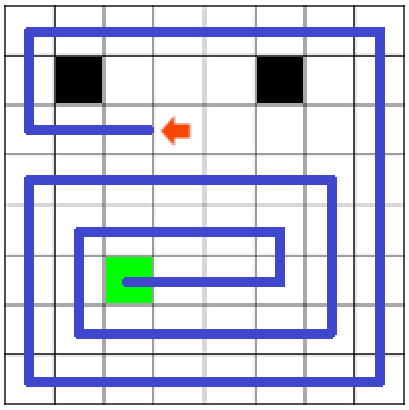
Explanation of Correct Answer

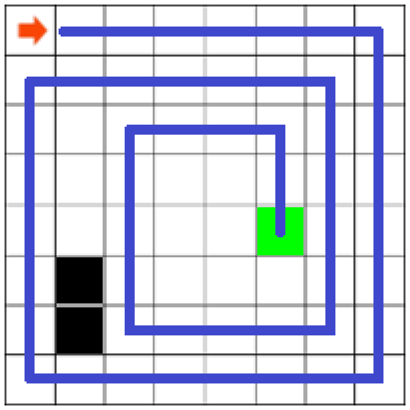
On the third figure, the robot does not reach the green target field, since he either encounters the 3 block obstacle, the right edge or his previous paths during his travels. The robot's path is illustrated below:



The other three grids, where the robot always reaches the green target field, are shown below:







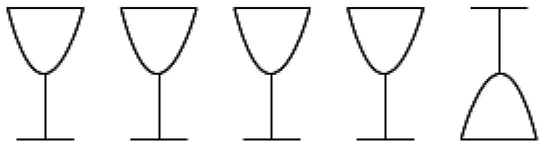
Connections to Computer Science:

As mentioned in previous problems, algorithms are fundamental in computer science. The algorithm in this problem provides a sequence of steps to solve a particular problem (in this case, to find the target field). It is crucially important that algorithms are designed in a way that they *always* solve the given problem and not only some of the cases of the problem. Therefore the algorithm presented in this task, should be replaced by a better one, which will always find the target field.

Glasses

Question:

There are 5 empty glasses on a table. One is facing down and four are facing up.



Flipping a glass changes it from facing up to facing down, or from facing down to facing up. In one turn, you must flip exactly three different glasses. The glasses which are flipped do not need to be adjacent.
What is the minimum number of turns to make all glasses facing up?

Possible Answers:

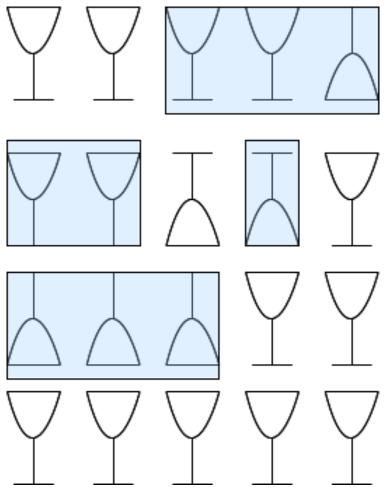
2 turns
3 turns
5 turns
it is not possible to make all glasses facing up

Correct Answer:

