UNIVERSITY OF WATERLOO

The CENTRE for EDUCATION in MATHEMATICS and COMPUTING

2022 Beaver Computing Challenge (Grade 9 & 10)

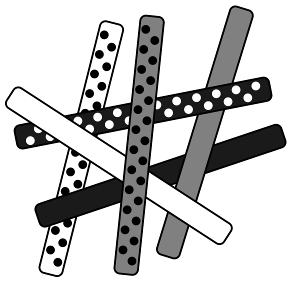
Questions, Answers, Explanations, and Connections

Part A

Pick Up Sticks

Story

Ana drops six sticks on a table as shown.

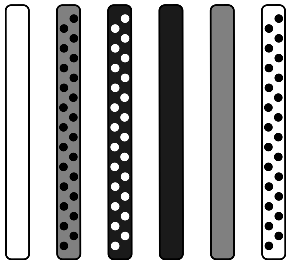


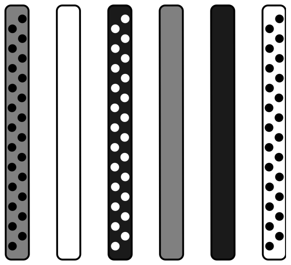
Then she picks all the sticks up according to the following rules:

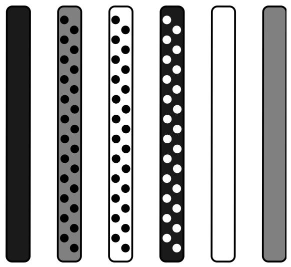
1. Pick up one stick at a time.
2. Only pick up a stick if no other stick is on top of it.

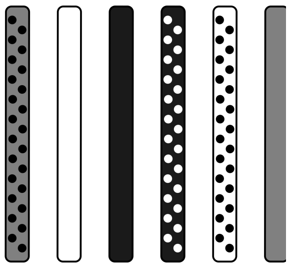
Question

In which order did Ana pick up the sticks?

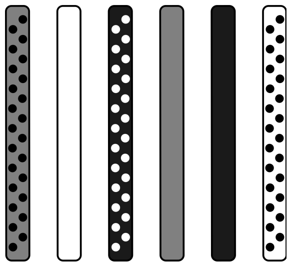
(A) 

(C) 

(B) 

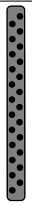
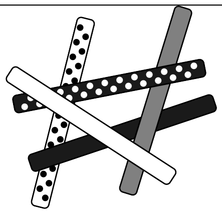
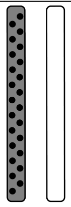
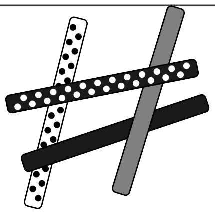
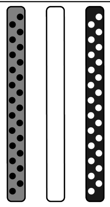
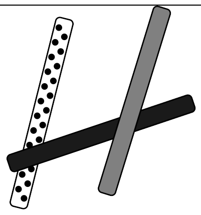
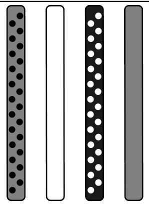
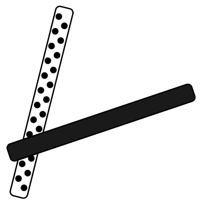
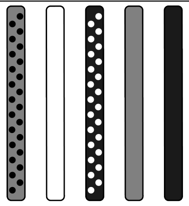
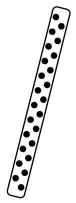
(D) 

Answer

(C) 

Explanation of Answer

According to the rules, Ana must pick up the stick on the top of the pile each time. Therefore, she must pick up the sticks as follows.

Sticks Removed	Sticks Remaining
	
	
	
	
	

Thus, Ana picked up the sticks in the order shown in Option C.

Connections to Computer Science

This task focusses on following operations in *sequential order*. That is, each stick must be removed from the top of the pile one at a time. After each operation, the *state* of the pile of sticks changes, and a new stick at the top.

The pile of sticks can be thought of as a simple *data structure* is used in computer science, called a *stack*. A stack follows the *last in, first out* or *LIFO* rule: the item that is placed in last is the first to be removed. Stacks are used frequently in computer science. One such usage is keeping track of operations that can be "undone", such as edits to a document, browsing history in a browser, or searching through a maze by way of *backtracking* if a dead end is reached.

Country of Original Author

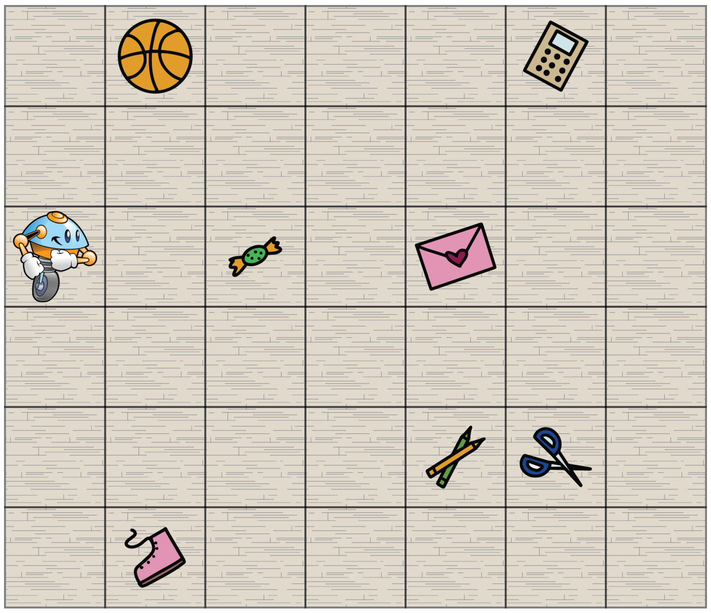
Brazil

Lost In Space

Story

The mission to explore planet Castor was a success, except for the astronauts losing their personal belongings!

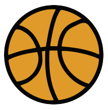
A self-driving robot was sent back to Castor in order to collect all the missing items. The robot can only drive north (↑), south (↓), east (→), and west (←). The robot is currently located in the first column on the third row and it has detected the location of seven lost items as shown:

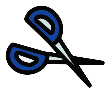


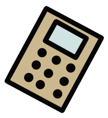
The robot is programmed to identify the item it can get to by driving through the least number of cells. Then it moves to the cell containing that item, and picks the item up. The robot repeats this program until all detected items have been picked up.

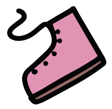
Question

Which item does the robot pick up last?

(A) 

(B) 

(C) 

(D) 

Answer

(D)

Explanation of Answer

We will say the *distance* that the robot is from an item is the number of cells it must drive through to get to that item, including the cell containing that item. Let's record these values as we trace through the robot's program in order to determine the item it picks up last.

Current Location	**Items Detected (Distance)**	**Item Picked Up**	**New Location**
1st column in 3rd row	basketball (3 cells), calculator (7 cells), candy (2 cells), letter (4 cells), pencils (6 cells), scissors (7 cells), shoe (4 cells)	candy	3rd column in 3rd row
3rd column in 3rd row	basketball (3 cells), calculator (5 cells), letter (2 cells), pencils (4 cells), scissors (5 cells), shoe (4 cells)	letter	5th column in 3rd row
5th column in 3rd row	basketball (5 cells), calculator (3 cells), pencils (2 cells), scissors (3 cells), shoe (6 cells)	pencils	5th column in 5th row
5th column in 5th row	basketball (7 cells), calculator (5 cells), scissors (1 cell), shoe (4 cells)	scissors	6th column in 5th row
6th column in 5th row	basketball (8 cells), calculator (4 cells), shoe (5 cells)	calculator	6th column in 1st row
6th column in 1st row	basketball (4 cells), shoe (9 cells)	basketball	2nd column in 1st row
2nd column in 1st row	shoe (5 cells)	shoe	2nd column in 6th row

Since all items have been picked up, the last item picked up by the robot is the shoe.

Connections to Computer Science

The process that the robot 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, the robot makes decisions based only on its current location and the relative distances to the items at this location. It does not take into consideration the overall layout of the items and the distances between pairs of items. That is, instead of trying to look for the *global minimum*, instead the robot looks for a *local minimum*.

Greedy algorithms usually produce answers very efficiently, but unfortunately, they do not always guarantee the correct result. For example, the *change making problem* requires you to make a certain amount of change with the fewest number of coins. If you use a greedy algorithm, this can be done by repeatedly using a coin of as large a value as possible. You stop this process when the desired total change is reached. In most currency systems, this approach will produce the correct minimum number of coins. For example, in Canada, if you used this approach to make change for \$1.50, you would end up with one loonie and two quarters which is a total of three coins. However, if Canada introduced a coin worth 75 cents, then you could make change with just two coins in total, and the greedy algorithm will fail.

However, even if a greedy algorithm does not produce the absolute best answer, sometimes they can be used to find a solution that is “good enough”. This is called an *approximation algorithm*. In some cases, solutions that are “good enough and fast” are more desirable than “perfect and extremely slow”.
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*.

The area of *artificial intelligence*, and specifically *machine learning*, uses greedy algorithms to attempt to find the best approximation to particular problems, such as *image classification*.

