



UNIVERSITY OF
WATERLOO



The CENTRE for EDUCATION in
MATHEMATICS and COMPUTING



2020
*Beaver
Computing
Challenge*
(Grade 7 & 8)

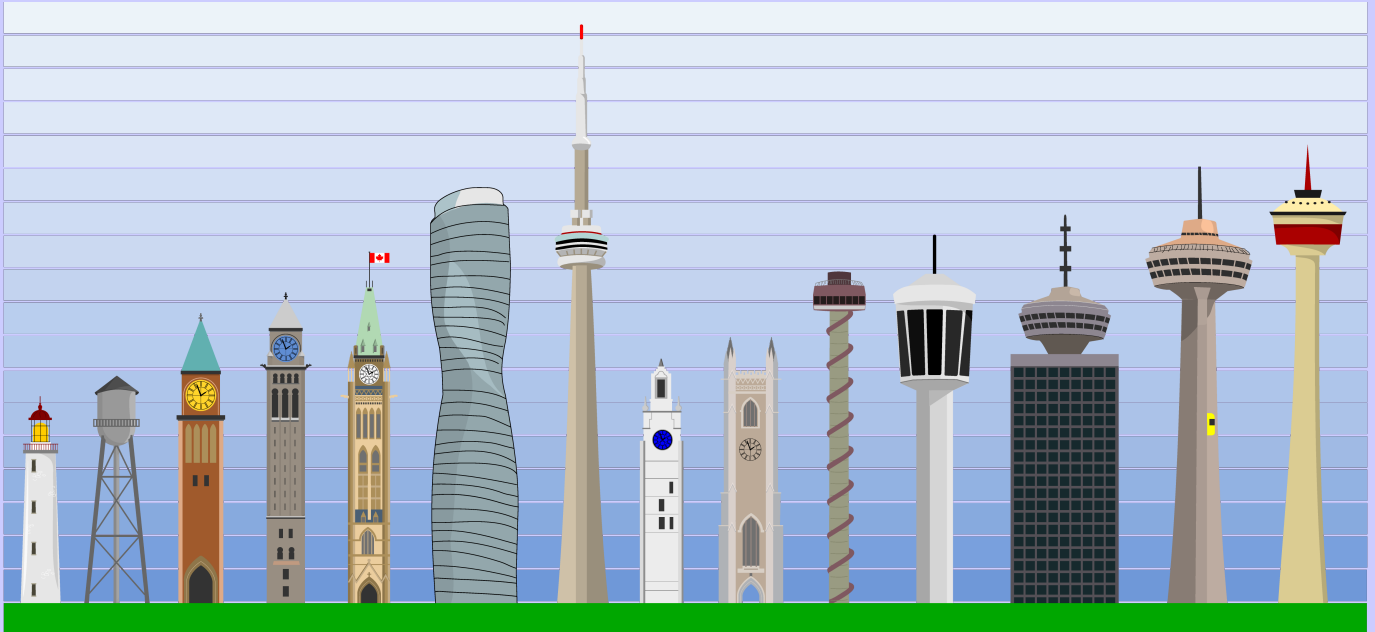
*Questions,
Answers,
Explanations,
and
Connections*

Part A

Skyline

Story

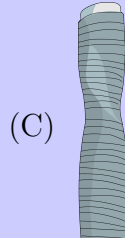
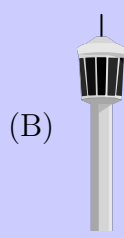
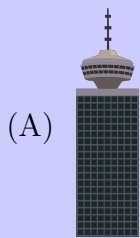
A skyline consists of 14 towers as shown.



The height of a tower is measured from the bottom of its base to its highest point, including any flagpoles or antennas.

Question

If the towers are listed from shortest to tallest, which tower would be 10th in the list?



Answer

(A)



Explanation of Answer

Notice that the skyline consists of one sequence of towers that gets taller from left to right followed by a second sequence of towers that gets taller from left to right. Also notice that the last two towers in each of these sequences make up the four tallest towers overall. (This did not have to be true. It could have been the case, for example, that the four tallest towers were all at the end of one of the two sequences.)

Since there are 14 towers in total, if we ignore the four tallest towers, the tallest remaining tower will be 10th in the list. The tallest remaining tower is the one in Option A.

Connections to Computer Science

In this task, you are asked to determine which tower would be the 10th in a list of towers if all the towers were ordered by height. Arranging items in order is called *sorting*.

There are many well-known sorting algorithms and this task is related to the *mergesort algorithm*. The key idea behind this technique is to separately sort the two halves of the items. Then you must *merge* these two sorted lists to produce one completely sorted list. If you chose to solve this problem by sorting all 14 towers, then you could have been merging the sorted towers in the left half with the sorted towers in the right half.

Country of Original Author

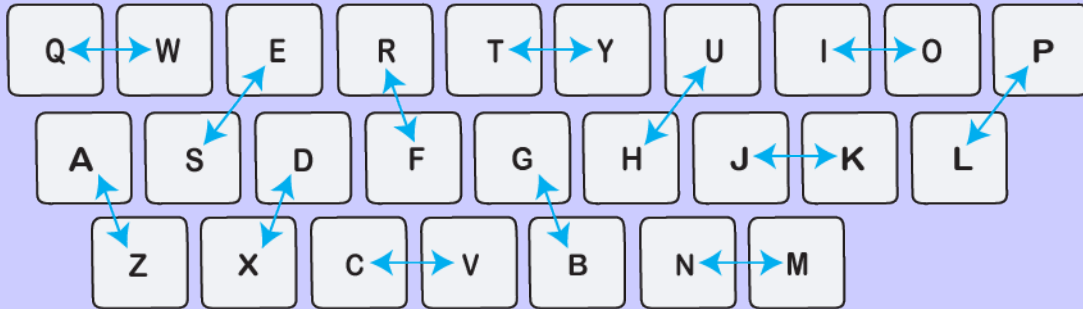
Canada



Crypto Keys

Story

Jan uses a special keyboard for writing secret messages. When a key on the keyboard is pressed, a different letter is displayed on the screen, according to the following keyboard map:



The arrows indicate which letter is displayed when each key is pressed. For example, when Jan presses the “S” key, the letter “E” is displayed on the screen, and when Jan presses the “E” key, the letter “S” is displayed on the screen.

Jan types a message and the letters “QOEU” are displayed on the screen, in that order.

Question

What was the original message typed by Jan?

- (A) WASH
- (B) WITH
- (C) WISP
- (D) WISH

Answer

(D) WISH

Explanation of Answer

Jan’s original message can be determined by matching each letter displayed on the screen with the key that must be pressed in order to display that letter, according to the arrows on the keyboard map. For example, since the first letter displayed on the screen is “Q”, the first key pressed must have been the “W” key. The keys that must have been pressed, in order, are shown in the following table.

Letter displayed	Q	O	E	U
Key pressed	W	I	S	H

Therefore, the original message typed by Jan was “WISH”.

Connections to Computer Science

Cryptography is an area of study that lies at the intersection of computer science and mathematics.