3 turns

Explanation of Correct Answer

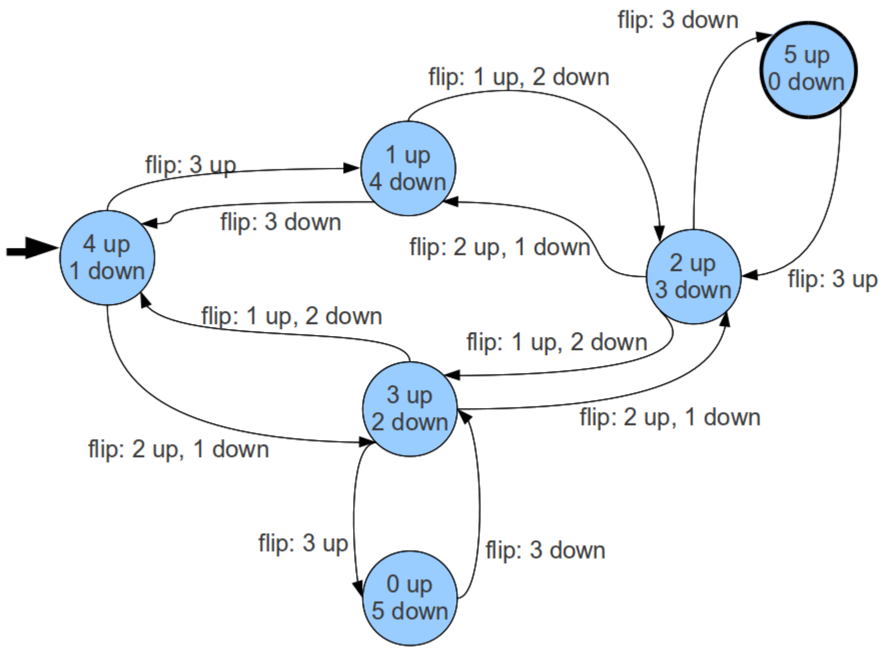
The following illustration shows that 3 turns is a possible solution:



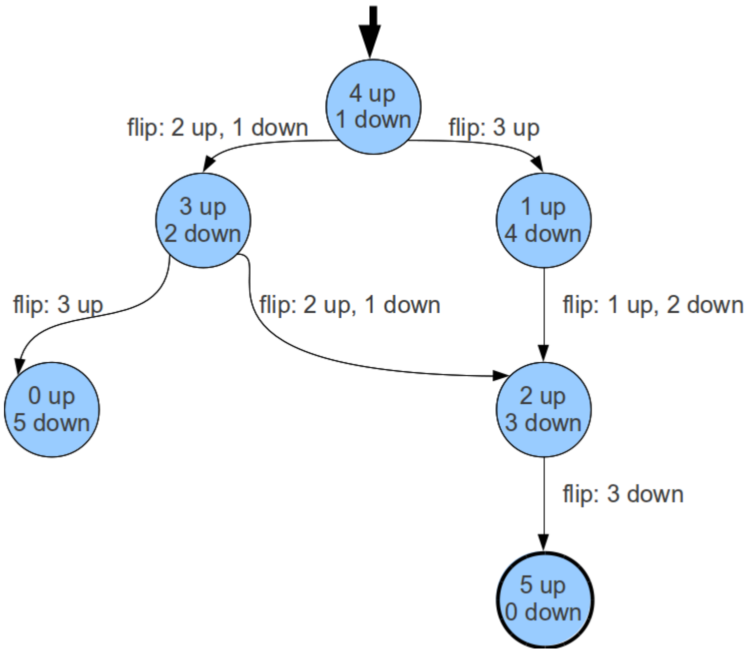
Notice that there must be an odd number of turns, since after the first turn, there will be either 2 (if we flip the glass facing down and two other glasses) or 4 (if we flip three glasses which are facing up) glasses which are facing down. Using the same analysis, on the next turn, there will be an odd number (1,3,5) of glasses which are facing down. Thus, we require more than 2 moves and in general we require an odd number of moves. We have shown a solution which uses 3 turns, which must be the minimum.

We can also visualize the transformation as changing states in a *deterministic finite automaton* (also known as a *finite state machine*). To understand what is happening in the picture below:

- start at the circle marked with the large black arrow (this is called the *start state*)
- move between the circles based on the words above the arrow
- when the arrow says "flip: X up, Y down," it means that X of the glasses that are currently right-side-up should be flipped upside-down, and Y of the glasses that are currently upside-down should be flipped right-side-up.
- we would like to end up in the state which has a darker line around it (in this case, the state marked "5 up, 0 down") (this is called the *final state*)



Since we are looking for a minimum number of steps, we can create a tree through this DFA where we only add a circle when we have not already encountered it already. This is called a breadth-first search through the DFA, and notice that the shortest path from the start to the finish is 3 steps, and that there, in fact, two different ways to get from the start state to the final state.