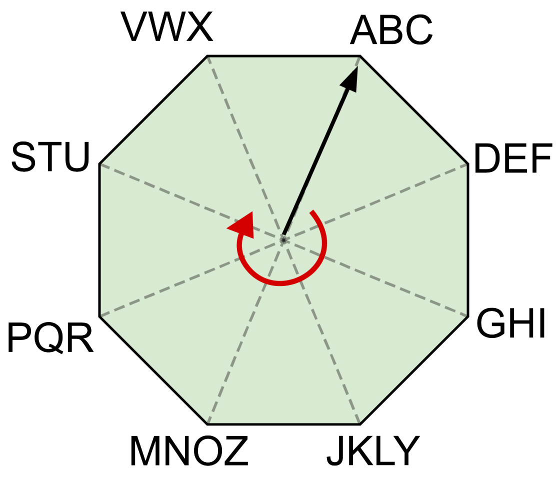
Country of Original Author

China

Octagon Cipher

Story

In an octagon cipher, groups of letters are placed at each vertex of an octagon. An arrow points from the center of the octagon to a letter group, and the arrow can rotate clockwise.



This octagon cipher is used to create secret versions of words. For each word, the arrow begins pointing to ABC. Then a pair of digits is generated for each letter in the word as follows:

- The first digit is the number of vertices the arrow should be rotated from its current position to reach the desired letter group (0, 1, 2, 3, 4, 5, 6, or 7).
- The second digit is the position of the desired letter in the letter group (1, 2, 3, or 4).

The pairs of digits are then separated by dashes. For example, the secret version of the word TREE is 62-73-42-02.

Question

What is the secret version of the word WATER?

(A) 72-11-26-32-53

(B) 62-11-62-22-43

(C) 62-11-26-22-53

(D) 72-11-62-32-43

Answer

(D) 72-11-62-32-43

Explanation of Answer

We will walk through the process of creating the secret version of the word WATER.

- The arrow begins pointing to ABC. To reach the letter group with W we must rotate the arrow 7 vertices to the letter group VWX. The letter W is the second letter in this group, so the first pair of digits is 72.
- The second letter in the word is A. To reach the letter group with A we must rotate the arrow 1 vertex from the current position to the letter group ABC. The letter A is the first letter in this group, so the second pair of digits is 11.
- The third letter in the word is T. To reach the letter group with T we must rotate the arrow 6 vertices from the current position to the letter group STU. The letter T is the second letter in this group, so the third pair of digits is 62.
- The fourth letter in the word is E. To reach the letter group with E we must rotate the arrow 3 vertices from the current position to the letter group DEF. The letter E is the second letter in this group, so the fourth pair of digits is 32.
- The last letter in the word is R. To reach the letter group with R we must rotate the arrow 4 vertices from the current position to the letter group PQR. The letter R is the third letter in this group, so the last pair of digits is 43.

Putting these pairs of digits together gives us 72-11-62-32-43, which is the secret version of the word WATER.

Connections to Computer Science

One method of keeping data secret so that unauthorized persons cannot access it is *encryption*, which is the key area of study in *cryptography*. Cryptography began as early as 3500 years ago, which simple methods of encryption such as replacing each letter in the message with a different letter, which is known as a *substitution cipher*. Simple substitution ciphers can be easily *decrypted* by using *frequency analysis*: some letters, such as "E" and "T" appear much more than other letters, such as "Q" or "Z" in English text. Therefore, someone reading a long encrypted message can try to guess that the most frequently occurring encrypted letters are probably "E" or "T" and begin to decrypt the message.

An alternative method to a direct substitution cipher is to use *rotation-based* ciphers, which typically were mechanical devices with various rotors that had sequences of letters, and with each letter, the rotor would move some amount. The most well-known of these sorts of devices is the *Enigma machine* used to transmit messages during World War II. Similar to this machine in this task, each letter rotates a different amount, such that the sequence of letters "PA" and "SA" (51-31 and 61-21) have the letter "A" encoded with two different values. This type of encryption method is much more difficult to decrypt without knowing the original starting point of the arrow, and is much less vulnerable to frequency analysis.

Many computer systems, such as *databases* and *web browsers*, rely on strong encryption methods to prevent unauthorized access to information.

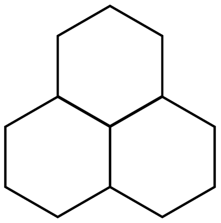
Country of Original Author

Slovakia

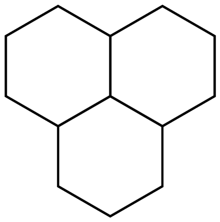
Colourful Tower

Story

Luis has hexagon pieces in three different colours. Whenever Luis arranges three pieces in a way that resembles an upright triangle, the three pieces must either be *all the same colour*, or *all different colours*. These rules do not apply to other three-piece arrangements. In particular:

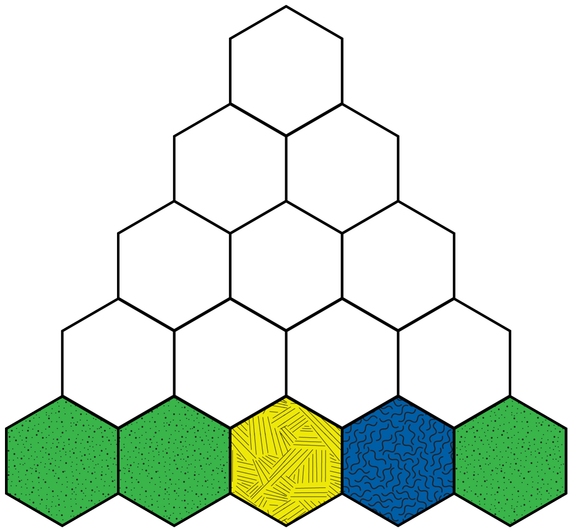


All colours the same
or all colours different



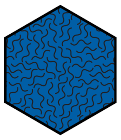
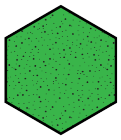
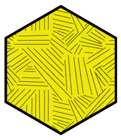
No colour rules

Luis arranges his hexagon pieces in a way that resembles a tower as shown:

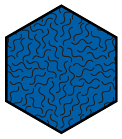


Question

Which hexagon piece must be at the very top?

(A)  (B)  (C)  (D) There is more than one possibility

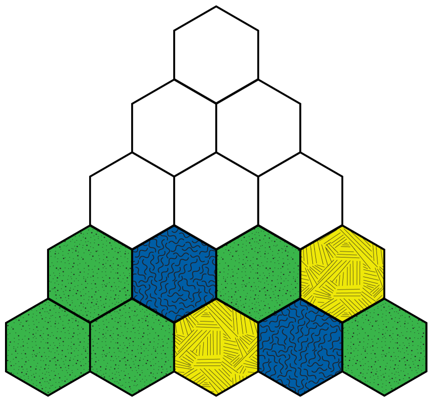
Answer

(A) 

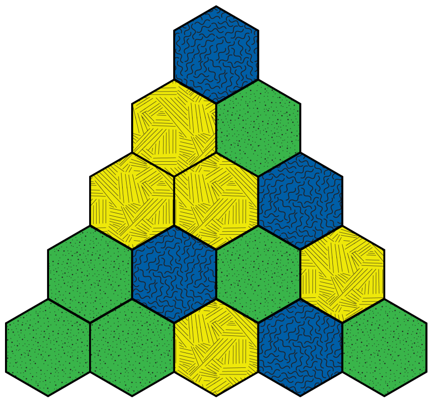
Explanation of Answer

Once we know the colours of two hexagon pieces beside each other, we can determine the colour of the piece directly above them. If two pieces side-by-side are the same colour, then the piece above them will be that colour also. However, if two pieces side-by-side are different colours, then the piece above them will be neither colour.

Using this idea, the next row up from the bottom will be:



We can continue building the tower from the bottom up until it is complete:



The hexagon piece at the very top is blue.

Connections to Computer Science

This task relies on two fundamental concepts in *algorithms*: the concept of *conditional decisions* and *repetition*.

The conditional decision in this task can be thought of as the following:

```
IF colour of cell below left is the same as colour of cell below right
THEN
    colour of this cell is the same as cell below left
OTHERWISE
    colour of this cell is different than both cell below left and cell below right
```

Conditional statements are often represented in *programming languages* using *if-then-else* statements, which allow certain steps to happen only if a given decision/question has a *true* value.

To solve the entire task, the above steps need to be *repeated*, from the bottom levels up to the very top level. In computer programs, repetition is often represented as a *loop*.

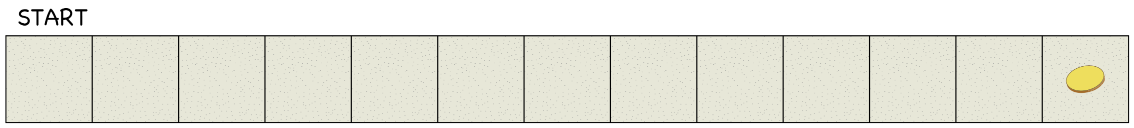
Country of Original Author

Vietnam

Jumping Game

Story

Verunka has invented a game to play on her sidewalk. Her sidewalk is 13 squares long and there is a coin on the very last square.



