UNIVERSITY OF WATERLOO

The CENTRE for EDUCATION in MATHEMATICS and COMPUTING

2020 Beaver Computing Challenge (Grade 9 & 10)

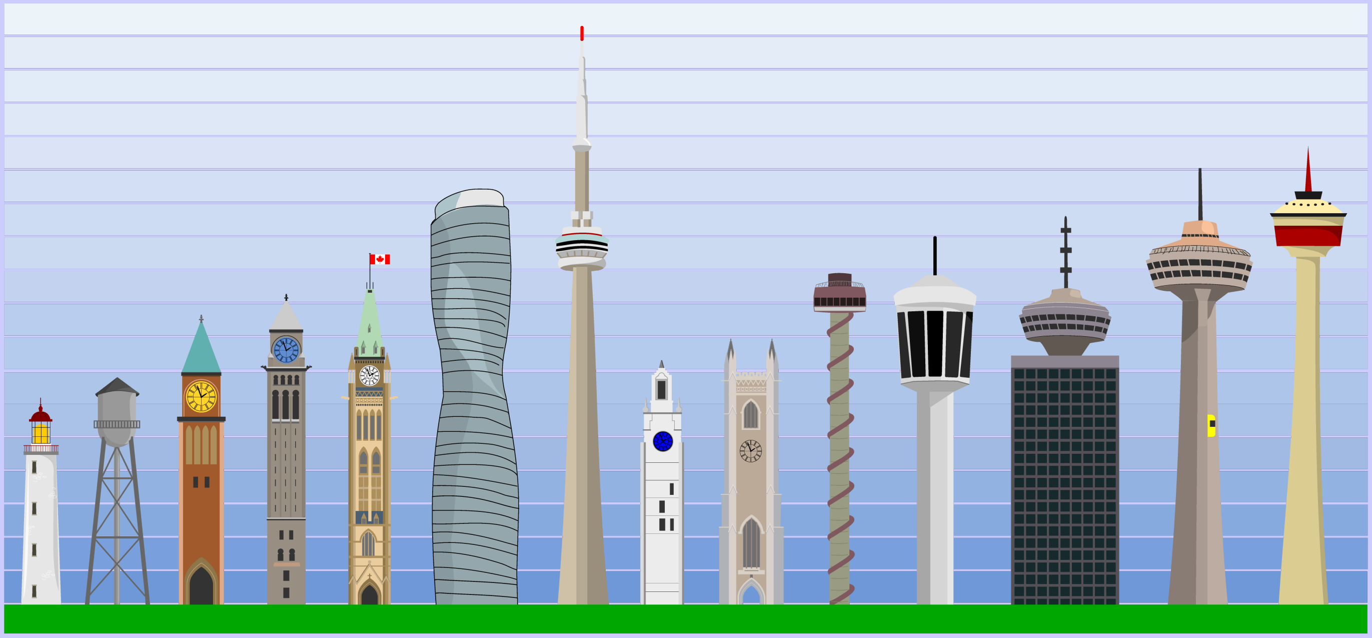
Questions, Answers, Explanations, and Connections

Part A

Skyline

Story

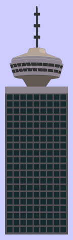
A skyline consists of 14 towers as shown.

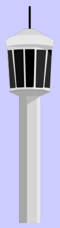


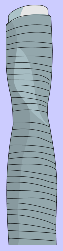
The height of a tower is measured from the bottom of its base to its highest point, including any flagpoles or antennas.

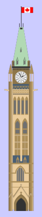
Question

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

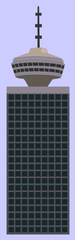
(A) 

(B) 

(C) 

(D) 

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.

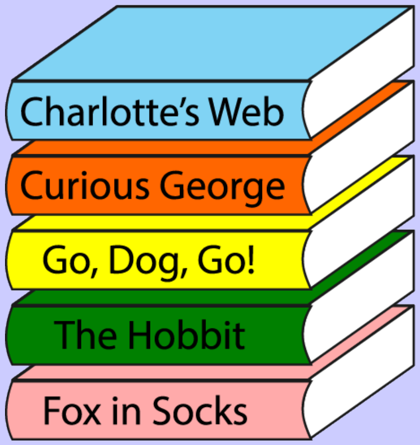
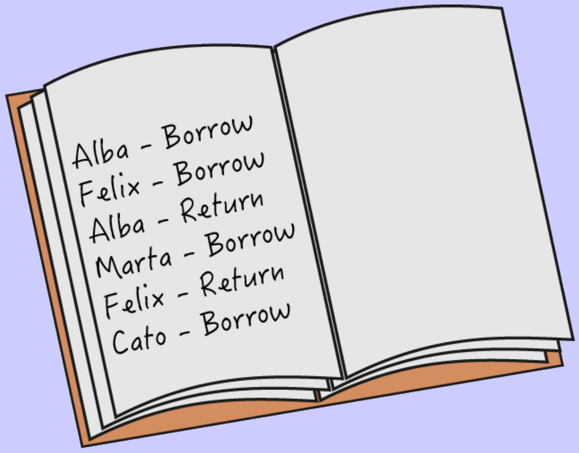
Country of Original Author

Canada

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:
	

Charlotte's Web
Curious George
Go, Dog, Go!
The Hobbit
Fox in Socks

Alba - Borrow
Felix - Borrow
Alba - Return
Marta - Borrow
Felix - Return
Cato - Borrow

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.

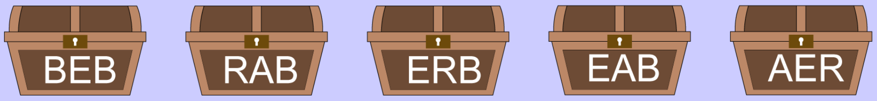
Country of Original Author

Canada

Locked Chests

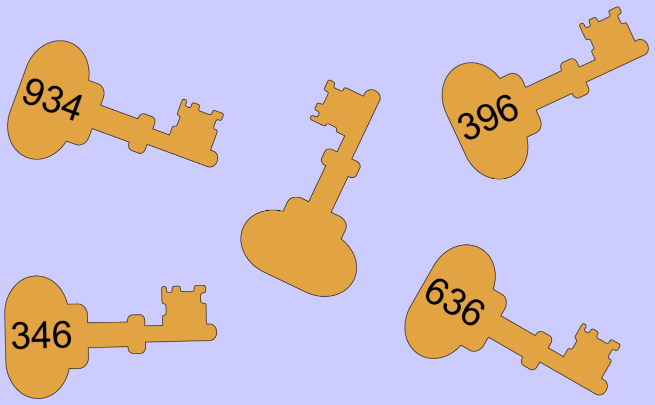
Story

Five different chests are engraved with letters as shown:



Each chest has a key labelled with digits corresponding to the chest's engraved letters. Each digit always corresponds to the same letter.

The keys fell on the floor and one label was lost:



Question

What is the lost label?

(A) 496

(B) 639

(C) 436

(D) 649

Answer

(A) 496

Explanation of Answer

Chest BEB is the only one where the first and the third letter match. So, this chest matches key 636, which is the only key where the first and third digit match. So we know that the letter B corresponds to the digit 6 and the letter E to the digit 3.