Connections to Computer Science:

This problem involves following an algorithm, and keeping track of state of the "system" or "variables." The state is the number of glasses that are oriented up (which also determines the number of glasses that are down).
Reasoning about parity and arguing correctness for an algorithm are important aspects of computer science. One possible way to analyze the solution is to consider either deterministic finite automata or to consider a breadth-first search.

Bebrocarina

Question:

A bebrocarina is a musical instrument with the following features.

- It can play only 6 different tones.
- The tones can be arranged from lowest to highest
- After having played one tone, it is possible to play only the same tone, the next higher tone (if it exists) or the next lower tone (if it exists).

This means that melodies can be represented using only three different symbols:

- = means "the current tone must be the same as the previous tone", and
- - means "the current tone must be one lower than the previous tone", and
- + means "the current tone must be one higher than the previous tone."

For example, melody `- +` means "play 3 tones, the second tone is lower than the first one and the third tone is higher than the second tone (i.e. the same as the first tone)."
For which of these melodies is there no starting tone that makes playing the melody possible?

Possible Answers:

```
+ = = = + = = = + = = = + = = = +
- - - = + - = - - = = = +
- - - - - = + + + + + = - - - - -
- - + - - + - - = - + - -
```

Correct Answer:

`- - + - - + - - = - + - -`

Explanation of Correct Answer

If a bebrocarina can play only 6 different tones, the player can move from the highest to the lowest tone (or vice versa) in as little as 5 steps. These 5 steps correspond to "-" (or "+") marks in notation of melody. Thus, the difference between the number of "-" marks and "+" marks should not be bigger than 5 at every point in the representation. The number of "=" marks has no effect, and thus, we can have any number of "=" marks.

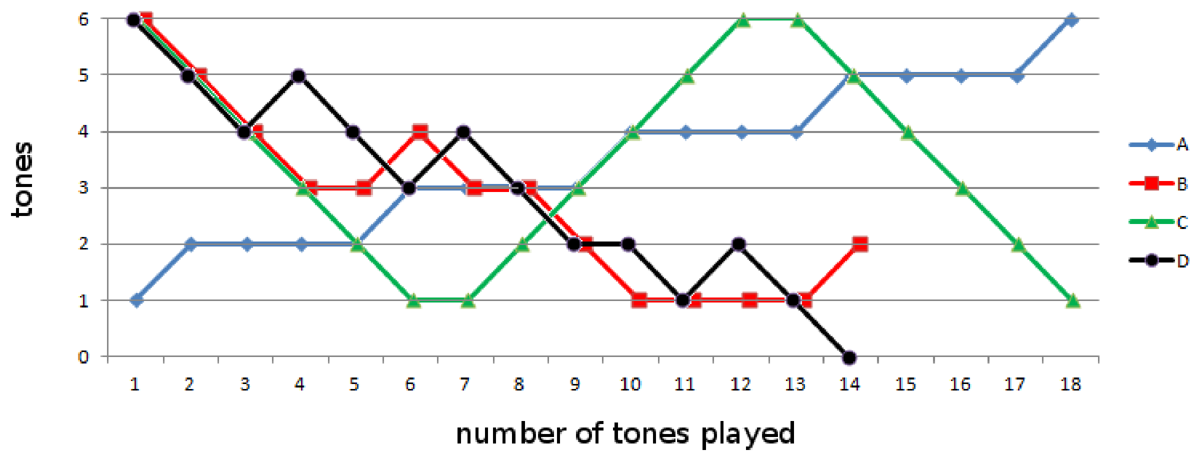
The first possible answer can be played because the sequence of the tones passes through only 5 different tones.