The keyboard used in this task is based on the *monoalphabetic substitution cipher* example called the *Vatsyayana cipher*. The idea came from an Indian text from the 4th century AD. The Vatsyayana cipher creates unique pairs for the alphabet letters – one letter always matches another letter and one letter can be used only in one pair. During *encryption* in an original message one letter is directly substituted by a paired letter. Interestingly, the exact same process is used to *decrypt* the message. By directly substituting a letter in the encrypted message with its paired letter, the original message can be retrieved. This encryption method is easy to crack as, once someone is sure about a letter pair(s), they can decrypt all other pairs.

Modern cryptographic techniques, such as those that are used for banking transactions through the internet, use much more advanced computational methods that rely on the difficulty of hard mathematical problems, such as factoring a number which has two very large prime factors.

Country of Original Author

Canada



Cookies

Story

Four children ask for cookies.

Adam says "I don't want stripes on my cookie."

Bella says "I want my cookie to be a circle or a square."





Cai says "I want a cookie with little round dots."

Diego says "I want a star-shaped cookie."





Question

Which of the following assignment of cookies will satisfy all the children's requests?





(A)

Adam	Bella	Cai	Diego
			





(B)

Adam	Bella	Cai	Diego
			





(C)

Adam	Bella	Cai	Diego
			

(D)

Adam	Bella	Cai	Diego
			

Answer

	Adam	Bella	Cai	Diego
(C)				

Explanation of Answer

We can determine the correct answer by considering the children's requests in any order.

One way to do this is to first look at what Cai says. This eliminates Option A and Option B for which Cai is given a cookie with hearts and stripes respectively, and not little round dots. So now we know that the right answer must be Option C or Option D.

If we look at what Diego says next, we can eliminate Option D because in Option D, Diego is given a circle-shaped cookie instead of a star-shaped one. This means only Option C remains as a possibility and so it must be the correct answer. We can confirm that Adam and Bella's requests are also satisfied by the assignment of cookies in Option C.

Connections to Computer Science

In this problem, you are asked to solve a *constraint satisfaction problem*. The constraints are the restrictions that each child has for their desired cookie.

In general, a constraint satisfaction problem involves finding a solution that obeys a given set of rules. Other well-known examples are sudoku puzzles and trying to colour a map such that two neighbouring regions are not the same colour. These problems are often very difficult to solve. Trying all the possibilities often takes too long so computer scientists look for more advanced *search algorithms* that will produce a solution.

Country of Original Author

South Korea

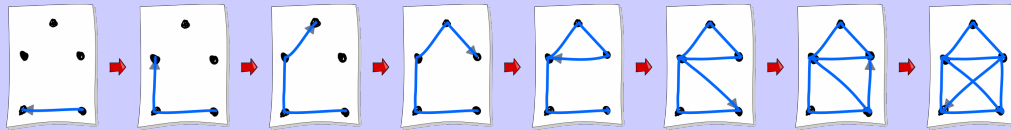


Connect the Dots

Story

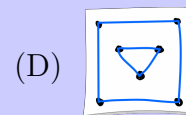
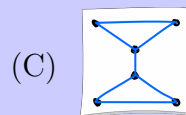
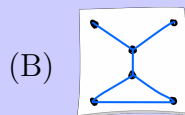
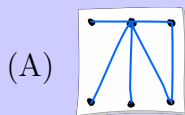
Zhi likes to draw. He creates his pictures by drawing dots and then connecting them with line segments in one motion, never picking up his pencil and never drawing the same line segment twice.

This is how Zhi draws a picture of a house:

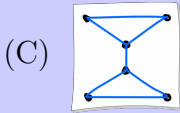


Question

Which of the following pictures can Zhi draw?

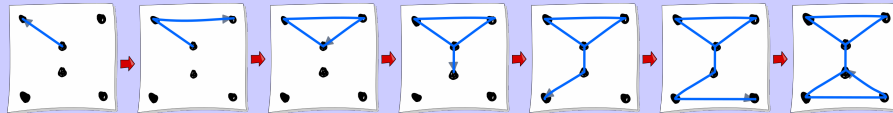


Answer



Explanation of Answer

The steps below show that the picture in Option C can be drawn by Zhi.



We should also justify why Zhi cannot draw the other pictures.

Since a line segment cannot be drawn twice, a dot joined to exactly one line segment must be where Zhi begins or finishes his drawing. Since there are more than two dots joined to exactly one line segment in Option A, Zhi cannot draw this picture.

Using this same idea, to draw Option B, Zhi must begin drawing at one of the top two dots and finish drawing at the other of the two top dots. This forces him to draw the line segment in the middle twice which is not allowed. For this reason, Zhi cannot draw the picture in Option B.

Finally, Option D consists of two disconnected pieces so there is no way for Zhi to draw this picture without picking up his pencil.

In general, if a picture is disconnected, then it cannot be drawn. Otherwise, we can determine if a picture can be drawn by counting the number of line segments joined to each dot. Most of these totals must be even numbers (i.e. 2, 4, 8, etc.). Specifically, a connected picture can be drawn exactly when all of these totals are even numbers or all of the totals, except for two totals, are even numbers. Can you convince yourself of this fact?

Connections to Computer Science

These drawings are made of points and lines connecting the points. In computer science this is a way of representing objects and relationships among objects: the points represent objects and the lines represent connections or relationships among them. Such a representation is called a *graph*. A graph consists of a set of *vertices* or *nodes* (usually depicted as points or circles) and a set of *edges* (usually depicted as lines, possibly curved, sometimes directed with an arrow). The edges connect vertices. A sequence of connected edges in a graph is called a *path*.

A graph representing Zhi's drawing is special: the graph has to be connected (there is a path from any vertex to any other vertex by following edges of the graph) and either no vertices have an odd number of adjoining edges, or exactly two of the vertices have an odd number of adjoining edges.

For graphs that have these two special properties, there is a way to follow all the edges in the graph exactly once. A drawing like this is called an *Eulerian path* named after Leonhard Euler (1707 – 1783) who first described this problem. Euler came up with the concept by trying to solve the problem of the Seven Bridges of Königsberg. Today we can find Eulerian paths by applying the well-known *Fleury's algorithm* or *Hierholzer's algorithm*.

Country of Original Author

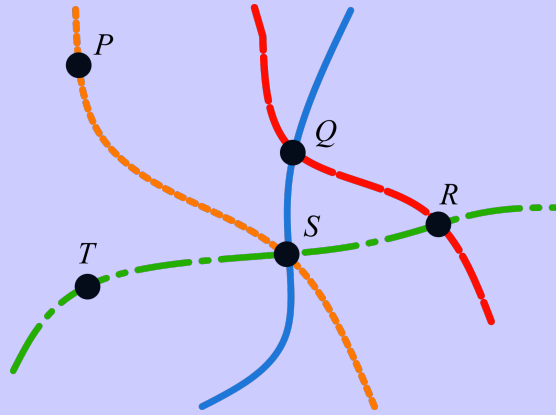
Macedonia



Towns and Highways

Story

A map of five towns (black dots) and four highways (coloured lines) is shown.