Letter	A	B	E	R
Digit		6	3	

Chest AER is the only one that ends with a letter other than B. So, this chest matches key 934, which is the only key that ends with a digit other than 6. Thus, the letter A corresponds to the digit 9 and the letter R to the digit 4.

Letter	A	B	E	R
Digit	9	6	3	4

Now that we know all the letter-digit pairings, we can match the remaining chests to keys, and discover that chest RAB matches key 496, which is missing.

Connections to Computer Science

Cryptography is an area of study that lies at the intersection of computer science and mathematics. The mapping between digits and letters 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 – a letter always matches another letter and each letter can be used only in one pair. During *encryption* of 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

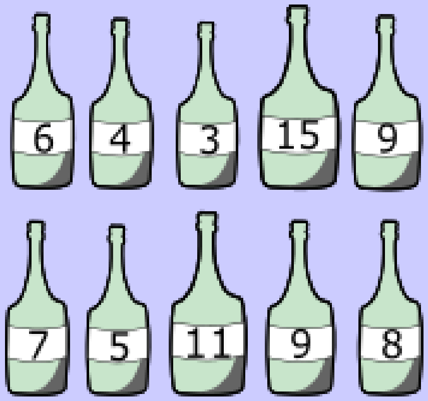
Cyprus

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).

3
4
5
6
7
8
9
9
11
15

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

Ancient Texts

Story

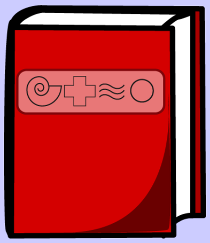
Symbols form the titles of ancient texts. Each type of symbol is associated with a digit as shown below. Some different symbols are associated with the same digit.

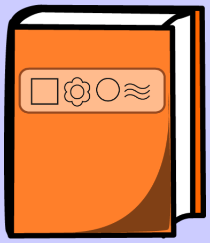
symbol	□	△	☆	○	◎	◇	♡	☁	✚	✿	☾	🐟	⛰	≋	💧	🌀
digit	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6

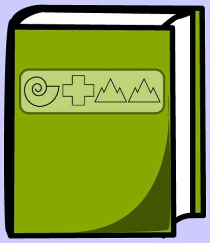
The *special number* of a text is the sequence of digits associated with the symbols in the title of the text (in order). For example, 56432 is the special number of the text with the title ◎◇○☆△.

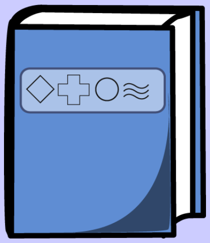
Question

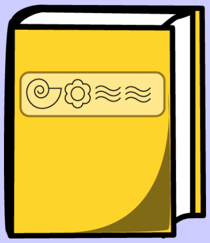
Which of the following texts has the same special number as the red text below?



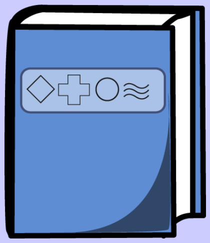
(A) 

(B) 

(C) 

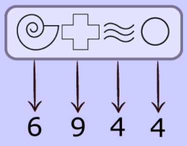
(D) 

Answer

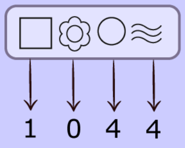
(C) 

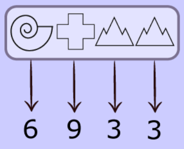
Explanation of Answer

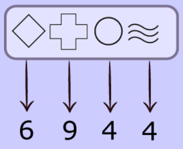
First, we use the given associations to determine the special number of the red text:

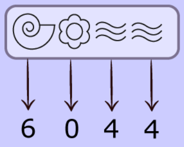


Then, we determine the special numbers for each of the texts in the answer options and determine which text has the special number 6944.

(A) 

(B) 

(C) 

(D) 

Of the options, the only text that has the special number 6944 is the one in Option C.

Connections to Computer Science

When storing information, one important goal is to be able to quickly *retrieve* a particular piece of information. One method to accomplish this goal is to store information in a *hash table*.

A *hash table* consists of several locations where we can store information. The position of where to store a piece of information is determined by a *hash function*. For this task, the hash function for the symbols, which were the *keys*, was to assign them to one of the digits from 0 to 9, the *values*.

As was also realized in solving this task, sometimes there are *collisions* in the hash function, where two keys yield the same value. In order to deal with this, various *collision resolution strategies* have been developed to maintain efficiency. Hashing can provide extremely efficient lookup times even with a massively large amount of data.

Country of Original Author

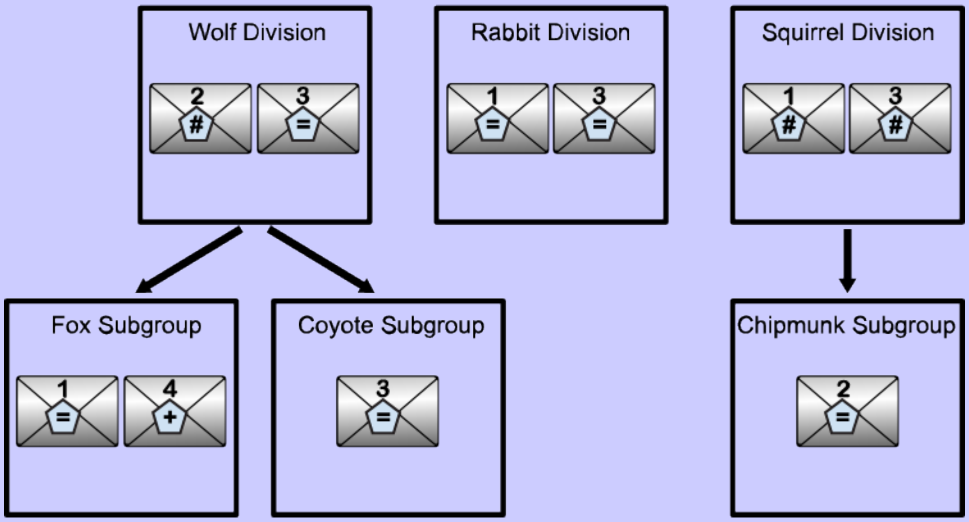
Taiwan

Part B

Beaver Intelligence Agency

Story

For security reasons, a secret message was broken into four parts (1, 2, 3, and 4). Copies of these parts were then sent to the divisions and subgroups of the Beaver Intelligence Agency (BIA) as shown:



Labels on the copies of the message parts indicate who has access to it:

- "+" means everyone in the BIA has access to this copy.
- "#" means the division that receives it and the division's subgroups (indicated by arrows) have access.
- "=" means only the division or subgroup that receives it has access to this copy.

Question

Which one of the following has access to all four parts of the message?