The second possible answer has 6 "+" marks but the one "-" mark in between, yields the total number of tones as 5.

The third possible answer has a sequence of 5 "-" marks leading from the highest to the lowest tone and the following sequence of 5 "+" marks returns the melody to the highest tone. The last sequence of 5 "-" marks leads to the lowest tone once more. We do not need more than 6 tones.

The last possible answer contains 9 "-" signs and only 3 "+" signs, the difference is 6.

These four possible answers are illustrated in the graph below: we care only about the "maximum difference" between the highest tone played and the lowest tone played, and that the last possible answer requires 7 different tones.



Connections to Computer Science:

All information processed by a computer must be encoded into some computer readable format: at the lowest level, this is *binary* (i.e., 0s and 1s), but it could also be ASCII values or one of many different file formats (e.g., .png, .gif, .doc, etc.). In this problem, we are encoding the difference in tones using symbols.

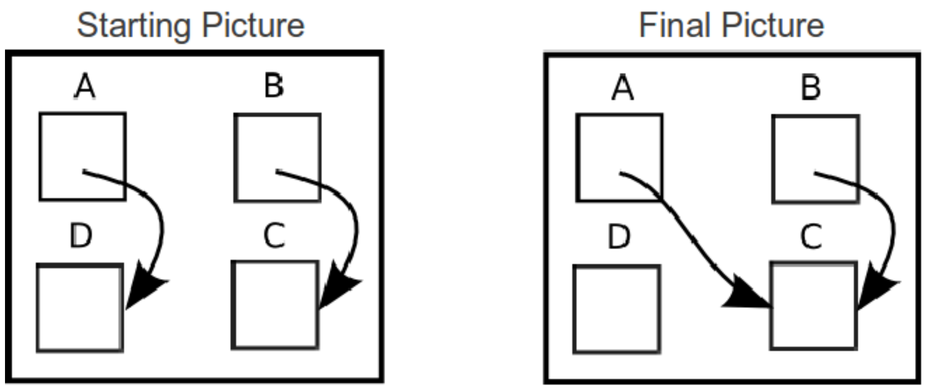
With these encodings, we have *abstracted* away some details to make the problem easier to understand. We do not actually concern ourselves with the specific tone, but rather, the relative difference between two tones. Abstraction is a key concept in computer science: for example, we do not concern ourselves *how* variables are stored in a computer program, but what their values actually are.

Change Direction

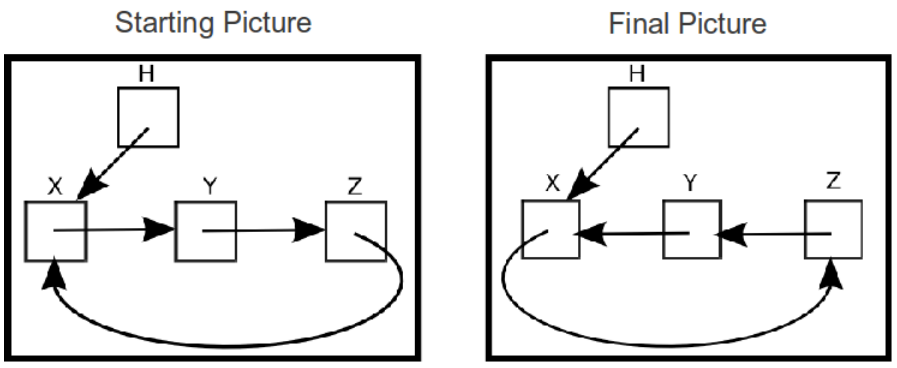
Question:
The instruction A<=B changes a picture of boxes and arrows in the following way:

- The arrow which points out of the box labelled A is removed.
- Then, a new arrow out of the box labelled A is added. This new arrow points to the same box as the arrow out of the box labelled B points to.

For example:



What sequence of instructions (performed in order) changes the following starting picture to the following final picture?