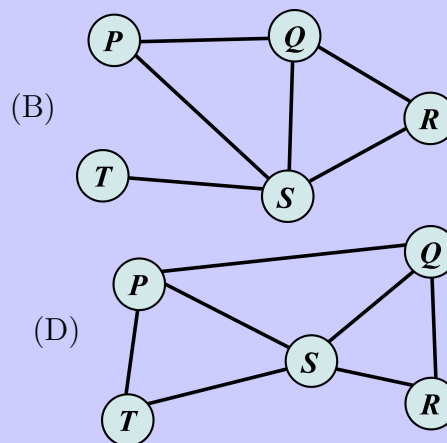
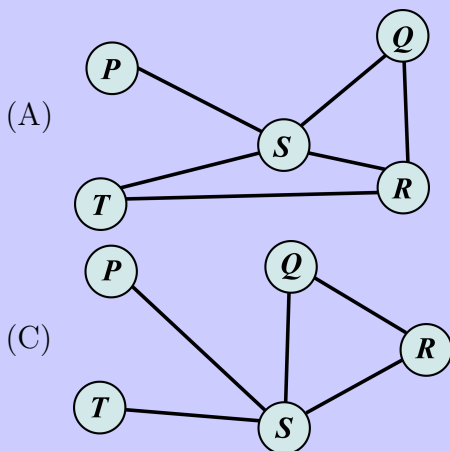


To represent this map using a diagram, there is one labelled circle per town and the following is true for every two towns:

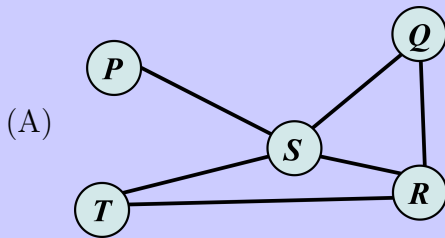
1. If you can drive from one town to the other using exactly one of the four highways, then a straight line joins their circles.
2. If you cannot drive from one town to the other using exactly one of the four highways, then no straight line joins their circles.

Question

Which diagram represents the given map?



Answer



Explanation of Answer

By studying the map we can see that:

- You can drive from P to S using the orange highway.
- You can drive from Q to R using the red highway.
- You can drive from Q to S using the blue highway.
- You can drive from T to S using the green highway.
- You can drive from T to R using the green highway.
- You can drive from S to R using the green highway.

The diagram in Option A has exactly six straight lines joining circles. These correspond to the six points above. The diagram in Option C has five straight line connections which is too few. The diagram in Option D has seven straight line connections which is too many. The diagram in Option B is missing a straight line joining the towns corresponding to T and R and has a line segment joining towns P and Q which should not be there.

Connections to Computer Science

Being able to focus on the important information in a problem is a key concept in computational thinking. This process is sometimes called *abstraction*, which is where the most important details are focussed on and irrelevant details are ignored in order to make solving the problem easier.

In this problem, the abstraction we use is a *graph*. The circles in the diagram are called *vertices* or *nodes* of the graph, and the connections between vertices are known as *edges*. The key concept in this problem is determining which vertices should be connected by edges, but the relative placement of the vertices and whether the edge is straight or curved are irrelevant details that can be ignored.

Country of Original Author

Slovakia



Part B

Library Books

Story

Beavertown Library has only a small pile of books. When a beaver wishes to borrow a book, they take the book that is on the top of the pile and record their name. When a beaver returns a book, they place their book on the top of the pile and record their name again.

At the beginning of the week the pile of books was arranged as shown:



The library's records at the end of the week show the following information:



Question

Which book did Cato borrow?

- (A) Charlotte's Web
- (B) Curious George
- (C) Go, Dog, Go!
- (D) The Hobbit

Answer

(B) Curious George

Explanation of Answer

At the beginning of the week, Alba borrowed a book and then Felix borrowed a book. Looking at how the books were arranged at the beginning, this tells us that Alba borrowed *Charlotte's Web* and Felix borrowed *Curious George*.

At the end of the week, Cato borrowed a book immediately after Felix returned a book. Since books are taken from the top of the pile and returned to the top of the pile, this means Cato borrowed the same book that Felix returned. Since Felix borrowed only one book that week, we know that he must have returned *Curious George*. Thus, Cato borrowed *Curious George*.

It is interesting to notice that we do not need to consider all of the books that were borrowed and returned. For example, it does not matter which book Marta borrowed.

Connections to Computer Science

In this problem, you are asked to keep track of which books are available as they are borrowed and returned. The key property is that the last book returned is the first book that must be borrowed the next time someone borrows a book. We say that this follows the *last-in first-out* or *LIFO* principle.

In computer science, a collection of items that follows this principle is called a *stack*. It is one fundamental way that a collection can be stored. In particular, a *computer program* will often be broken into smaller pieces called *subroutines*. The order in which these subroutines are executed is often determined using a stack. Another example of a stack is a web browser's back button: the last page visited is the first page revisited when the back button is pressed.



Country of Original Author

Canada



















Market Exchange



Story

A beaver goes to a market to trade items. It has one carrot  but needs one fir tree .

Each stall of the market allows a different trade as shown:

Stall	Give	Get
P		
Q		
R		
S		
T		
U		
V		
W		

Question

Which of the following sequences of stalls should the beaver visit in order to trade its carrot  for one fir tree .

- (A) P, Q, T
- (B) W, T, U
- (C) S, V, U
- (D) S, R, U

Answer

(C) S, V, U

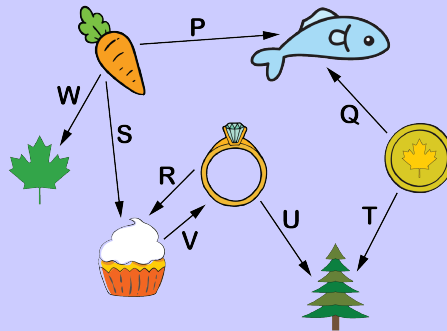
Explanation of Answer

At stall S the beaver can trade the carrot 🥕 for a cupcake 🍰.

At stall V the beaver can trade the cupcake 🍰 for a ring 💍.

At stall U the beaver can trade the ring 💍 for a fir tree 🌲.

To consider the other options, the following diagram can help us understand the possible trades at various stalls. An arrow from item X to item Y , labelled Z , means that at stall Z the item X can be traded for item Y .



Notice how you can now trace the connection from a carrot to a fir tree. Also, notice that when starting at the carrot, the only way to reach the fir tree is by following the arrows labelled S , V , and U , in that order.

Connections to Computer Science

The ways in which items are traded at this market can be modelled by a *directed graph*. We can represent the items as the *vertices* of the directed graph and a possible trade of one item for another as a (*directed*) *edge*. In the diagram, the edges are the arrows labelled with the particular stall that can trade one item for another.

The reason we say “directed” is because the trade is only in one direction. For example, if item A can be traded for item B at stall X, then we have a directed edge from item A to item B labelled with X. In this case, we call B a *neighbour* of A.

In this problem, we want to find a *directed path* from the carrot to the fir tree. One way to do this is to perform a *breadth-first search*, where all of the neighbours of the carrot are determined, and then repeatedly determine the neighbours of each of those neighbours, and so on.

The idea of searching for a path in a directed graph has many applications, such as mapping out a driving route, determining how to send information through the internet, and determining recommendations for who you may want to connect with on social media platforms.