Verunka marks each square (except the last) with either an X or an O. Then, she begins playing by jumping on the square labelled START and using the following rules:

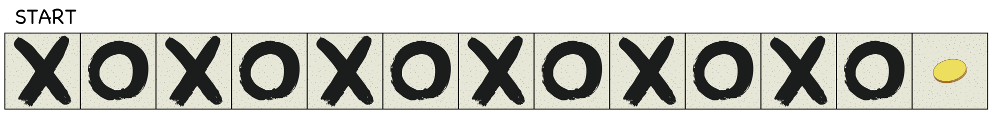
- After landing on a square marked X, jump forward 3 squares.
- After landing on a square marked O, jump backwards 1 square.

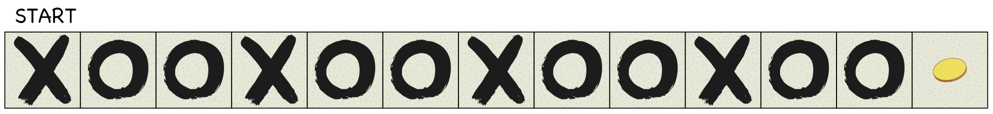
Verunka wins if three things happen:

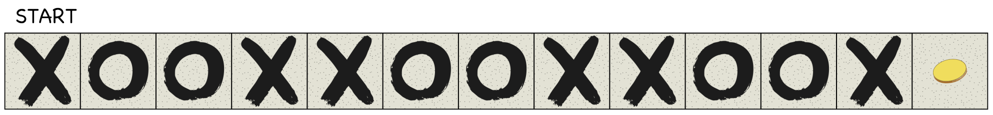
1. She can always follow the rules (i.e. remain on the sidewalk).
2. She lands on the square with the coin.
3. She visits all 13 squares on the sidewalk.

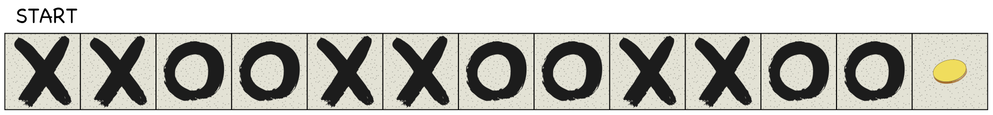
Question

For which marking will Verunka win the game?

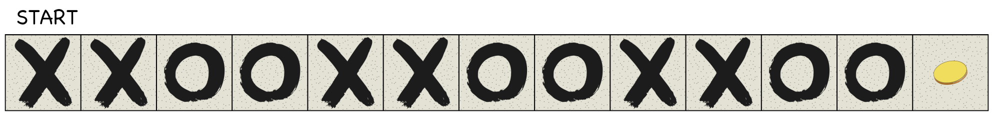
(A)


(B)


(C)


(D)


Answer

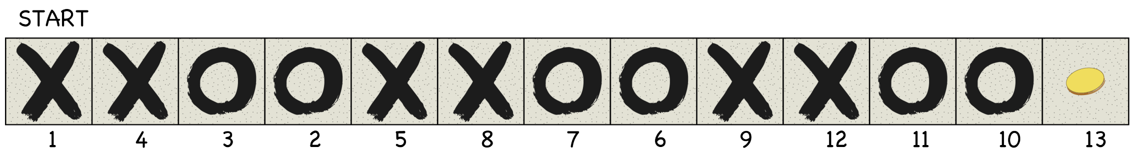
(D)


Explanation of Answer

In Options A and C, Verunka will not be able to follow the rules. That is, she will reach an X less than 3 squares from the coin.

Verunka lands on the square with the coin in Option B, but she does not visit each square along the way. Specifically, she never visits any squares marked with an O.

The correct answer is Option D as shown. The numbers underneath the squares indicate the order in which Verunka visits them.



Connections to Computer Science

This task focusses on *tracing* a simple *algorithm* on certain *input*.

The algorithm is fairly straightforward, and can be modelled using a *conditional statement*: if Verunka lands on an “X” square, then jump forward three squares; if Verunka lands on an “O” square, then jump backwards one square. Many *programming languages* have the ability to write conditional statements, typically with an *if-statement*.

In order to trace this algorithm, we must keep track of the *state*, which will change at each jump. The state for this particular task must keep track of the current square that Verunka occupies as well as all the squares that she has visited. In programming languages, this could be implemented by way of an *array*, where each cell of the array keeps track of whether or not Verunka has jumped there.

Finally, the input to the algorithm is the sequence of “X” and “O” squares. The input can be viewed as *accepted* or *valid* if all squares are visited, and *rejected* or *invalid* otherwise. Determining the validity of input, such as password verification, is common occurrence in many *software systems*.

Country of Original Author

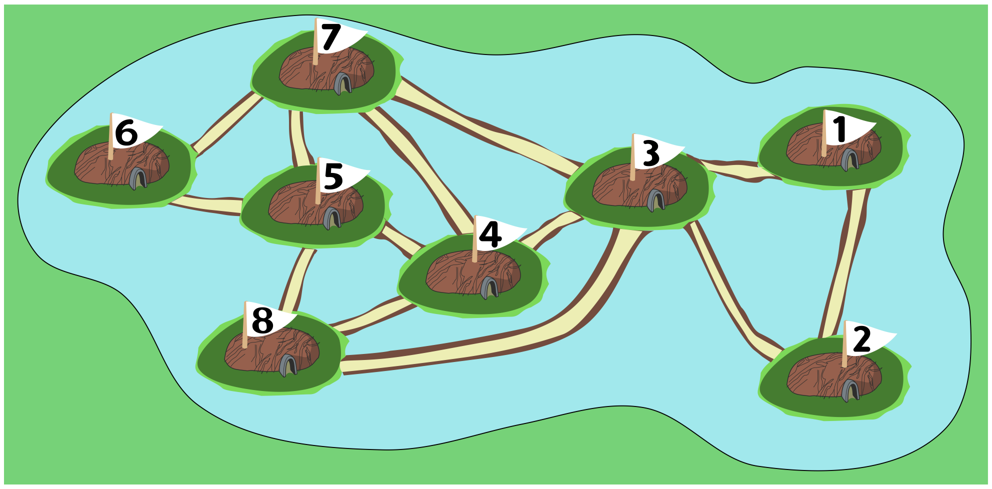
Czechia

Part B

Neighbours

Story

A beaver wants to visit his friend Mary. He doesn't know which home is hers, but he has the following map of her neighbourhood, which shows homes numbered from 1 to 8, and paths between some of the homes.



Two beavers are considered neighbours if there is a path that connects their homes directly.

The beaver knows the following information.

- Mary, Zac, and Pan each have exactly four neighbours.
- Niki has exactly two neighbours: Zac and Pan.

Question

What is Mary's house number?

(A) 5

(B) 7

(C) 4

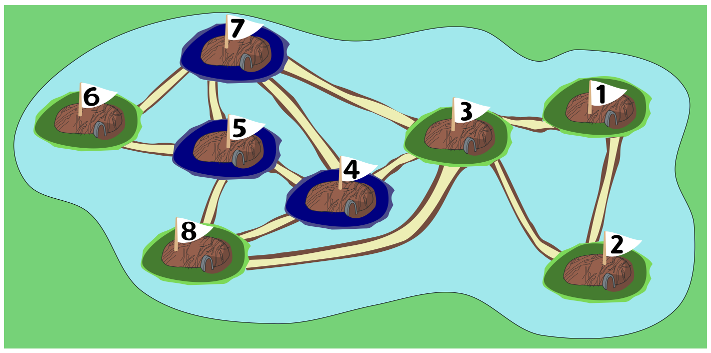
(D) 3

Answer

(C) 4

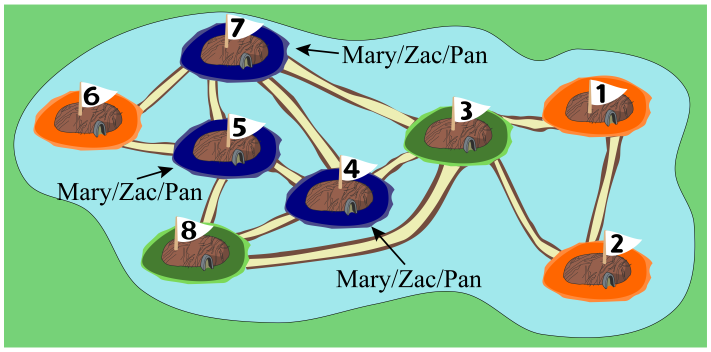
Explanation of Answer

To solve the problem, we first need to identify the homes with four neighbouring homes. There are three such homes, numbered 4, 5, and 7.