(A) Fox Subgroup

(B) Wolf Division

(C) Rabbit Division

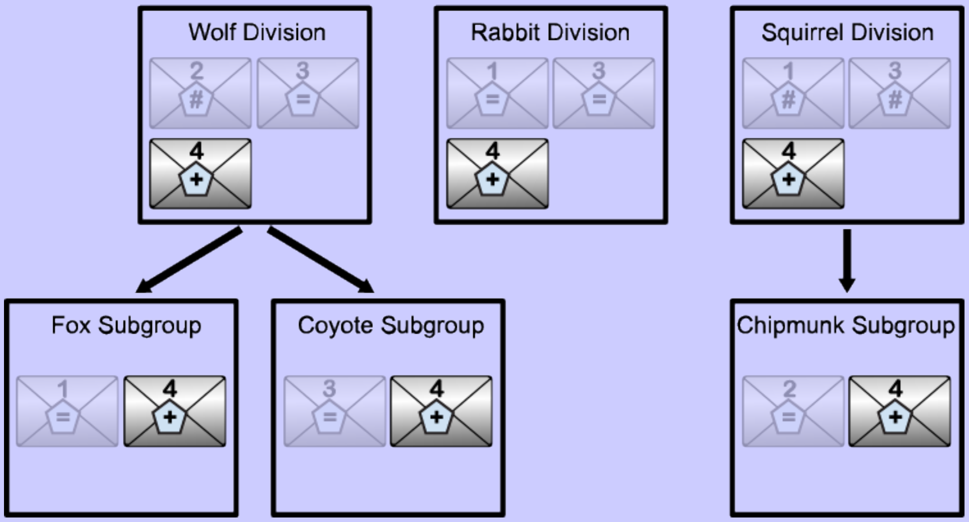
(D) Chipmunk Subgroup

Answer

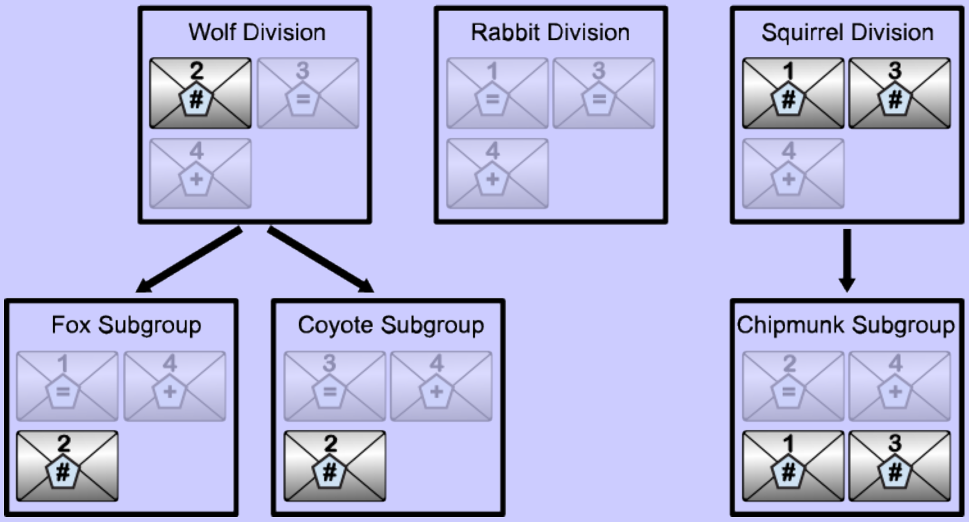
(D) Chipmunk Subgroup

Explanation of Answer

Parts labelled with "+" are accessible to everyone. We can visualize this by cloning each part labelled with "+" and placing a clone within every division and subgroup. The Fox Subgroup holds part 4 labelled with "+". Here is what the BIA looks like after cloning this part:



Parts labelled with "#" are accessible only to the receiving division and its subgroups. We can clone each part labelled with "#" and place a clone within each receiving division's subgroups. The Wolf Division holds part 2 labelled with "#" so the Fox and Coyote Subgroups receive a clone of part 2. The Squirrel Division holds parts 1 and 3 labelled with "#" so the Chipmunk Subgroup receives a clone of parts 1 and 3:



Parts labelled with "=" are only accessible to the receiving division or subgroup so we don't make clones of parts labelled with "=". Looking at the BIA after all the clones have been made, we see that only the Chipmunk Subgroup has access to all four parts of the message.

Connections to Computer Science

There are various *programming paradigms* that different programming languages follow. For example, there are *imperative programming languages* (such as C++ or Python), *functional programming languages* (like Scheme), and *object-oriented programming languages* (like Java). In object-oriented programming (OOP), an *object* is a way of storing both *data* and particular *operations* to access or transform that data. Most object oriented programming languages are *class-based*, meaning that objects are *instances* of classes.

In this task, the divisions and subgroups represent classes. Every class has access to the relevant objects with the corresponding type of visibility. *Public* objects (those marked with a +) can be accessed from every class in the program. *Private* objects (those marked with a "=") are only visible and changeable by the class itself. *Protected* objects (#) can be accessed by the class itself and by its subclasses. In this task, the subgroups can read "#" messages from their division leader.

Different types of visibility are very useful for organizing a program. A class does not have to deal with objects and properties from other classes, and access to information can be carefully controlled. In this way, some programming errors can be avoided.

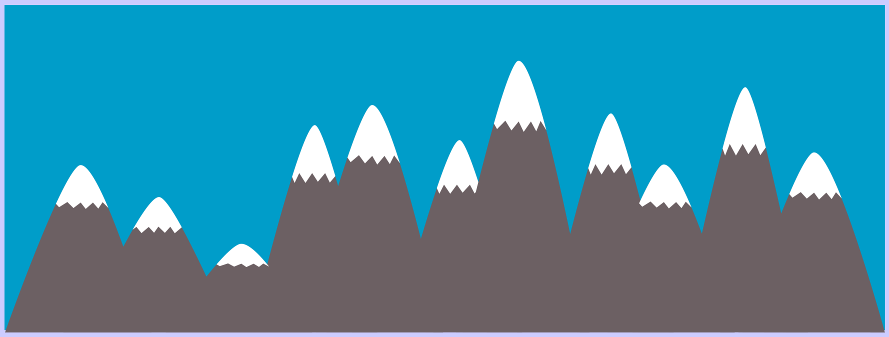
Country of Original Author

Austria

Mountain Climber

Story

Binsa is climbing in the mountain range shown which has 11 peaks each of a different height.



Binsa climbs by starting at the top of a random peak, then looking left and right. If she sees a peak immediately beside her that is higher than the one she is currently on, she climbs to the top of this higher peak. If two neighbouring peaks are both higher, she climbs to the top of the higher one. She continues to do this until there is no higher peak immediately beside her.

Question

From how many of the peaks (including the highest peak) will Binsa reach the highest peak?

(A) 3

(B) 4

(C) 6

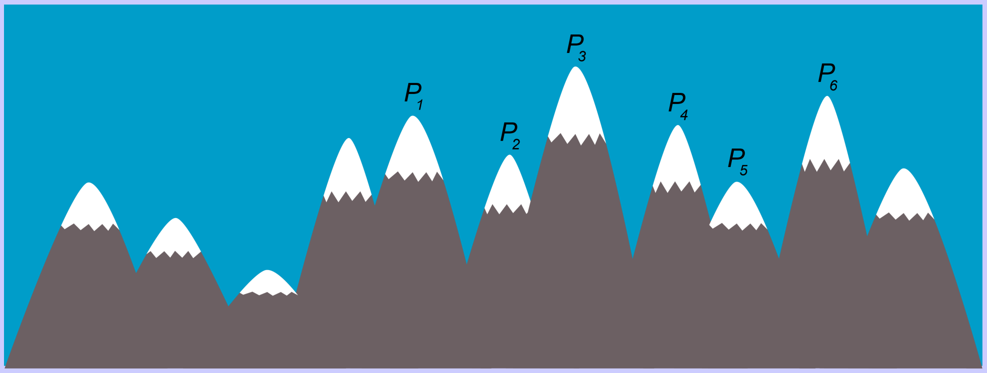
(D) 7

Answer

(A) 3

Explanation of Answer

We label the peaks as shown.