Possible Answers:

X<=Y	X <= Z	Z <= Y	Z <= X
Y<=Z	Z <= X	X <= Z	X <= Y
Z<=X	Y <= H	Y <= H	Y <= H

Correct Answer:

```
Z <= X
X <= Y
Y <= H
```

Explanation of Correct Answer

The first possible answer is incorrect because the original value of `X` has changed before `Z<=X` is followed. Also, the operation `X<=Y` will lose the reference to `Y` (that is, nothing will be pointing at `Y`, and we cannot access or change the value of `Y`).

The second possible answer is incorrect because it attempts to swap `X` and `Z` without temporary storage. As in the first possible answer, we lose the reference to `Y`.

The third possible answer is incorrect because it misreads `A<=B` to mean "box A points to box B."

The fourth possible answer is correct because unlike the first possible answer, it correctly uses the "temporary value H."

Connections to Computer Science:

Many computer programming languages have the concept of memory that is *linked* together, either by way of *pointers* (in C++, for example) or by *references* (in Java, for example). This concept of data related to each other also applies for applications such as databases.

The visualization for linked memory typically involves boxes and arrows: the boxes representing the actual data, and the arrows representing some sort of relationship between the data. These visualizations help keep track of relationships when they are altered, and ensure that the correct relationships can be achieved.

Find and Transpose

Question:

Randi is often given a name to find in a long list of names. She looks for the name by comparing it to each name in the list from left to right. She always finds the name and then stops looking. If the name is found anywhere but the first position in the list, she then swaps it with the name to the left of it in the list.
For example, if Randi is given the list:

Trevor, Julia, Mark, Isaac, Florence, Bob, Henry

and asked to find Mark, she compares Mark to Trevor, Julia and Mark and then changes the list to:

Trevor, Mark, Julia, Isaac, Florence, Bob, Henry

If she is then asked to find Henry, she compares Henry to every name in the list and then changes the list to:

Trevor, Mark, Julia, Isaac, Florence, Henry, Bob

Over these two searches, Randi performs a total of $3 + 7 = 10$ comparisons.
If Randi starts with a list of 10 different names and is asked to find each name exactly once, what is the maximum number of total comparisons she could possibly perform?

Possible Answers:

55
60
64
100

Correct Answer:
64

Explanation of Correct Answer
First assume that Randi does not move names when she finds them. Then she performs 1 comparison to find the first name, two to find the second name, three to find the third name and so on. This is a total of $1 + 2 + 3 + ... + 10 = 55$ comparisons.
Since Randi only looks for each name once, swapping a name with the preceding name can only have the effect of increasing the number of comparisons of the preceding name by one. The number of comparisons when searching for other names is not affected. Therefore transposing can only increase the number of comparisons (by one) for all but the first search. This means transposing can increase the number of comparisons by at most 9 in total for a total of 64 comparisons.
To show this is achievable, simply consider a search for the strings from the second to the last in order from left to right followed by a final search for the name originally first in the list. The first 9 searches will require $2+3+...+10 = 54$ comparisons and then 10 comparisons will be needed to find the name originally first but now last. This is a total of 64 comparisons.

Connections to Computer Science:
One of the fundamental problems in computer science is how to organize data in order to search within it quickly. There are many ways to do this: using binary trees, splay trees, skip lists, sorted arrays, etc. The technique outlined in this problem is the idea of moving found items closer to the "front," with the assumption that if we search for something once, it is quite likely that that same item will be searched for again. The *transpose heuristic* used by Randi in this problem is one technique for doing this. Other heuristics include *move-to-front*, which moves a found element to the very front of the list.
Moreover, this problem highlights the process of performing *worst-case analysis* for an algorithm. Computer scientists care about "what is the worst possible input for this algorithm, and how long will it take to execute on that input?" In this question, we are asking about the worst-case performance of the transpose heuristic on a list of size 10.