Country of Original Author

Switzerland



House Painting

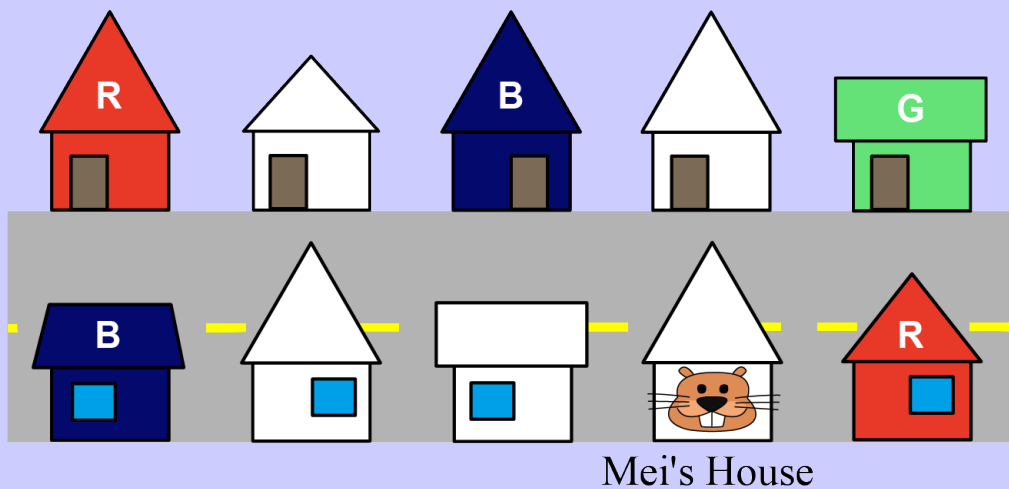
Story

To brighten up the street Mei lives on, each white house will be painted red (R), green (G) or blue (B).

After all the houses have been painted, the following must be true:

1. Two houses next to each other must not be the same colour.
2. A house must not be the same colour as the house directly across the street.

Before painting, these are the houses on Mei's street:



Question

Which colour(s) can be used for Mei's house?

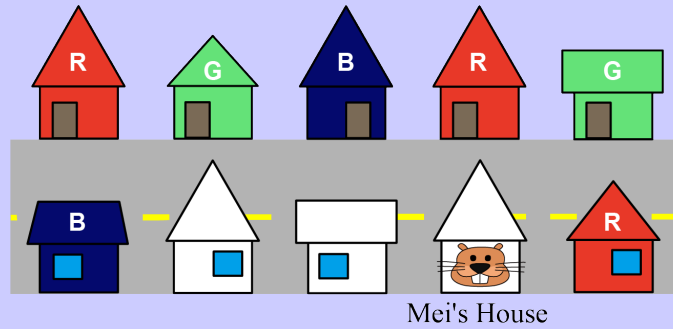
- (A) Only red can be used.
- (B) Only blue can be used.
- (C) Only green can be used.
- (D) Either red or green can be used.

Answer

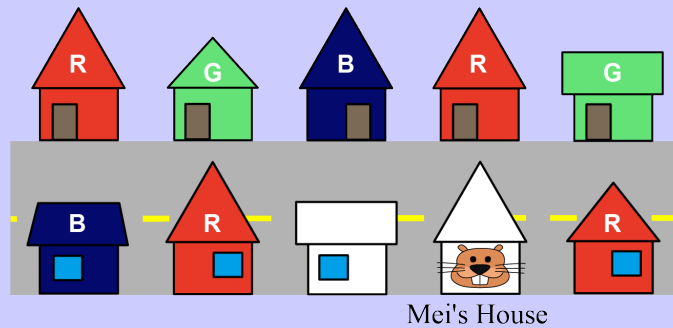
(B) Only blue can be used.

Explanation of Answer

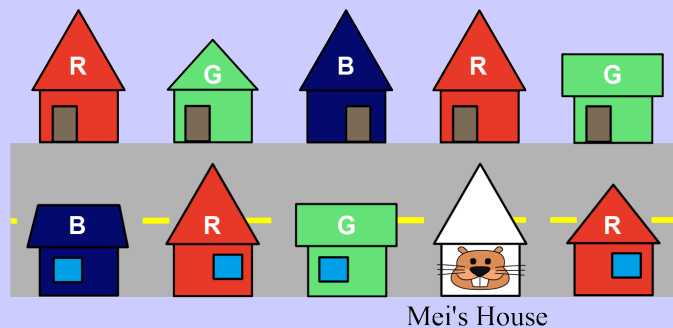
Since two houses next to each other cannot be the same colour, the two white houses along the top of the street must be painted green and red as shown here:



Knowing this, and the fact that houses directly across the street from each other must also be different colours, we can determine that the house two doors to the left of Mei's house must be red:



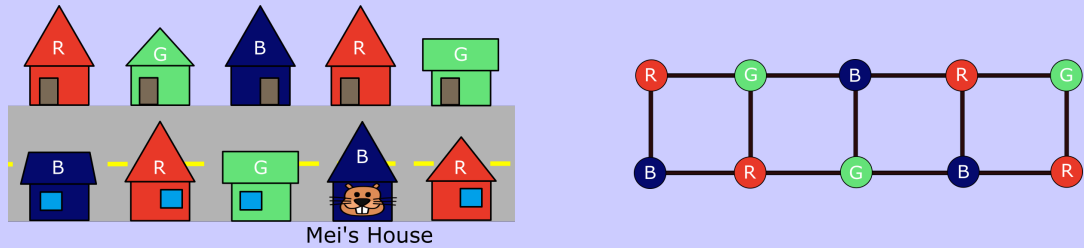
Continuing in this way, the house one door to the left of Mei's house must be green:



We have determined that the two houses next to Mei must be green and red and the house directly across from Mei must be red. Therefore Mei's house must be painted blue.

Connections to Computer Science

The ten houses and their position relative to each other form a *graph*, a representation that is often used in computer science to represent data and the connections between data. We can represent the original house pictures, on the left, with the more *abstract* graph representation on the right.



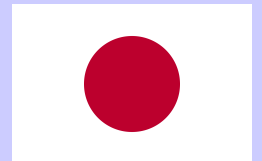
A graph is composed of *vertices* (the coloured circles in the diagram) and *edges* (the lines connecting these circles). For this task, each vertex corresponds to a house and each edge corresponds to one of the relations “is next to” or “is directly across the street from”.

This task requires us to colour the vertices of the graph in such a way that no edge connects two vertices of the same colour. The same question can be asked of any graph, but depending on which graph you choose, you may not always be able to do this with only three colours.

In fact, this problem is a version of a famous *graph colouring* problem, related to the famous *four-colour theorem*. It was believed that for any *planar graph*, which is a graph that can be drawn on a piece of paper without crossing edges, that at most four colours were needed to colour each vertex so that no edge joined two vertices with the same colour. To prove this, computer scientists tested over 1800 different configurations. This was the first use of an *exhaustive computer assisted* proof in mathematics.