Notice that the highest peak is P_3.

If Binsa starts on P_1 she will not climb any further since the peak on her left and the peak on her right are both lower. She will not reach the highest peak from here. In addition, if she starts on any peak to the left of P_1 she will not reach the highest peak since she will not be able to climb past P_1.

If Binsa starts on P_2, both of the neighbouring peaks are higher so she will climb the higher one, P_3, and reach the highest peak.

If Binsa starts on P_3, she has reached the highest peak.

If Binsa starts on P_4, she will climb to P_3 and reach the highest peak.

If Binsa starts on P_5, both of the neighbouring peaks are higher so she will climb the higher one, P_6.

From P_6, Binsa will not climb any further since the peak on her left and the peak on her right are both lower. She will not reach the highest peak from here. In addition, if she starts on any peak to the right of P_6 she will not reach the highest peak since she will not be able to climb past P_6.

Thus, there are exactly 3 peaks from which Binsa will reach the highest peak. They are P_2, P_3, and P_4.

Connections to Computer Science

The process that Binsa follows is called a *greedy algorithm*. The key idea in a greedy algorithm is to make a choice at each step based on "local information", regardless of whether there is a better option that might yield a better solution. In this task, even though Binsa is trying to reach the highest peak overall, she decides what to do next only by looking at the two mountains right beside her. That is, instead of trying to look for the *global maximum*, instead she looks for a *local maximum*.

Greedy algorithms do not always correctly solve problems, but when they do, they usually do so very efficiently. For example, the *change making problem* yields a fast greedy solution: in most currency systems, making a certain amount of change can be done by repeatedly using a coin of as large a value as possible. Stopping this process when the desired total is reached, will always minimize the number of coins used.

Unfortunately, not every greedy algorithm gives the best answer. However, even if a greedy algorithm is incorrect, sometimes they can be used to find a solution that is "good enough". This is called an *approximation algorithm*.

A specific type of greedy algorithm used in artificial intelligence is called the *hill-climbing algorithm*. The name of this technique comes from the idea behind this task. Additionally, the story behind this task is related to an idea from geography called *topographic isolation*. One of the great things about studying computer science is how it can be applied to almost any other subject area.

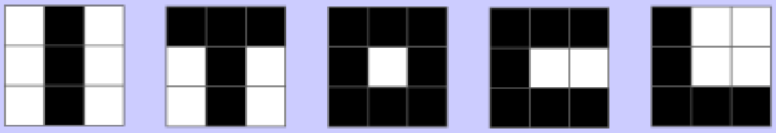
Country of Original Author

Canada

Image Scanner

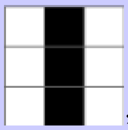
Story

The following five images represent the letters I, T, O, C and L, respectively. Each image is a 3-by-3 grid made up of nine pixels that are each black or white.



When a machine scans an image, instead of recording black or white at a pixel, it records how many of the other four images have the same shade (black or white) at that pixel.

For example, when scanning the image



, the machine records

0	3	1
1	1	3
1	4	1

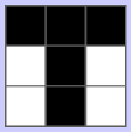
.

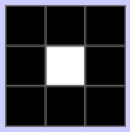
Question

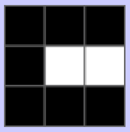
If the machine records

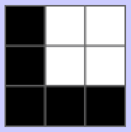
3	3	2
2	2	0
2	4	2

, what image did it scan?

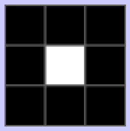
(A) 

(B) 

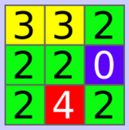
(C) 

(D) 

Answer

(B) 

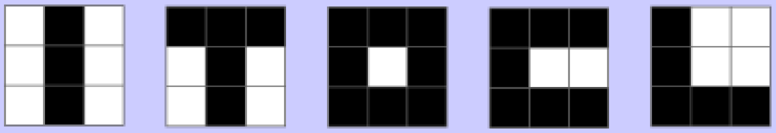
Explanation of Answer

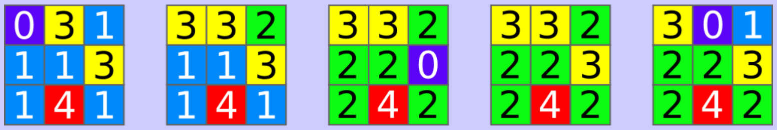
In the given image recording, 

the number 0 in the rightmost pixel in the middle row indicates that no other images have this shade at this pixel.

We can see that the images of the letters I, T, C, and L all have a white pixel in this location, but the image of the letter O has a black pixel. Since the image of the letter O has a unique shade at this pixel, it follows that the image scanned by the machine must be the image of the letter O.

Here are all images with their machine recording.





Connections to Computer Science

A *heat map* is a graphical representation of data where the individual values contained in a two-dimensional grid are represented as colours. One common use of maps is in weather reports, indicating temperatures in different regions: the hotter the temperature, the more intense or redder the colour is.

In computer science, heat maps are used in *image recognition* or other *machine learning* algorithms, to highlight important areas of the data.

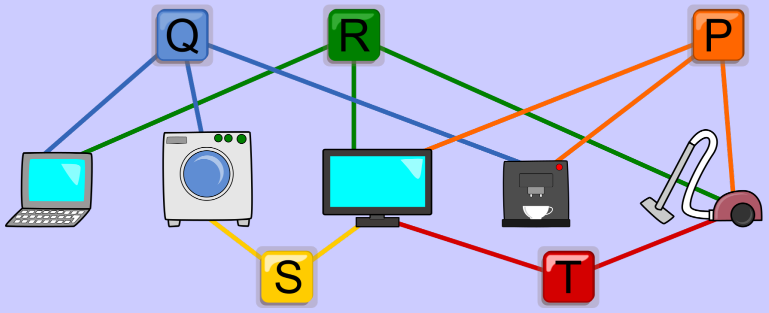
Country of Original Author

Germany

Household Appliances

Story

As a practical joke, someone has connected appliances to buttons P, Q, R, S, and T in a very strange way. Pressing a button toggles the on/off state of each appliance it is connected to. For example, pressing button T will turn the vacuum cleaner on if it is off and off if it is on. Pressing button T will also turn the television on if it is off and off if it is on.



All of the appliances are off.

You want only the television and coffee machine on (the third and fourth appliances from the left in the picture).

Question

Which of the following sequences of buttons should you press?