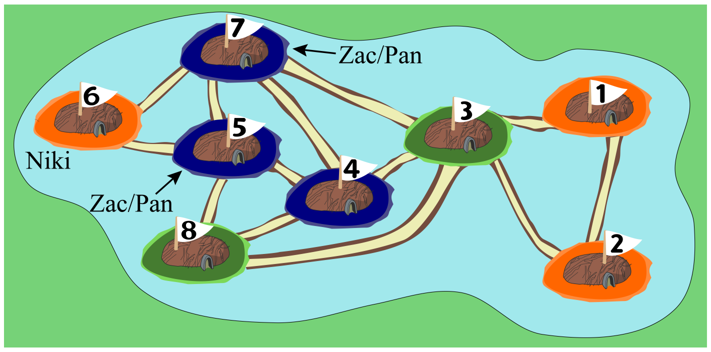


So Mary, Zac and Pan live in homes 4, 5, and 7, in some order.

Next, we identify the homes with two neighbouring homes. There are three such homes, numbered 1, 2, and 6.



Since Niki is neighbours with only Zac and Pan, it follows that Niki lives in home number 6. Then, Zac and Pan live in homes 5 and 7, in some order.



Therefore, Mary lives in home number 4.

Connections to Computer Science

This task combines together elements of *graph theory* and *logic*.

A *graph* is a set of *vertices* and a set of *edges* that connect together pairs of vertices. In this task, the homes represent vertices, and the paths represent edges. Graphs are used to model *networks*, such as electrical grids, flight paths, or the internet routing system. For this particular task, we use information about the *degree* of the vertices, which is the number of edges connected to a vertex.

There is also a few steps of *logical reasoning* in this task. Specifically, using the provided information to find the sets of homes that each person could live, and then reason about what homes that they could, or could not, live in. The use of logical reasoning is very important in writing *algorithms*, since each step of an algorithm involves refining, expanding, or applying past knowledge to move closer to the solution.

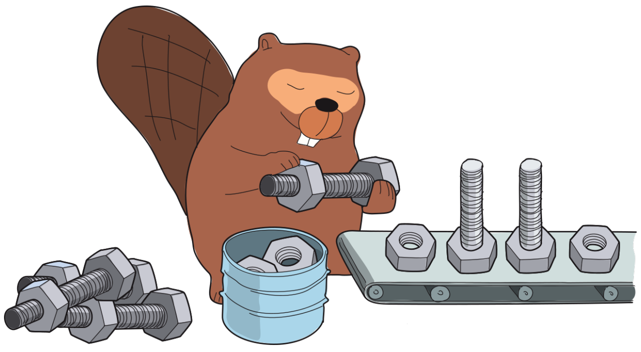
Country of Original Author

Cyprus

Nuts and Bolts

Story

At the Beaver Construction Factory, Lana works on the nuts and bolts assembly line.



Her job description is as follows:

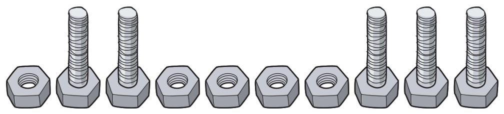
- Lana stands at one end of a long conveyor belt, which contains a line of nuts and bolts.
- Lana's job is to take each part, either a nut or a bolt, off of the conveyor belt.
- If Lana takes a nut from the conveyor belt, she puts it in the bucket beside her.
- If Lana takes a bolt from the conveyor belt, she takes a nut from the bucket beside her, attaches the nut and bolt together, and adds this to a pile of assembled parts.

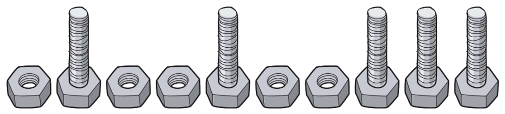
However, things can go wrong for Lana in two different ways:

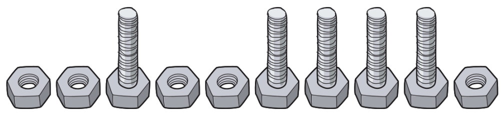
1. Lana takes a bolt from the conveyor belt, and there is no nut in the bucket to attach it to.
2. There are no more bolts on the conveyor belt, and there are still nuts in the bucket.

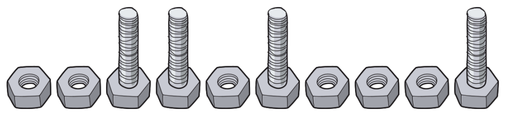
Question

Which sequence of nuts and bolts, when processed from left-to-right, will **not** cause things to go wrong for Lana?

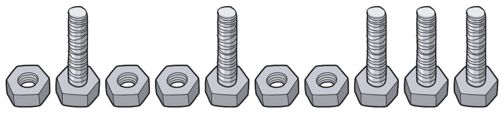
(A) 

(C) 

(B) 

(D) 

Answer

(C) 

Explanation of Answer

For Option C, we can check that things don't go wrong by keeping track of the state of the bucket and conveyor belt from left-to-right using N to represent a nut and B to represent a bolt.

Bucket	**Conveyor Belt**
empty	N B N N B N N B B B
N	B N N B N N B B B
empty	N N B N N B B B
N	N B N N B B B
N N	B N N B B B
N	N N B B B
N N	N B B B
N N N	B B B
N N	B B
N	B
empty	empty

Option A will go wrong after N B B, since there will be no nut in the bucket when the second bolt is encountered.

Option B will go wrong after N N B N N B B B B, since there will be no nut in the bucket when the fifth bolt is encountered. (Notice that there are only four nuts before this fifth bolt.)

Option D will go wrong after the entire sequence is processed, since there will be two nuts left in the bucket. (Notice that there are six nuts and four bolts in total.)

Connections to Computer Science

This task highlights the use of a *push-down automata* (PDA). A PDA is a way of describing a *algorithm* that relies on the current *state*, but also has an unlimited amount of memory in the form of a *stack*. In this task, the state is either having a nut or having a bolt on the conveyor belt, and the stack is the bucket which holds the nuts.

A PDA can be used to recognize, or *parse*, *words* in *context-free languages*. To recognize or parse a word in a language means to determine if the word, which is a sequence of symbols, follows the rules of the language. In this case, we can think of the nuts and bolts as a representation of *balanced parentheses*, where N is equivalent to a "(" and B is equivalent to ")". That is, balanced parentheses are valid arrangements of parentheses in arithmetic expressions. Examples of a sequence of parentheses which are not balanced are (((() or ())(. Detecting balanced parentheses is important in *compilers*, since many *programming languages* rely on parentheses to define *nested scopes* as well as write proper arithmetic expressions.

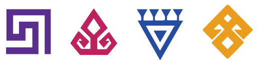
Country of Original Author

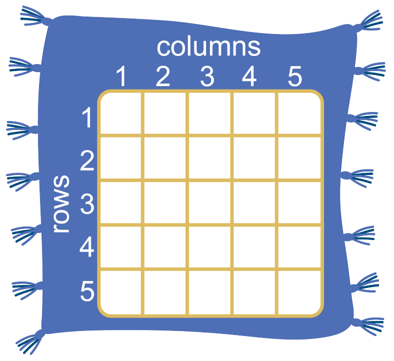
Canada

Rug Weaving

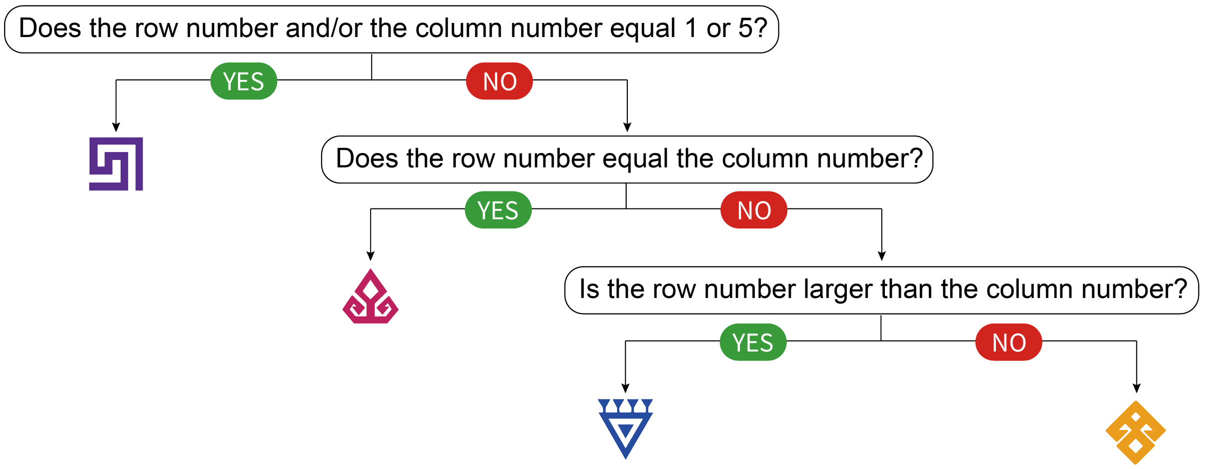
Story

Hale is a Turkish weaving artist. He is creating a square rug that consists of a grid of 25 squares arranged into 5 rows and 5 columns. On each square, Hale will place one of the following four symbols:





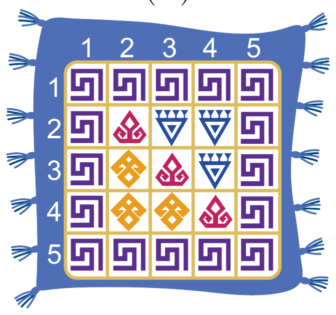
He decides which symbol to place on each square using the row number and column number of the square and following the instructions in the decision tree given below.