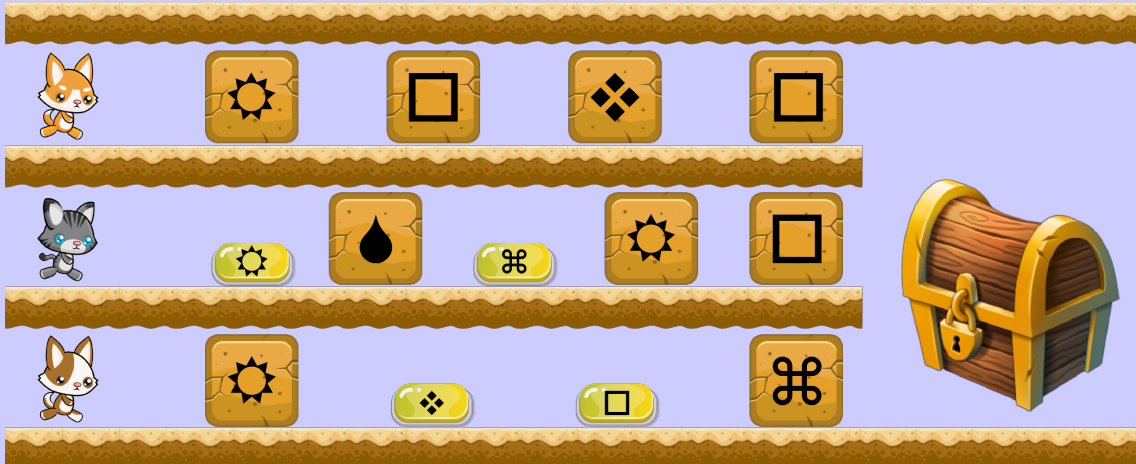
Country of Original Author

Japan




Treasure Hunt

Story



Three explorers are working together to find a hidden treasure chest. They each take a different path (upper, middle, or lower) and they explore their paths by running from left to right.

There are several large obstacles  blocking their paths.

When an explorer encounters an obstacle, they must wait until it is crumbled before they can proceed. An obstacle is crumbled when one of the explorers steps on a stone  that is marked with the same symbol as the obstacle. In fact, stepping on a stone crumbles *all* obstacles that are marked with the same symbol as the stone.

The obstacles, stones, and explorers are arranged as shown.

Question

Which explorer can get to the treasure chest?

- (A)  (B)  (C)  (D) No explorer can get to the treasure chest.

Answer

(A)



Explanation of Answer

We will name the explorers *A*, *B*, and *C* from top to bottom.

Explorer *B* cannot get to the treasure chest because there is no stone marked with a rain drop. This means Explorer *B* will never get past the first obstacle on their path.

Explorer *C* will get past the first obstacle on their path, but cannot get past the last obstacle. This is because the stone marked with the symbol on this last obstacle is on Explorer *B*'s path *after* the obstacle marked with a rain drop. Since Explorer *B* cannot get past the obstacle marked with a rain drop, the last obstacle on Explorer *C*'s path will never be crumbled.

Explorer *A* *will* be able to reach the treasure since all obstacles on their path can be crumbled. The first obstacle, which is marked with a sun, will be crumbled by Explorer *B*. Crumbling the obstacles marked with a sun will allow Explorer *C* to go through the first obstacle on their path and then step on the two stones that come next. These stones will crumble all the remaining obstacles on Explorer *A*'s path.

Connections to Computer Science

The three explorers in this task can be thought of as *processing units* working in parallel using a mechanism of locks to share resources. Each processing unit is able to *release* locks, such as when an explorer steps on a stone, and also *pause execution* until a lock is released, such as when an explorer encounters an obstacle.

In the last decade, increasing the speed of a processor has become difficult due to the physical limits of technology. Processor developers addressed this difficulty by putting several processors, or *cores* in a single computer. Most current commercially available processors are either dual- or quad-core processors.

To take advantage of multiple processors and make programs run faster, computer scientists have to find ways to split computation into separate parts that can run in *parallel*. Sometimes the processes need to wait for each other before continuing their execution. Locks and *interrupt signals* are used to manage this.

Country of Original Author

Pakistan



Water Bottles

Story

Dani is required to entirely fill as many empty water bottles as possible using a 50 litre tank.

Suppose she is given the following 10 empty bottles where each bottle is labelled with the number of litres it can hold.



Question

What is the maximum number of bottles that Dani can fill entirely?

- (A) 4
- (B) 7
- (C) 8
- (D) 10

Answer

(B) 7

Explanation of Answer

Dani can approach this task by first arranging the bottles from smallest to largest (in terms of how many litres they can hold).



At any point, if Dani has two empty bottles that can hold different amounts of water, there is no advantage to filling the larger bottle. So Dani can now fill the bottles one by one from smallest to largest. The smallest 7 bottles can be filled entirely using $3 + 4 + 5 + 6 + 7 + 8 + 9 = 42$ litres of water. The tank will have 8 litres left but all the remaining bottles can hold more than 8 litres, so no more bottles can be filled entirely.



Connections to Computer Science

This problem can be solved using a *greedy strategy*. For this problem, we can fill the bottles in sorted order according to the amount of water each bottle holds.

More generally, a greedy strategy involves making the best choice at each stage in the hopes of finding the best solution. In this way, it makes one greedy choice after another, reducing the given problem into a smaller one.

However, not every problem can be solved optimally using a greedy strategy. For example, *knapsack problems* usually do not have optimal solutions that can be found using a greedy strategy. Nevertheless, the greedy strategy is still useful because it is easy to describe and implement and often gives a good approximation to the optimal solution.

Country of Original Author

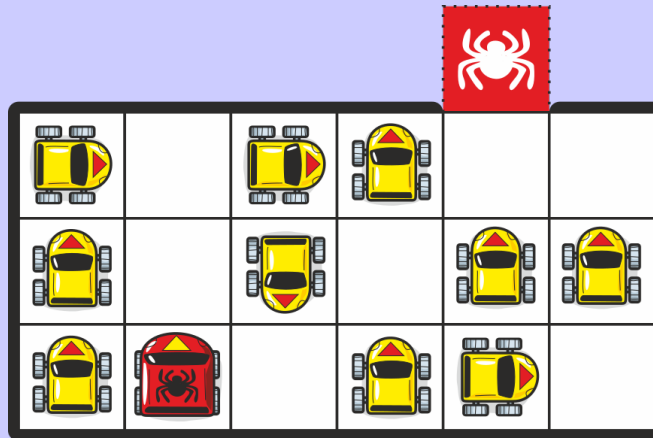
Thailand



Part C

Spider Car

Story



In the fenced area shown there are yellow cars and a single red spider car. Ayo is trying to get the spider car in the spider square just outside the fenced area.

In one *move*, Ayo can:

- drive one car forward one square,
- reverse one car backwards one square,
- rotate one car left (90 degrees) in its current square, or
- rotate one car right (90 degrees) in its current square.

There can only be one car per square at any given time and only the spider car can be moved into the spider square.

Question

What is the minimum number of moves Ayo needs to get the spider car in the spider square?