(A) T, R, Q, P

(B) R, Q, P, S

(C) S, P, T, R

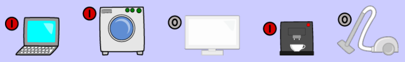
(D) Q, S, R, T

Answer

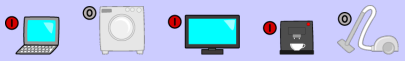
(D) Q, S, R, T

Explanation of Answer

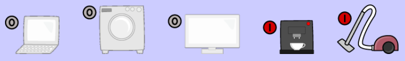
For the sequence in Option D, the buttons Q, S, R, and T are pushed. When Q is pushed, the laptop, washing machine, and coffee machine are turned on.



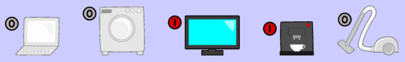
Next, S is pushed, the washing machine is turned off, and the television is turned on.



Next, R is pushed, the laptop and television are turned off, and the vacuum cleaner is turned on.



Finally, when T is pushed, the television is turned on and the vacuum cleaner is turned off.



At this point, the television and coffee machine are on, whereas all other appliances are off.

Another way to see that this is the correct answer while the others are not correct is to examine each button and appliance independently. Notice that if a button is pressed an odd number of times, the connected appliances are left in the "on" state, while an even number of presses leaves the appliances in the "off" state. The same holds for a single appliance: if the number of buttons pressed and connected to it is even, then the appliance will be turned off, and if it is odd, the appliance will be turned on.

- For the Option A (T, R, Q, P), the vacuum cleaner is connected to three of the pressed buttons: T, R, P. Hence, it will be "on" after all these buttons are pressed, which is not correct.
- For Option B (R, Q, P, S), the coffee machine is only connected to two of the pressed buttons, P and Q, which means that it will remain "off", which is not correct.
- For Option C (S, P, T, R), the washing machine is connected to only one of the pressed buttons, S, which means that it will be "on" after all these buttons are pressed, which is not correct.
- For Option D (Q, S, R, T), we have the following number of button presses for the appliances:
  - laptop: two button presses, Q and R, which means it will be off at the end.
  - washing machine: two button presses, Q and S, which means it will be off at the end.
  - television: three button presses, R, S, and T, which means it will be on at the end.
  - coffee machine: one button press, Q, which means it will be on at the end.
  - vacuum cleaner: two button presses, R and T, which means it will be off at the end.

Connections to Computer Science

We can think of the appliances in this task in terms of individual *binary digits*, also called *bits* (from BInary digiTS). Each time a button is pressed that affects an appliance, it switches between off and on, which computer scientists usually write as 0 and 1, respectively.

The binary system is used by computers, where all information is stored and manipulated as binary information. In particular, the computer hardware, such as the *central processing unit* (or *CPU*) operates with information as electrical signals, which are either "off" or "on". These electrical signals are processed through *logic gates*, such as AND, OR, and NOT gates. One other logic gate is the XOR, or *exclusive-OR* gate, usually written as $\oplus$. It works in the following way:

$$0 \oplus 0 = 0 \quad 1 \oplus 0 = 1 \quad 0 \oplus 1 = 1 \quad 1 \oplus 1 = 0$$

Notice that the only way that a value will be "on" is if exactly one of the two arguments to XOR is true.

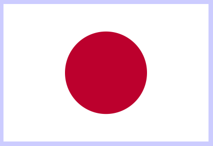
For this task, we can think of the applicances as five bits, and keep track of their *state* and how each of the buttons change the state. That is, initially, we can think of the applicances, as read from left to right, as the sequence 00000. After button Q is pressed, we know that the sequence would change to 11010. Pressing just button R would change the sequence to 10101. If we pressed both Q and R from the initial state, we would have

$$(00000 \oplus 11010) \oplus 10101 = 01111$$

Using this technique, we can determine the bit pattern for each button, XOR them, and determine which bits end up as 1 or 0, which indicates whether the appliance is on or off. Interestingly, the order in which we apply the XOR operations (button presses) does not matter.

Country of Original Author

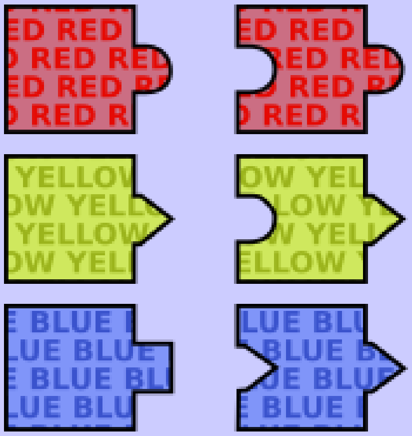
Japan



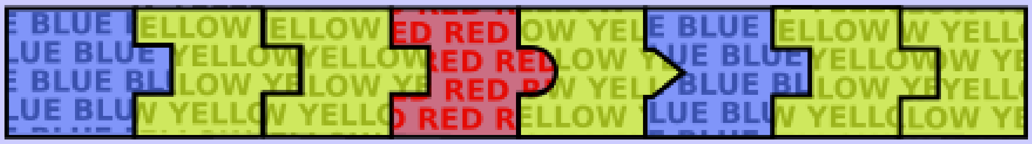
Puzzle Pieces

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

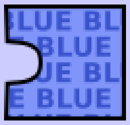
(D) BLUE → RED → YELLOW → BLUE → RED

Answer

(C) RED → RED → YELLOW → BLUE → BLUE

Explanation of Answer

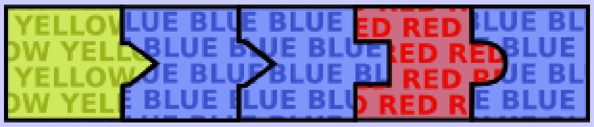
For Option C, notice that the last puzzle piece must have a flat right side and be blue. Thus, this piece must be:



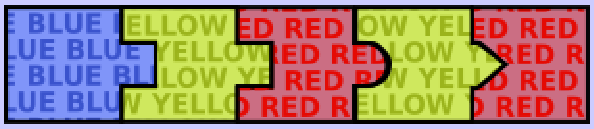
The second last piece must have a rounded tab on its right side, and also be blue. There are no blue pieces which have a rounded tab on their right side, they only have square or triangular tabs. Therefore, the sequence in Option C cannot be constructed.