Does the row number and/or the column number equal 1 or 5?
YES
NO
Does the row number equal the column number?
YES
NO
Is the row number larger than the column number?
YES
NO

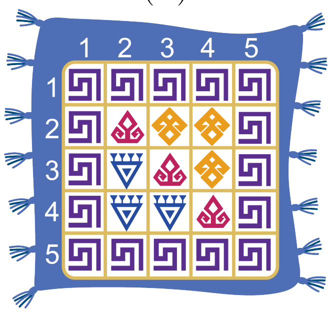
Question

Which of the following rugs is Hale's completed rug?

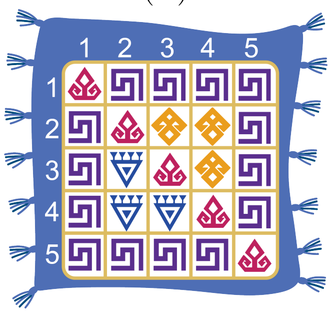
(A)



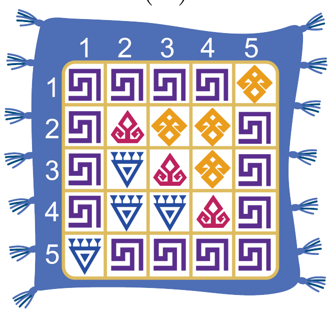
(B)



(C)

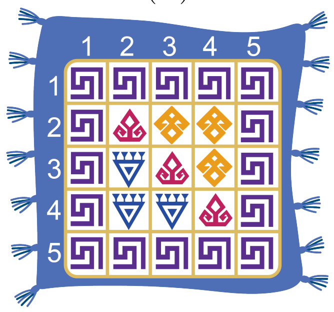


(D)



Answer

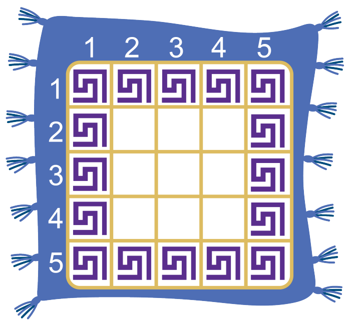
(B)



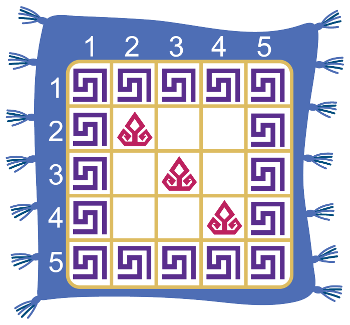
Explanation of Answer

Using the decision tree, we will place each symbol on the rug one at a time.

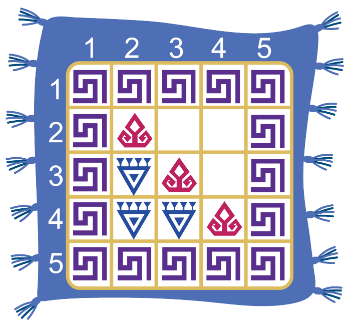
The symbol is placed in all squares in the 1st and 5th rows and columns, as shown.



The symbol is placed in the remaining squares whose row number is equal to their column number. There are three such squares, and they form a diagonal, as shown.



The symbol is placed in the remaining squares whose row number is larger than their column number. There are three such squares to the left of the diagonal, as shown.



The symbol is placed in all remaining squares, which completes the rug shown in Option B.

Connections to Computer Science

The picture used this problem is called a *decision tree*. A decision tree is composed of *nodes*. The *root node* is where we start the decision process. In this task, the root node asks questions about the row/column number being either 1 or 5. Some nodes have a set of *binary* decision branches: these are the results of questions being answered either “yes” or “no”. The other nodes are *leaf* nodes at the bottom of the tree and they represent the final *outcome* based on the decision path.

Decision trees are used in computer science in *artificial intelligence*, specifically in *machine learning*. For example, when a program is being designed to distinguish between various images, such as “dog”, “fish”, or “traffic light”, various features such as “rectangular shape”, “two eyes”, “fins” are used as decisions. Each feature has a “yes” or “no” outcome, and with repeated *training*, the program will develop enough distinguishing features to correctly *classify* an image.

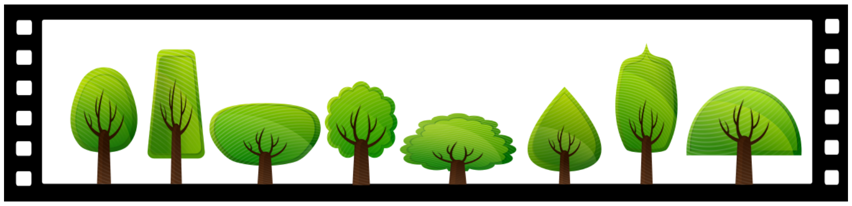
Country of Original Author

Türkiye

Forest Photos

Story

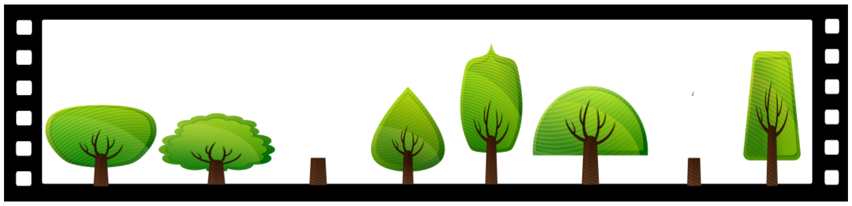
Dai has a camera with a *panorama mode*, which can take one continuous photo while it moves horizontally. Dai stood in the middle of a circle of eight trees, and took the following photo in panorama mode while rotating 360 degrees.

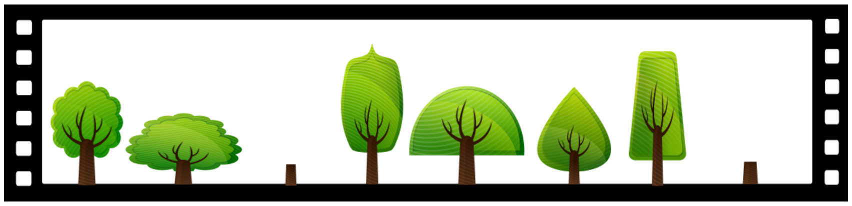


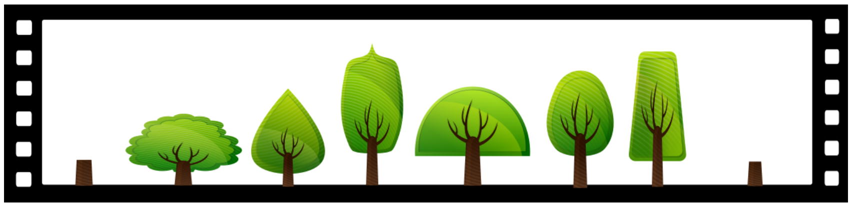
After a few days, Dai returned to the same location and took a second photo moving in the same direction, but starting from a different tree. She saw that two of the trees had been cut down.

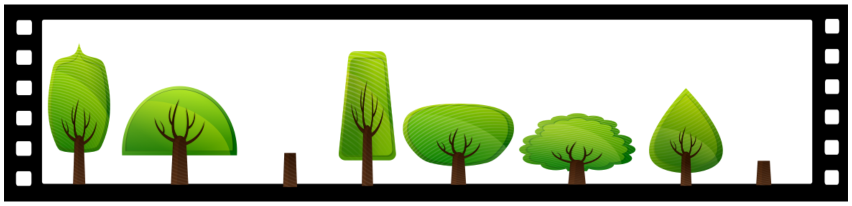
Question

Which of the following is Dai's second photo?

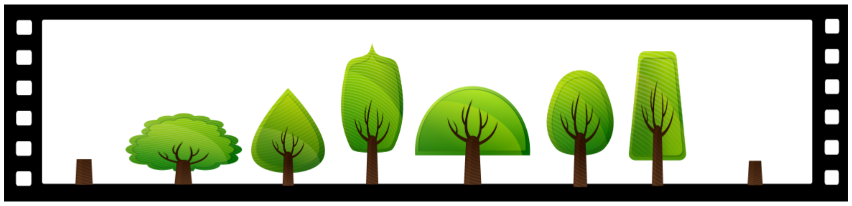
(A) 

(B) 

(C) 

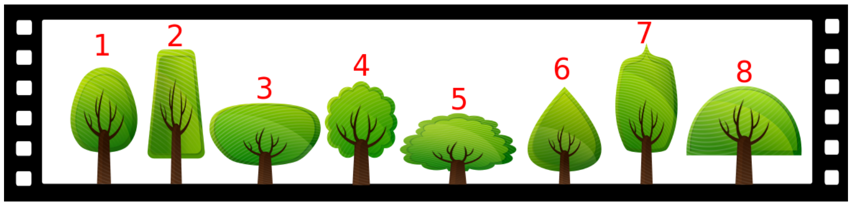
(D) 

Answer

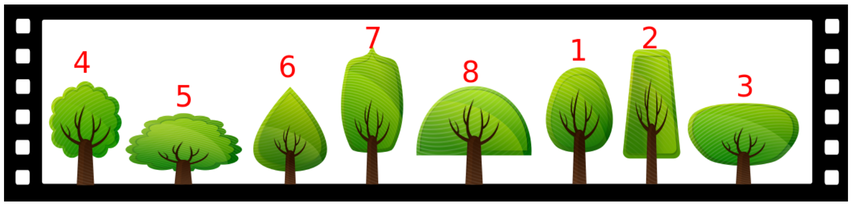
(C) 

Explanation of Answer

To help us, we notice that the trees are distinct and number them as they appear in the first photo:



The second photo corresponds to shifting these numbers by a fixed amount and "wrapping around". In particular, if we shift by 3 we get this:



This order of uncut trees corresponds to Option C where trees 4 and 3 have been cut down.