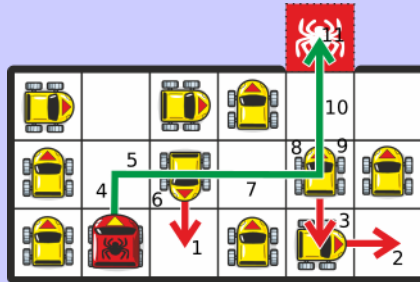
- (A) 9
- (B) 11
- (C) 13
- (D) 15

Answer

(B) 11

Explanation of Answer

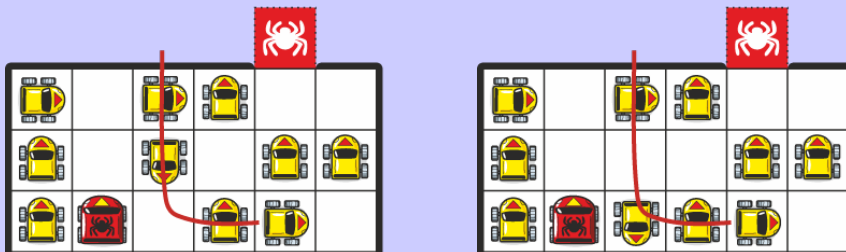
Ayo can get the spider car in the spider square in 11 moves as shown:



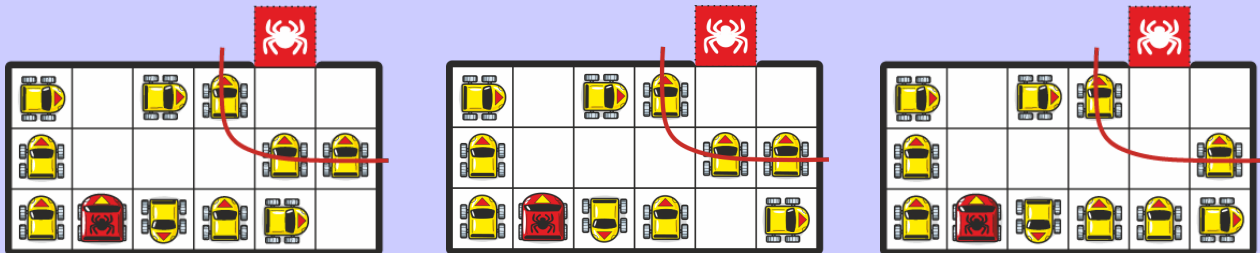
It now remains to prove that 11 moves is the minimum number of moves needed.

Suppose the spider car was the *only* car in the fenced area. To reach the spider square it must move up 3 times, right 3 times, and turn 2 times. These moves can be done in a variety of different combinations, but the spider car will always require these 8 moves. Of course, the spider car is *not* the only car in the fenced area and so more moves are needed to unblock its path.

First we need to create a way through the indicated “L-shaped” barricade. This can be done using one move as shown:



Now we need to create a way through the second “L-shaped” barricade. This cannot be done using one move (without blocking the exit). Thus, it needs at least two moves as shown:



Therefore, the minimum number of moves needed is $8 + 1 + 2 = 11$.

Connections to Computer Science

Proving a solution is minimal (or optimal) can be a very difficult thing to do. One way to do it is to search through all the possibilities. Since there are not too many cars, it is possible to examine each car and determine which other cars need to move so that it can exit. This is called an *exhaustive search* or the use of a *brute-force algorithm*.

Sometimes the number of possible solutions to a problem is so large that, even using a computer, exhaustive search takes too long. When this happens, there may be other computer science algorithms that can be applied in order to optimize the search. *Branch and bound* and *greedy algorithms* are two examples of these optimization techniques.

Country of Original Author

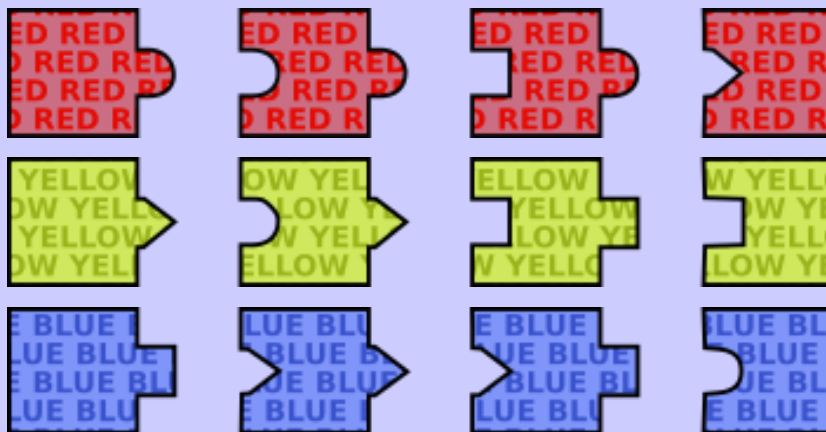
Lithuania



Puzzle Pieces

Story

A beaver has a puzzle with 12 different types of pieces, 4 of which are red, 4 of which are yellow, and 4 of which are blue, as shown below. There is an unlimited number of each type of piece.



Using these pieces, the beaver can create various colour sequences. The first piece in a sequence must have a flat left side and the last piece must have a flat right side. Pieces join in the usual way but two pieces can't be joined on their flat sides and pieces can't be rotated. One possible sequence is shown below.



Question

Which of the following colour sequences **cannot** be constructed?

- (A) YELLOW → BLUE → BLUE → RED → BLUE
- (B) BLUE → YELLOW → RED → YELLOW → RED
- (C) RED → RED → YELLOW → BLUE → BLUE
- (D) BLUE → RED → YELLOW → BLUE → RED

Answer

(C) RED → RED → YELLOW → BLUE → BLUE

Explanation of Answer

For Option C, notice that the last puzzle piece must have a flat right side and be blue. Thus, this piece must be:



The second last piece must have a rounded tab on its right side, and also be blue. There are no blue pieces which have a rounded tab on their right side, they only have square or triangular tabs. Therefore, the sequence in Option C cannot be constructed.

For the sequences in Options A, B and D, we can construct them as follows:

- Option A: YELLOW → BLUE → BLUE → RED → BLUE



- Option B: BLUE → YELLOW → RED → YELLOW → RED



- Option D: BLUE → RED → YELLOW → BLUE → RED



Connections to Computer Science

Computer scientists call valid expressions *well-formed*. Expressions containing errors are called *malformed*. In this task, you were asked to find the malformed expression.

As another example, consider mathematical expressions that involve arithmetic. There are rules about where operators and operands must be placed. The expressions $2 + 3$, 4×2 , and $(2 + 4) \div 3$ are well-formed but $2 + \times 5$ and $4 \times$ are malformed.

An expression that is well-formed can also be called *syntactically correct*. The *syntax* of a statement is the correct form or structure: for example, the sentence “My dog likes walks” is syntactically correct, but the sentence “Likes walks dog my” is not syntactically correct, since the form or order of the words is not correct. Most programming languages in computer science require that programs are syntactically correct.

In fact, computers are much more strict about syntax since computers do not have the same ability as humans to *infer meaning* from malformed sentences. For example, most English speakers could understand the meaning of “The stairs up I walked”, but an equivalently scrambled statement in a *computer programming language* would be rejected by the computer as a *syntax error*.