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

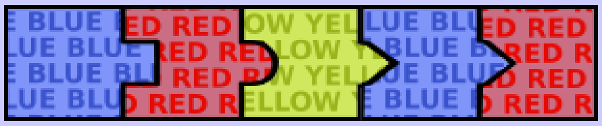
- Option A: YELLOW → BLUE → BLUE → RED → BLUE



- 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

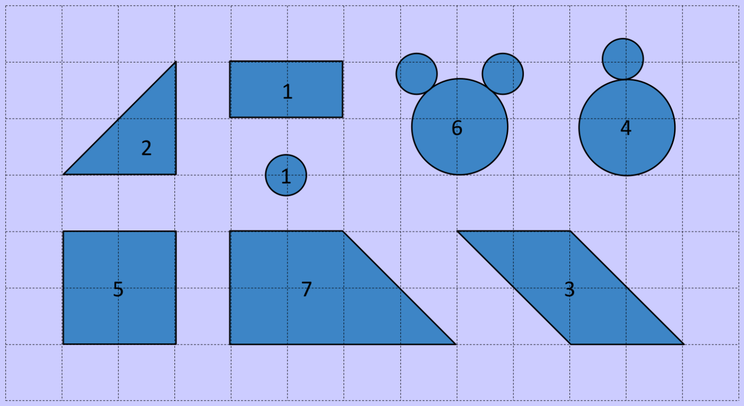
Macedonia

Part C

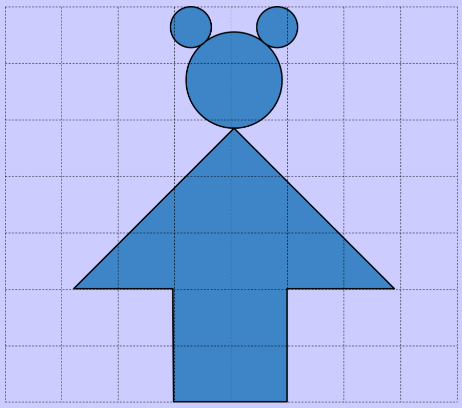
Craft

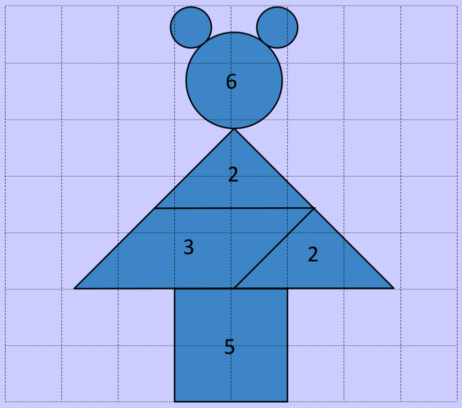
Story

The following shapes are available to make a craft. There is no limit on how many times each shape can be used, but you have to pay every time you use a shape. The number on a shape is the shape's cost (in dollars). The shapes can be rotated.



One way to make the craft shown on the left is by arranging shapes as shown on the right. The total cost of this construction is 18 dollars.





Question

What is the minimum possible total cost to make the same craft?

(A) 13 dollars

(B) 14 dollars

(C) 15 dollars

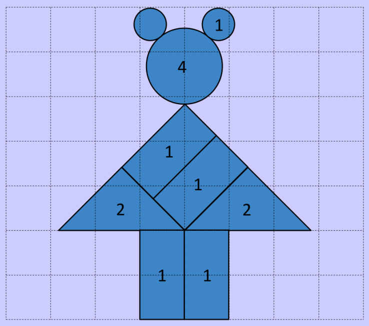
(D) 16 dollars

Answer

(A) 13

Explanation of Answer

The image below shows how to assemble the craft for a total cost of $1+4+2+1+1+2+1+1=13$ dollars. Since this is the smallest option given, we know that Option A must be the correct answer.



To be more complete, we will show why the minimum cost for making the craft must be at least 13 dollars. To do this, we will break down the craft into smaller parts. Only the three shapes with curved sides can be used to form the "head" of the craft, and the remaining five shapes can be used to form the "body" of the craft (the "torso" and the "legs").

There are only two ways to make the head of the craft: using the single shape identical to the head, costing 6 dollars, or by combining the other two curved shapes, costing $1+4=5$ dollars. Therefore, the minimum cost to make the head of the craft is 5 dollars.

Now we need to show that the minimum cost to build the torso and legs of the craft must be at least 8 dollars (more than 7 dollars).

The bottom left and bottom right of the torso must each be built using a trapezoid that costs 7 dollars, parallelogram that costs 3 dollars, or triangle that costs 2 dollars. We will consider all the possibilities.

If the trapezoid is used, other pieces must also be used to complete the torso and legs and so the cost to build the torso and legs would be more than 7 dollars.

If the parallelogram is used, then two triangles must be used as in the example given in the question. This leaves the legs incomplete and so the cost to build the torso and legs would be more than 7 dollars.

The remaining possibility is to place a triangle at each of the bottom corners of the torso (costing $2+2=4$ dollars). Two square areas (at the top of the torso and forming the legs) would still need to be built. The cheapest way to cover these square areas is to use a total of four rectangles (costing another 4 dollars). Hence, the minimum cost to cover the torso and legs is at least $4+4=8$ dollars.

Connections to Computer Science

One approach to solve this problem is to use the *divide and conquer* technique. The key idea is to repeatedly break down a problem into two or more subproblems of the same or related type, until these become simple enough to be solved directly. The solutions to the subproblems are then combined to give a solution to the original problem.

In this case, we can "solve" the head and body, in terms of the minimum cost, and then combine them together. We can also subdivide the body further as long as we also take into account pieces that overlap between the torso and legs.

To solve the head and legs it is relatively straightforward. However, the torso is more complex, and, in general, any tangram is *NP-hard* meaning that the best known algorithm for solving this problem is *exhaustive search* where we must try all pieces in all orientations in all configurations. This takes an *exponential* amount of time, in terms of the number of pieces. Thus, finding a solution can take a very long time even if there are as few as 10 pieces.

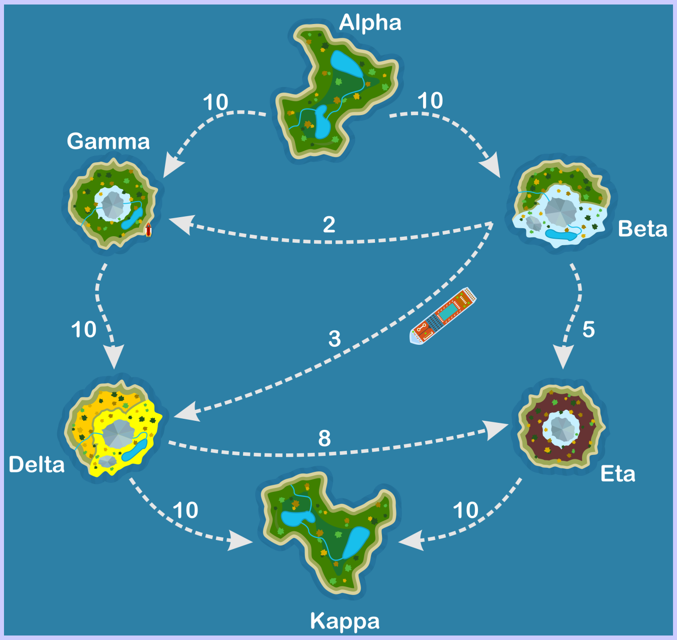
Country of Original Author

China

Vegetable Shipment

Story

A nation consists of six islands called Alpha, Beta, Gamma, Delta, Eta, and Kappa. All vegetables are grown on Alpha and shipped to the other islands. Vegetables are shipped only on the transportation routes indicated by the dotted arrows in the diagram. The number on each arrow represents the maximum amount of vegetables (in tonnes) that can be shipped along that route in a single day.



For example, up to 2 tonnes can be sent from Beta to Gamma in a single day, and up to 8 tonnes can be sent from Delta to Eta in a single day. Alpha always has enough vegetables to ship 20 tonnes per day.