In Option A, tree 3 appears on the left followed by tree 5, so this does not correspond to a shift with "wrap around" and cannot be Dai's second photo.

In Option B, the leftmost tree is tree 4, so the shift must be by 3 as in Option C. However, tree 6 is the sixth tree from the left which is inconsistent with this shift. This also cannot be Dai's second photo.

In Option D, the leftmost tree is tree 7, so the shift must be 6. Ignoring the trees that have been cut down, the trees should appear in the order 7, 8, 1, 2, 3, 4, 5, 6. However, tree 6 is not the rightmost tree (it is one position earlier), which means this is also not the correct answer.

Connections to Computer Science

In computer science, data is often *transformed* in some way. For example, a list of employee salaries may be transformed to create a list of the income tax owing for each employee.

In this task, we transform the original data, in two ways: first, we shift each tree "back" three positions, wrapping around if necessary; second, we cut down the trees at the end of the rotated list.

This sort of processing of linear data by applying various transformations is an instance of *functional programming*. The most common usage of functional programming is in *spreadsheets*, where some columns of data are combined together and transformed to create new columns of data, which may be further refined or transformed.

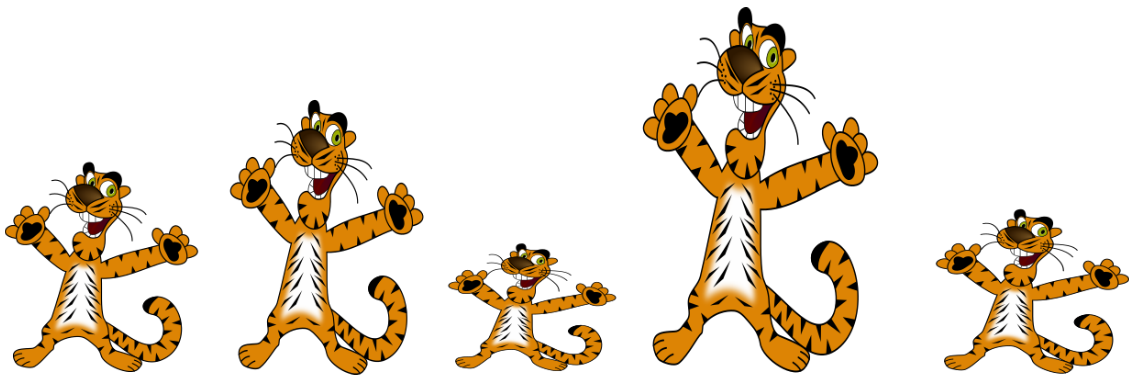
Country of Original Author

Uruguay

Tiger Dolls

Story

At a carnival, five tiger dolls are initially on a shelf in the order shown below. Bo wants to reorder the dolls so that their heights increase from left to right. Bo rearranges the dolls by switching the positions of two dolls at a time.



Question

What is the fewest number of switches Bo can make in order to place the dolls in the desired order?

(A) 3

(B) 4

(C) 5

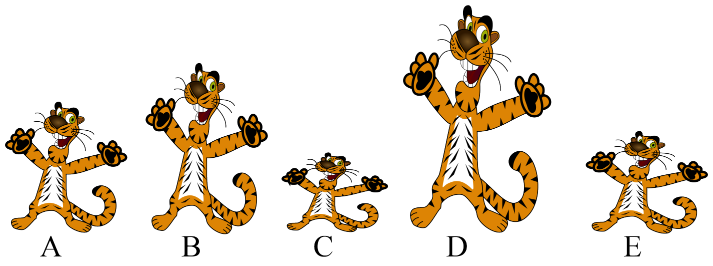
(D) 6

Answer

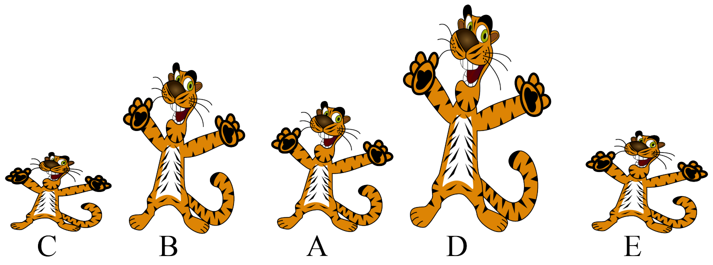
(A) 3

Explanation of Answer

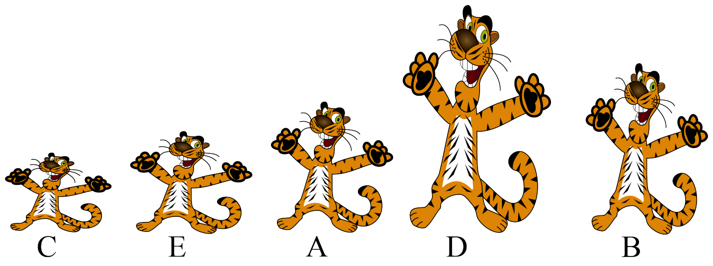
We label the five dolls A, B, C, D, and E, as shown. Then we will show how we can place the dolls in the desired order in 3 switches.



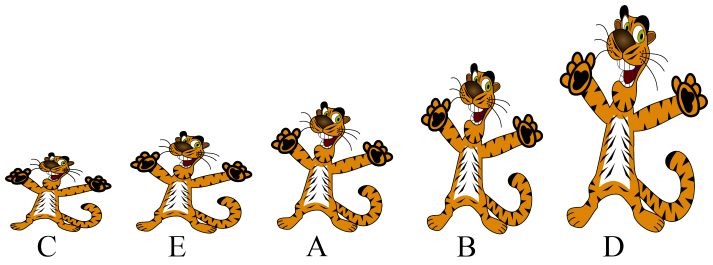
First, switch dolls A and C to obtain the following result.



Then, switch dolls B and E to obtain the following result.



Finally, switch dolls D and B to obtain the following result. The dolls are now in the desired order.



Note that there are other such way that three switches could work. For example, switching D and E, then switching D and B, and then switching A and C.

Since 3 is the least value among the four options, it must be the correct answer. However, we should justify more completely why the correct answer cannot be less than 3. This is because no doll is initially on the shelf in its desired position. Therefore each doll must switch positions at least once. However, with fewer than three switches, at most four dolls can change positions.

Connections to Computer Science

This task involves *sorting* elements. *Sorting algorithms* are algorithms that arrange elements of a list into a certain order, usually either ascending order or descending order. From the beginning of computing, the sorting problem has attracted a great deal of research. Many sorting algorithms have been developed: for example, *bubble sort*, *insertion sort*, and *quicksort*. Each algorithm has its own characteristics. Some algorithms are faster than others, some are more complicated, and some others use special techniques such as *recursion*.

Many sorting algorithms rely on two basic principles: *comparing* two values from the list for their relative size, and *swapping* (or exchanging the positions of) those elements. In this task, we are trying to minimize the number of swaps. One algorithm that does this most efficiently for this particular input is *selection sort*: repeatedly select the smallest element from list of unsorted elements, and swap it to be at the end of all sorted elements, where the first swap will put the minimum element into the first position. It is worth noting that for a different *permutation* of elements, it may not be possible to sort them using selection sort using at most 3 swaps. Finding such a permutation is a worthwhile exercise.

Country of Original Author

China

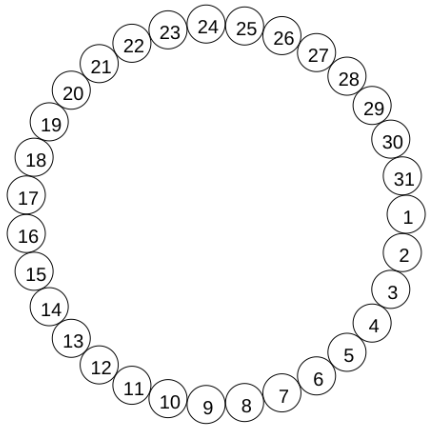
Part C

Classroom Seating

Story

There are 31 empty chairs in a classroom. The chairs are placed in a circle and numbered 1 to 31, as shown. Students enter the classroom, one at a time, and fill the chairs in the following way:

1. When a student enters the classroom, they sit on the chair that has the number of the day of the month on which they were born, unless that chair is already occupied.
2. If that chair is already occupied, then the student starts at that chair and walks around the circle in a clockwise direction, sitting on the first free chair they encounter.