Country of Original Author

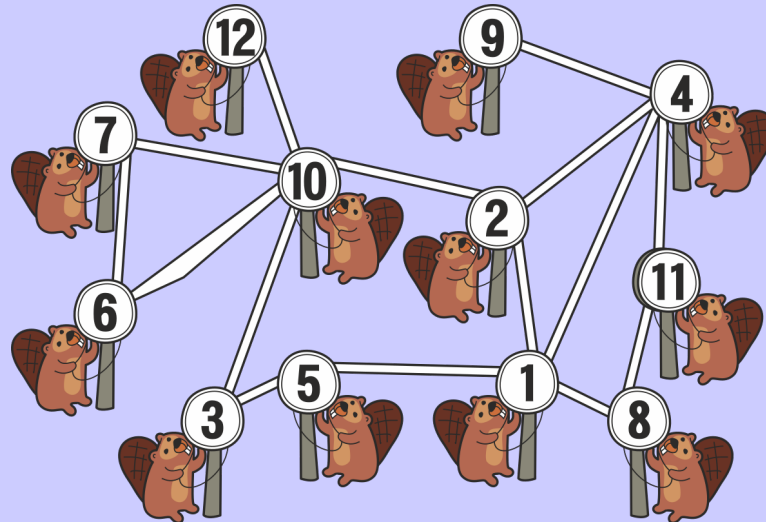
Macedonia



Spreading the News

Story

Twelve beavers share news with each other using a network of wires as shown:



Two beavers can share news if they are directly connected by a wire. For example, beaver 6 is directly connected to beavers 7 and 10 but not to beaver 3.

All beavers want to hear news as quickly as possible. As such, a beaver with news uses all of its wires simultaneously to inform the other beavers that it is directly connected to.

For example, if beaver 8 has news it will inform beavers 1 and 11 right away. Next, beavers 1 and 11 will further spread the news, at the same time, to beavers 2, 4, and 5. The beavers continue to spread the news using their wires until all beavers have been informed.

Question

If there is news that should be spread as quickly as possible, which beaver should be informed of the news first?

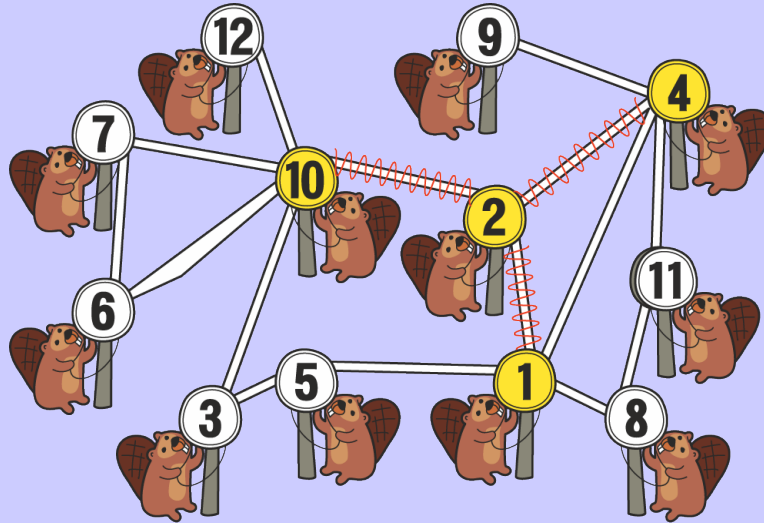
- (A) 1
- (B) 2
- (C) 4
- (D) 10

Answer

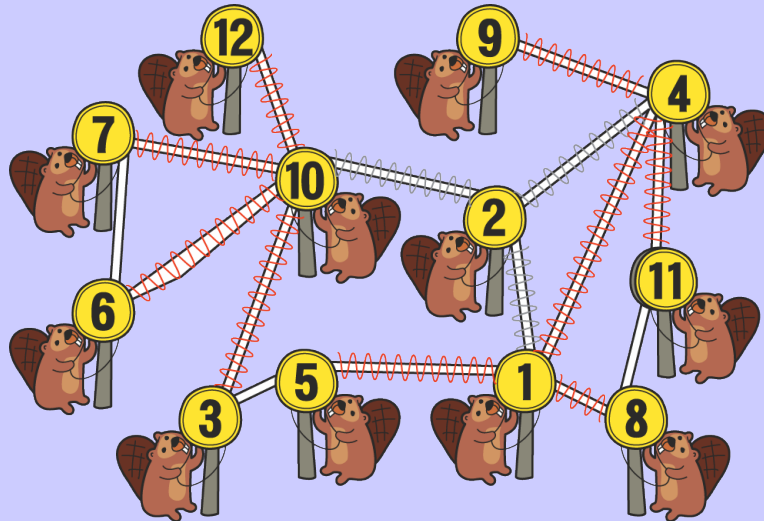
(B) 2

Explanation of Answer

If beaver 2 is informed of the news first, it will take exactly 2 steps to inform all of the other beavers. In the first step, beavers 1, 4, and 10 are informed.



Those three beavers inform all of the other beavers in the second step.



This means that all of the beavers are at most two wires away from beaver 2. Among the twelve beavers, there is no other beaver that is also at most two wires away from all others. In addition, no beaver is exactly one wire away from all others. Therefore, informing beaver 2 ensures that news is spread as quickly as possible.

Connections to Computer Science

Many real-life problems are solved by computer programs that represent the underlying problem as a *graph*. A graph consists of a set of *vertices* or *nodes* (usually depicted as points) and a set of *edges* (usually depicted as line segments), that connect these vertices.

In this task, the underlying problem is to find the *graph centre*. The centre of a graph is the vertex u , where the largest distance between u and any other vertex is as small as possible. In general, there can be more than one graph centre, but in this particular problem, there was only one such vertex, which is beaver 2.

Finding the centre of a graph is useful in *facility location problems*, where the goal is to minimize the worst-case distance to the facility. For example, placing a hospital at a central point reduces the longest distance an ambulance has to travel. If we have a small graph, such as in this particular task, we can use trial and error, as it is simple to calculate the distance between all pairs of vertices. For large problems, the centre can be found using more sophisticated algorithms like the *Floyd-Warshall algorithm*.