Shipments take very little time to complete. For example, it is possible for vegetables to be shipped from Alpha to Gamma to Delta in a single day, as long as the individual daily route limits are not exceeded.

Question

What is the largest amount of vegetables that can be shipped from Alpha to Kappa in a single day?

(A) 19 tonnes

(B) 18 tonnes

(C) 15 tonnes

(D) 12 tonnes

Answer

(B) 18

Explanation of Answer

We first demonstrate that it is possible to ship a total of 18 tonnes of vegetables to Kappa in a single day.

One way to do this is for Alpha to ship 10 tonnes to Gamma and 8 tonnes to Beta. The amount of vegetables on the five islands other than Alpha will then be:

Island	Beta	Gamma	Delta	Eta	Kappa
Vegetables (in tonnes)	10	8	0	0	0

Next, Gamma should ship 10 tonnes to Delta while Beta should ship 3 tonnes to Delta and ship 5 tonnes to Eta. The vegetables will now be distributed as follows:

Island	Beta	Gamma	Delta	Eta	Kappa
Vegetables (in tonnes)	0	0	13	5	0

At this point, Delta should ship 10 tonnes directly to Kappa and another 3 tonnes to Eta. The distribution of vegetables will then be

Island	Beta	Gamma	Delta	Eta	Kappa
Vegetables (in tonnes)	0	0	0	8	10

Finally, Eta should ship 8 tonnes of vegetables to Kappa after which there will be a total of 18 tonnes of vegetables on Kappa.

We now must argue that it is impossible for more than 18 tonnes to be shipped to Kappa. To see this, we break the islands into two groups. Group X consists of Alpha, Beta, and Gamma, and Group Y consists of Delta, Eta and Kappa.

Notice that there are only three shipping routes that go from an island in Group X to an island in Group Y. The total amount of vegetables that can be shipped from Group X to Group Y cannot exceed the total of the capacities of these three routes. That is, no more than $10 + 3 + 5 = 18$ tonnes of vegetables can be shipped from Group X to Group Y in a single day. Since Alpha is in Group X and Kappa is in Group Y, there is no way for more than 18 tonnes of vegetables to be shipped from Alpha to Kappa in a single day.

Connections to Computer Science

The problem of finding a *maximum-flow* in a network is common in computer science. A famous theorem (Ford-Fulkerson, 1956) says that the maximum flow (how many tonnes of vegetables can be shipped) is given by the capacity of the *minimum cut* (the division of Alpha, Beta, and Gamma into one group, and Delta, Eta, and Kappa in the other group) which if removed would disconnect the *source* (Alpha) from the *sink* (Kappa).

A common example is the Internet, where we want to move as much data as possible between routers. Other examples include image processing, airline scheduling, network reliability, and distributed computing. In most of these applications, and particularly for the Internet, solving this problem quickly is important since the Internet is composed of millions of such nodes, each with a capacity, so routing through routers in this network needs to be done as efficiently as possible.

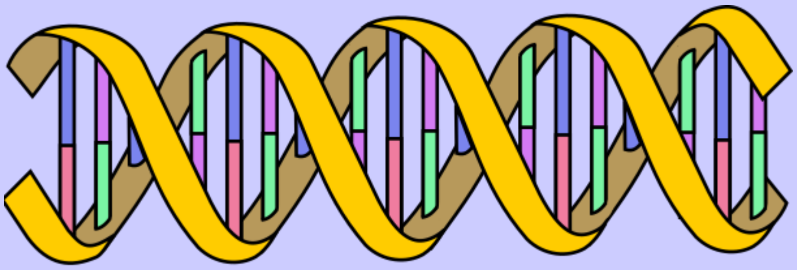
Country of Original Author

Indonesia

DNA Sequence

Story

Genes in cells contain DNA which can tell us a lot about a living thing. A DNA sequence is formed from nitrogen bases. Each nitrogen base is one of four types: Adenine (A), Guanine (G), Cytosine (C), or Thymine (T). DNA can mutate to form a new sequence that is different from the original sequence.



Vormi is a creature for which each mutation is one of three kinds:

1. Substitution: Change one occurrence of a base to another base type.
   Example: AGGTC becomes AGATC (change second G to A).
2. Deletion: Remove one occurrence of a base.
   Example: AGGTC becomes AGTC (delete one G).
3. Duplication: Replace one occurrence of a base with two occurrences of the same base.
   Example: AGGTC becomes AGGTTC (duplicate T).

Question

If Vormi's DNA sequence is initially GTATCG, what sequence **cannot** be the result after exactly three mutations?

(A) GCAATG

(B) ATTATCCG

(C) GAATGC

(D) GGTAAAC

Answer

(D) GGTAAAC

Explanation of Answer

Options A, B, and C can be obtained in exactly three steps:

- Option A: substitution (T to C $\rightarrow$ GCATCG), substitution (T to A $\rightarrow$ GCAACG), substitution (C to T $\rightarrow$ GCAATG).
- Option B: substitution (G to A $\rightarrow$ ATATCG), duplication (T $\rightarrow$ ATTATCG), duplication (C $\rightarrow$ ATTATCCG).
- Option C: substitution (T to A $\rightarrow$ GAATCG), substitution (C to G $\rightarrow$ GAATGG), substitution (G to C $\rightarrow$ GAATGC).

Options A, B, and C can be reached by 3 steps of gene mutation, while Option D can be reached by 4 steps: duplication (G $\rightarrow$ GGTATCG), duplication (A $\rightarrow$ GGTAATCG), substitution (T to A $\rightarrow$ GGTAAACG), deletion (G $\rightarrow$ GGTAAAC).

To see why Option D cannot be reached in exactly 3 steps, notice that there must be more duplications than deletions because there is one more base in GGTAAAC than in GTATCG. Thus, if there are exactly 3 steps resulting in Option D, there are only two possibilities:

- There are 2 duplications and 1 deletion. In this case, the only way to generate AAA in the middle of the sequence is to duplicate A twice. The single deletion cannot be used to both match the GG at the beginning of GGTAAAC and the C at the end of GGTAAAC.
- There is 1 duplication and 2 substitutions. In this case, the only way to generate AAA in the middle of the sequence is to duplicate A and perform a substitution on either T. This leaves a single substitution which cannot be used to both match the GG at the beginning of GGTAAAC and the C at the end of GGTAAAC.

Since neither of these possibilities can be realized, we know Option D cannot be reached in exactly 3 steps.

Connections to Computer Science

To solve this problem, we need to find the distance between one DNA sequence and another. We have to try all the choices and calculate the minimum distance of each given DNA sequence. We can think of each DNA sequence as a *word*, over an alphabet composed of the letters {A, C, G, T}.

The minimum distance between two words is known as the *Levenshtein distance.* The Levenshtein distance is sometimes referred to as the *edit distance* because we wish to find the minimum number of single character edits to transform one word into the other. Typically, operations allowed for an edit are *insertion* (adding one character), *deletion* (removing one character), and *substitution* (replacing one character with another). In this task, the insertion operation is replaced by the duplication operation.

One important application of computing the Levenshtein distance is a *spell checker*, since we want to find the “closest” words to a misspelled word.