For example, suppose that Geeta and Seeta were both born on April 20, Arun was born on January 21 and Zubin was born on September 22.

- If they enter the classroom in the order Geeta, Seeta, Zubin, Arun, then Geeta sits on chair 20, Seeta sits on chair 21, Zubin sits on chair 22, and Arun sits on chair 23.
- If they instead enter the classroom in the order Seeta, Arun, Zubin, Geeta, then Seeta sits on chair 20, Arun sits on chair 21, Zubin sits on chair 22, and Geeta sits on chair 23.

Question

Suppose six students enter the classroom and are seated as shown:

Student	**Birthday**	**Chair Number**
Abha	May 11	13
Byram	February 12	12
Chetan	September 14	14
David	August 11	11
Eesha	April 13	15
Fatima	July 12	16

Which of the following statements **cannot** be true?

(A) Chetan was the first student to enter.

(B) Fatima was the sixth student to enter.

(C) Eesha entered before Abha.

(D) Byram entered before David.

Answer

(C) Eesha entered before Abha.

Explanation of Answer

Chetan sits in the same chair as his birthday, so he could be the first to sit down. Option A can be a correct statement.

Since she ended up in chair 16, Fatima must have arrived after four other students were already seated in chairs 12 to 15. David is seated in chair 11 which means he arrived before Abha. And, since Abha is sitting in one of the chairs in the range 12 to 15, then five people entered the classroom before Fatima. Therefore Option B must be a correct statement.

Since Eesha ended up in chair 15, it follows that students must have been already sitting in chairs 13 and 14 when she arrived. Since Abha is seated in chair 13, it follows that Abha must have arrived before Eesha. Thus, Option C cannot be a correct statement.

Byram and David are both seated in the chairs with the same number as their birthdays, so their order can be interchanged. Option D can be a correct statement.

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 table is represented by the chairs in the room. The hash function uses each student's birthday as the *key* to determine the ideal position for the student to sit.

As was also demonstrated in this task, sometimes there are *collisions* in the hash function, where two different keys yield the same value. In order to deal with this, various *collision resolution strategies* have been developed to maintain efficiency. In this task, the next available empty position was used: this collision resolution strategy is called *linear probing*. Hashing can provide extremely efficient lookup times even with a massively large amount of data.

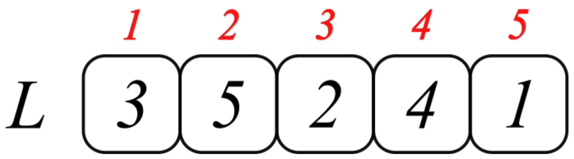
Country of Original Author

India

Lists

Story

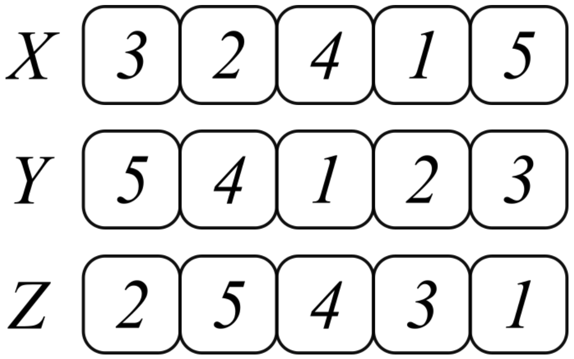
We can represent a sequence of numbers using a list of boxes. Each number in the sequence is placed in a box and the position of each number in the sequence is indicated above the box. For example, the following list is labelled L and represents the sequence 3, 5, 2, 4, 1.



We can use the notation $(L\ N)$ to represent the number that is in position N in list L. For example, $(L\ 2)$ represents the number in position 2 in list L and so $(L\ 2) = 5$. Similarly, $(L\ 5) = 1$.

Note that we can use this notation more than once in an expression. For example, consider the expression $(L\ (L\ 3))$. Since $(L\ 3) = 2$, substituting this value gives $(L\ \mathbf{(L\ 3)}) = (L\ \mathbf{2}) = 5$.

Consider the following three lists that are labelled X, Y and Z.



Question

What is the number represented by the expression $(X\ (Y\ (Z\ 3)))$?

(A) 2

(B) 3

(C) 4

(D) 5

Answer

(A) 2

Explanation of Answer

We look up the value $(Z\ 3) = 4$ and substitute to get $(X\ (Y\ (Z\ 3)) = (X\ (Y\ 4))$.

Then we look up the value $(Y\ 4) = 2$ and substitute to get $(X\ (Y\ 4)) = (X\ 2)$.

Looking up the value $(X\ 2) = 2$ gives us the final answer.

Connections to Computer Science

Storing information in efficient *data structures* is an essential component of *programming*. One very simple data structure is a *list*, which is where this task derives its name.

A list can store any type of information: numbers, names, or other sorts of larger data. In this task, the list is also used to store *addresses* of elements. The address of an element is the location of where the element is being stored: we refer to an address as a *pointer* or an *indirect access* to the element.

Similarly, if an address is stored in a list, we can store the address of where that address is stored: this is a *pointer to a pointer*, which is another layer of indirection. In this task, when we use the notation $(L\ n) = v$, this means that the value v stored at address n.

When any piece of information is stored in *random access memory (RAM)*, the computer needs to keeps track of the address of that information in order to later retrieve it. These addresses are typically stored in a *symbol table*, which is simply a list of pointers to each piece of information.

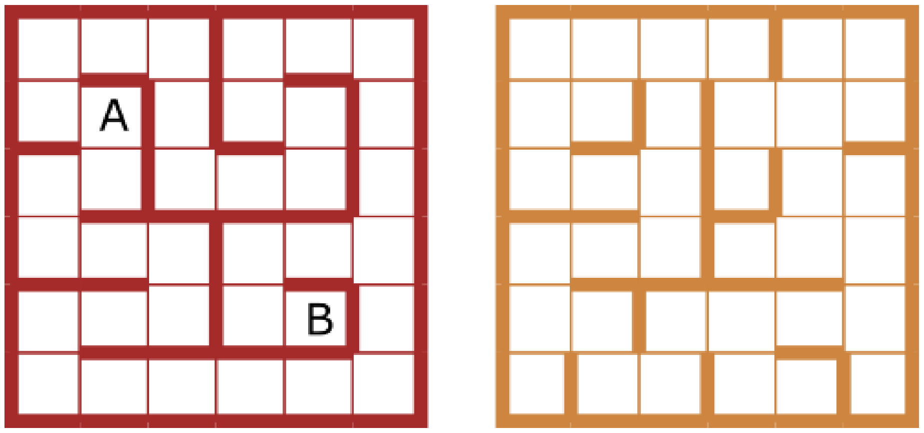
Country of Original Author

Austria

Maze

Story

A beaver is in a maze that consists of two 6-by-6 floors as shown. The bold lines are walls.



The beaver can move between two adjacent cells within one floor if there is no wall between the cells; this takes one second. The beaver can also magically move to the corresponding cell (same row and same column) of the other floor; this takes five seconds.

For example, if the beaver is in cell A, there are three possible moves:

1. Move left. This move takes 1 second.
2. Move down. This move takes 1 second.
3. Move to the corresponding cell of the other floor. This move takes 5 seconds.

The beaver starts at cell A and wants to reach cell B as soon as possible.

Question

What is the shortest time needed for the beaver to reach cell B?

(A) 17 seconds

(B) 18 seconds

(C) 19 seconds

(D) 20 seconds

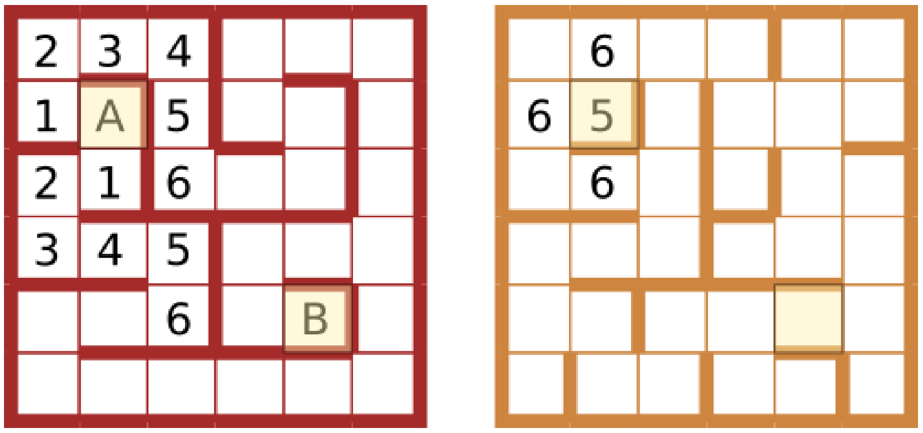
Answer

(B) 18 seconds

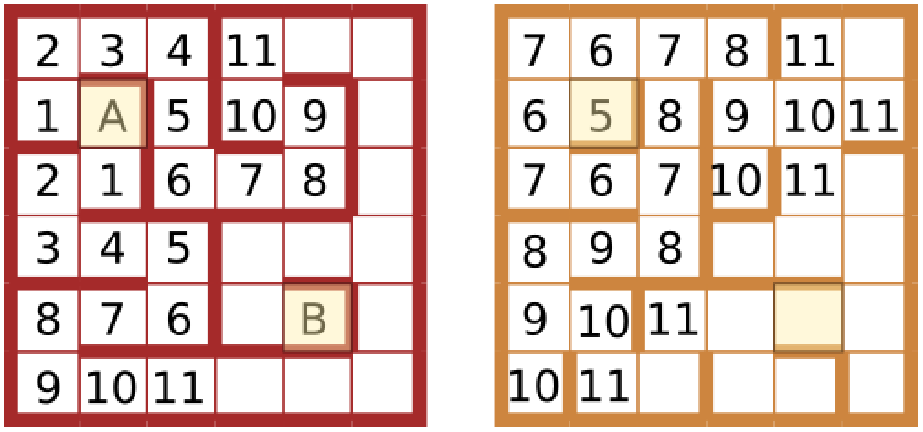
Explanation of Answer

We will find all the cells that can be reached form cell A in one second, two seconds, three seconds, and so on (in that order).