Country of Original Author

Macedonia

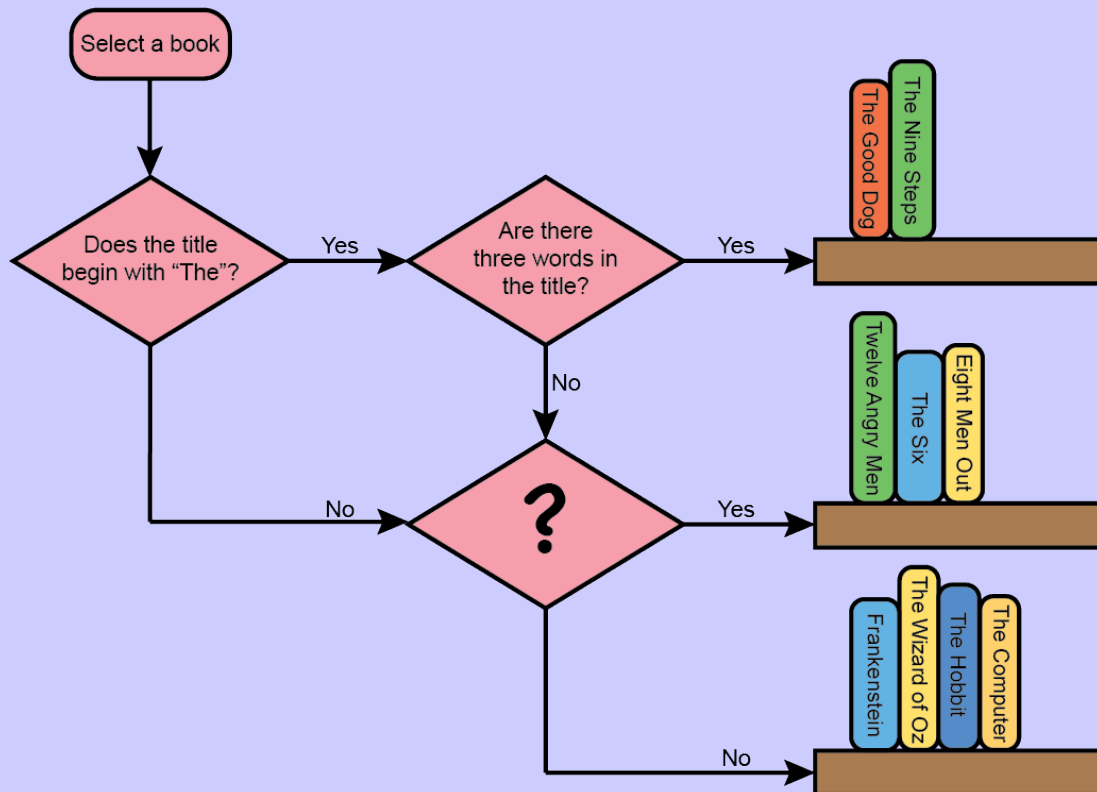


Book Organizer

Story

Bora uses a flowchart to organize her books onto three shelves. When she gets a new book, she starts at the top of the flowchart and follows its instructions to determine on which shelf the book belongs.

Each diamond in the flowchart includes a “yes” or “no” question about the book’s title. The answer determines which arrow leading away from the diamond Bora will follow. When an arrow points at a shelf, the book is added to that shelf. Otherwise, Bora continues to move through the flowchart.



Question

If Bora’s books end up on the shelves as shown, which of the following questions could have appeared in the diamond marked with a question mark (?) in the flowchart?

- (A) Does the title include the word “Men”?
- (B) Are there fewer than four words in the title?
- (C) Is the letter “i” in the title?
- (D) Does the title include a number?

Answer

(D) Does the title include a number?

Explanation of Answer

Consider the diamond marked with a question mark (?) and the books on the shelves at the end of the “yes” and “no” arrows leading away from this diamond.

Option A is incorrect because the book “The Six” does not include the word “Men” in its title but it is on the shelf at the end of the “yes” arrow.

Option B is incorrect because the books “Frankenstein”, “The Hobbit”, and “The Computer” have fewer than four words in their titles but they are on the shelf at the end of the “no” arrow.

Option C is incorrect because the book “Twelve Angry Men” does not have the letter “i” in its title but it is on the shelf at the end of the “yes” arrow.

Option D is correct because all three books on the shelf at the end of the “yes” arrow do include numbers in their titles and none of the four books on the shelf at the end of the “no” arrow include numbers in their titles.

Connections to Computer Science

A *flowchart* is a type of diagram that represents an *algorithm*, *workflow* or *process*. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. These boxes capture the three main concepts in any algorithm: *sequence*, *selection*, and *repetition*. The sequence actions are drawn in rounded rectangles and capture the concept of doing one action followed by another. The selections (or decisions) are drawn in diamond shapes and capture the concept of doing one of two possible actions, based on whether some decision is true or false. The repetition action is illustrated by having an arrow return to an earlier part of the flowchart, capturing the concept of repeating some actions, usually until a certain condition is met. These three basic operations describe all algorithms in *imperative programming languages*.

Flowcharts are also commonly used during the planning stages of code writing. By using a flowchart to plan out how a program should behave, programmers can agree on its behaviour and often find and fix errors in logic before coding even begins.

Country of Original Author

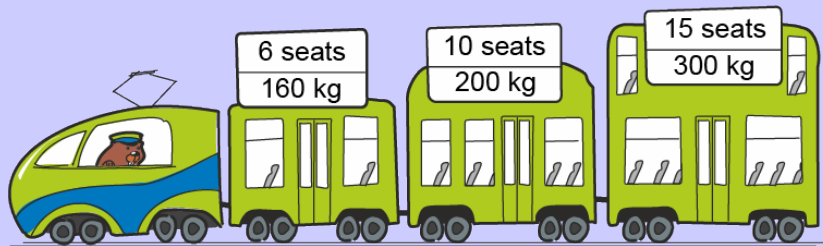
New Zealand



Train Trip

Story

A train has three carriages, with the number of available seats and luggage limits as shown:



Eight beaver families would like to go on a train trip, but

- every beaver must sit on its own seat,
- if one member of a family sits in a carriage, then all members of that same family must sit in that same carriage,
- a family's luggage must be in the same carriage as the family, and
- the total luggage weight has to be within the limits of each carriage.

Details about each family and their luggage are given in the following table:

Family	Number of Members	Luggage Weight (kg)
Avsec	3	50
Bizjak	4	80
Cerar	5	110
Dolenc	4	80
Erjavec	2	40
Furlan	3	70
Gabric	6	130
Hacin	5	100

Question

What is the maximum number of families that can go on the trip?

- (A) 5
- (B) 6
- (C) 7
- (D) 8

Answer

(C) 7

Explanation of Answer

The train has 31 seats but there are 32 individual beavers that want to go on the trip. Therefore, it is not possible for all 8 families to go.

It is possible for 7 families to go on the trip. One way 7 families can ride the train while meeting all the constraints is as follows:

Carriage	Families	Total Seats	Total Luggage Weight (kg)
First	Gabric	6	130
Second	Avsec	3	50
	Cerar	+ 5	+ 110
	Erjavec	+ 2 = 10	+ 40 = 200
Third	Bizjak	4	80
	Dolenc	+ 4	+ 80
	Hacin	+ 5 = 13	+ 100 = 260

Connections to Computer Science

This problem is a variation of a *knapsack problem*: given a set of items, each with a weight and a value, determine the maximum value that can be put in the knapsack without exceeding the knapsack's weight limit. This problem is one of many *combinatorial optimization* problems.

In general, knapsack problems are *NP-complete*, which means that there are no known efficient solutions, and most computer scientists believe that there will never be any efficient solutions to NP-complete problems. The most efficient algorithms known to completely solve these problems require *exponential running time*, which means that for even small input sizes, the running time could be years before a solution is found.

Notice that the *greedy algorithm* does not yield an optimal solution. In this problem, taking either the smallest families, or the least luggage weight, will not produce an optimal solution.

For this smaller problem, rather than the general case, we can solve it with careful choices and some *trial and error*.

Country of Original Author

Slovenia