Country of Original Author

Indonesia

Mixed Results

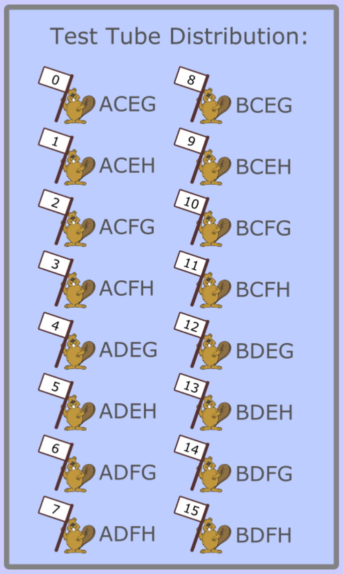
Story

A doctor has 16 patients numbered $0, 1, 2, \ldots 15$ and 8 test tubes labelled A, B, C, D, E, F, G, and H.

Exactly one patient is ill. The doctor takes a blood sample from each patient and divides it into four test tubes mixing it with samples from other patients.

The Test Tube Distribution shown indicates which test tubes the blood samples for each beaver are mixed into. For example, the blood of patient 0 was divided amongst test tubes A, C, E and G.

Sending a test tube to a lab will produce an *infected* result if it contains the blood from the ill patient. Otherwise, a test tube will produce a *healthy* result. The first three lab results are shown below.



Test results so far:

🟢 Test tube C - healthy

🔴 Test tube A - infected

🟢 Test tube E - healthy

Question

In order to identify the ill beaver on the fourth lab test, which of the following test tubes could be sent to the lab?

(A) Test tube B

(B) Test tube D

(C) Test tube F

(D) Test tube G

Answer

(D) Test Tube G

Explanation of Answer

We know there is exactly one patient who is ill, and so we have three conditions:

1. they must be one of the patients 0 through 7, since Test Tube A reported the infected case there;
2. they must not be one of the patients 0 through 3, since Test Tube C reported all of these patients were healthy;
3. they must not be one of patient 4 or 5, since Test Tube E reported both of these patients were healthy.

Thus, we can see that the only remaining patients who satisfy the three conditions listed above are patients 6 and 7. Only Test Tubes G and H distinguish these two patients, and only Test Tube G was one of the options available as a possible answer so it is the correct answer.

Connections to Computer Science

Group Testing is a technique that can be traced back to blood tests by Richard Dorfman during World War II. Since then, the technique has been applied to many areas, not only in biology, but also engineering and computer science, particularly in relation to *defect detection*.

One example application is to determine which lightbulb is defective in a string of N lightbulbs connected in series. Testing individual bulbs would take N tests in the worst case, but with *binary splitting*, we can test with far fewer than N tests. The concept of *binary splitting* is connected to the idea of *binary search* which is an efficient way to locate data in a sorted linear structure.

In this particular task, the connection to binary can be observed in the test tube distribution. Notice that if we assign A, C, E, and G to the value 0 and B, D, F, and H to the value 1, then each group of test tubes is just the binary representation of the patient number. For example, patient 0 is $0 = ACEG = 0000_2$, patient 1 is $1 = ACEH = 0001_2$, and patient 15 is $15 = BDFH = 1111_2$.

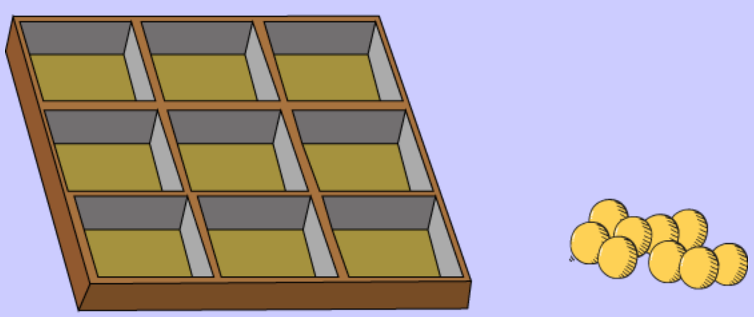
Country of Original Author

Ireland

Nine Marbles

Story

Hira has a box with nine compartments:



Hira chooses 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9 marbles and places them in the box according to the following rules:

- Each marble is in a different compartment.
- The total number of marbles in each row is even.
- The total number of marbles in each column is even.

Question

In how many different ways can Hira place the marbles in the box?

(A) 12

(B) 16

(C) 64

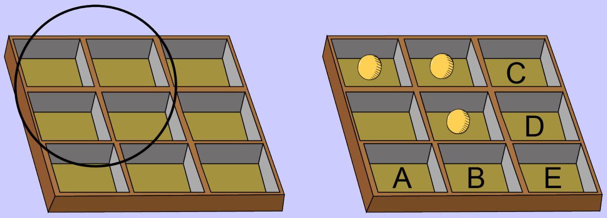
(D) 512

Answer

B) 16

Explanation of Answer

Consider the top left 2×2 section of the box. There are 4 compartments in this section and each compartment will either have a marble in it or not. Therefore, there are $2^4 = 16$ different ways Hira can place marbles in this section. One way Hira can place marbles in this section is as shown:



Since the first column only has one marble, Hira must place a marble in compartment A to make the column total even. The second column already has an even number of marbles so Hira must leave compartment B empty. Using similar reasoning, Hira must leave compartment C empty and place a marble in compartment D.

In general, the placement of marbles in the top left 2×2 section of the box leaves Hira with no choice regarding how to fill compartments A, B, C, and D. For each of compartments A, B, C, and D, the requirement that the column totals and row totals must be even will force Hira to either include a marble or not.

What about compartment E?

Suppose the bottom row (excluding E) contains an odd number of marbles. Then either A contains a marble and B is empty, or vice versa. Suppose A contains a marble and B is empty. Since A contains a marble the two compartments above A must be different. Since B is empty the two compartments above B must be the same. Since the two compartments above A are different but the two compartments above B are the same, the compartments C and D will be different. This means that the third column (excluding E) also contains an odd number of marbles. Since the bottom row and third column, both excluding E, both contain an odd number of marbles, Hira is forced to place a marble in compartment E.

Similarly, it can be shown that if the bottom row (excluding E) contains an even number of marbles, then it is also true that the third column (excluding E) contains an even number of marbles. In this case, Hira is forced to leave compartment E empty.

In summary, we have shown that no matter how marbles are placed in the top left 2×2 section of the box, there is exactly one way to place marbles in the other five compartments according to the rules. The only choice Hira has is selecting one of the possible 16 arrangements of marbles for the 2×2 section of the box.

Connections to Computer Science

Transmitting data both safely and accurately is an important computer science task. One way to make sure that data has not been lost or changed during transmission is to perform a *parity check*.

A *parity bit* is added to the end of the original message based on the contents of the original message. When the message is received at its destination, the parity bit is recalculated and if it doesn't match the parity bit received, it serves as evidence that the message has been corrupted during transmission.

In this task, the final row and final column serve as parity bits. If the box of marbles was sent as a message, the receiver could check the totals and if they are not even, report back to Hira that something has gone wrong.

In general, parity bits can be generalized to *check digits*, which are used in credit card verification, ISBN codes for books in libraries, and barcodes for products that are scanned at a store.

Country of Original Author

Vietnam