The cells that can be reached from cell A in one second are precisely the cells on the same floor as cell A that are adjacent to cell A. Any other cell adjacent to one of these cells and on the same floor can be reached in two seconds from cell A. The same idea allows us to find all the cells that can be reached in a minimum of 3 or 4 seconds. To find all the cells that can be reached from cell A in a minimum of 5 seconds, we need to also consider the cell on the other floor corresponding to cell A. The cells that can be reached from cell A in a minimum of 6 seconds are those adjacent to a cell on the same floor that can be reached from cell A in a minimum of 5 seconds, and any corresponding cells on the other floor that can be reached in 1 second. The result of our findings up to this point is illustrated below.

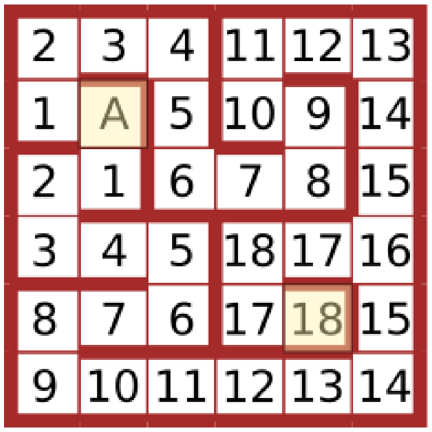


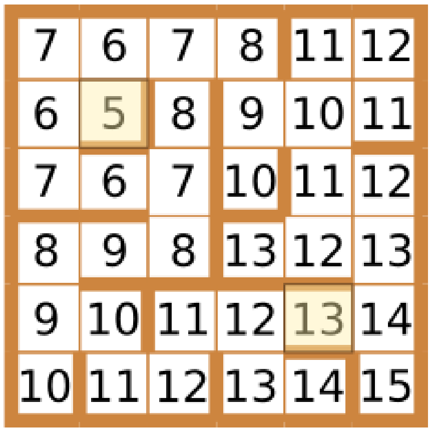
We continue in this way. In general, we know that a cell can be reached from cell A in a minimum of n seconds if it is adjacent to a cell on the same floor that can be reached from A in a minimum of $n-1$ seconds or if it corresponds to a cell on the other floor that can be reached from cell A in $n-5$ seconds. Below we show our findings later in this process. It is worth paying special attention to the 11 in the cell in the 5th row and 3rd column of the second floor.



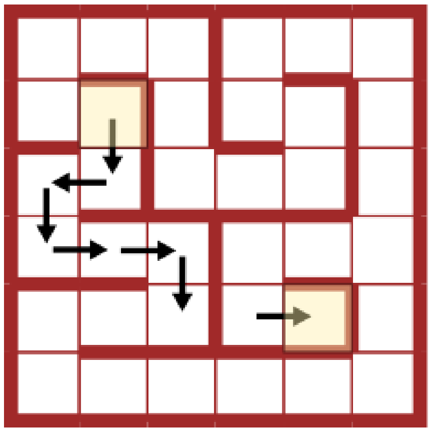
Explanation of Answer Continued

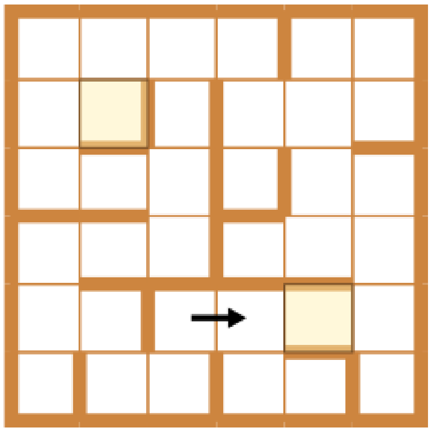
At the completion of this process, we determine the shortest time needed for the beaver to reach each cell from cell A:





We see that the shortest time needed to reach cell B from cell A is 18 seconds. The path that takes 18 seconds for the beaver to reach cell B from A is shown below.





Connections to Computer Science

The *shortest path problem* is a well-studied *graph theory* problem. The goal of the shortest path problem is to find the shortest distance between two *vertices* in the given *graph*. Solving the shortest path problem underlies routing *packets* in a computer network, finding the best route to take on a city map, etc.

This task focusses on a *grid graph*, where each vertex can be thought of as a cell in a 2-dimensional grid. Solving the shortest path problem on grid graph has various applications: specifically *wire routing in VLSI (very large scale integration) design* of circuits.

Common VLSI circuits are placed on a 10-layer silicon crystal while this task focusses only on a 2-layer problem. In multilayer VLSI circuits, *cross-level wiring* (moving between layers) is more expensive than *intra-layer wiring* (moving around a single layer). This task described moving between floors takes more time (5 seconds) compared to moving within one floor (1 second), which is analagous to what happens in real-world VLSI construction.

VLSI design involves routing millions and billions of wires to connect transistors, inputs, outputs and memory cells together to get VLSI circuit for digital devices, such as cellphones.

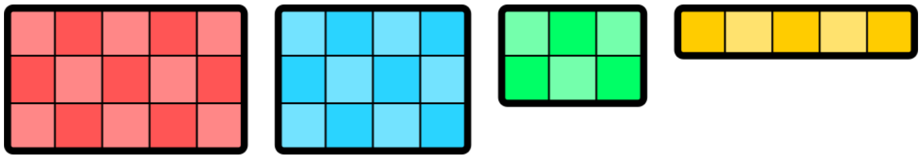
Country of Original Author

Uzbekistan

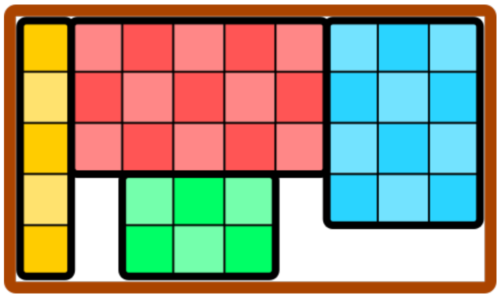
Variety Pack

Story

A company sells four different bottled drinks. The bottles are all identical in shape and size, but different drinks have different coloured labels. The red drink is always packaged in a 3 by 5 crate holding 15 bottles, the blue drink in a 3 by 4 crate holding 12 bottles, the green drink in a 2 by 3 crate holding 6 bottles, and the yellow drink in a 1 by 5 crate holding 5 bottles.



The company wants to sell a "Variety Pack" that includes exactly one crate of each of the four drinks. The Variety Pack is to be packaged in a rectangular container with all four drink crates placed flat on the base of the container. The following diagram shows how a Variety Pack can be made using a rectangular container that is 5 bottles wide and 9 bottles long.



Notice that 7 additional bottles would need to be placed in this container in order to fill the area of the base.

Question

Suppose that a rectangular container is chosen for the Variety Pack so that the four drink crates can be packaged with the least possible amount of empty space on the base of the container. In this case, how many additional bottles would need to be placed in the container in order to fill the area of the base?

(A) 1

(B) 2

(C) 4

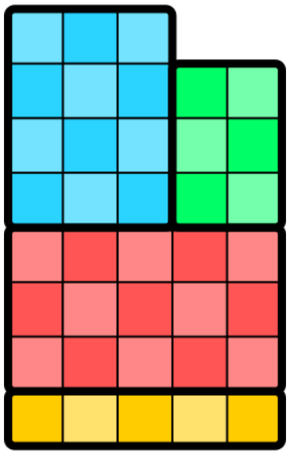
(D) 6

Answer

(B) 2

Explanation of Answer

A possible arrangement of crates that that only needs 2 bottles to fill the area of its base is shown below.



Even though Option A gives the smallest value, we should argue why the correct answer is not 0 or 1.

The number of bottles in all 4 crates put together is $15 + 12 + 6 + 5 = 38$. A container that holds 38 bottles with 0 gaps must have dimensions 1 by 38 or 2 by 19. You will never be able to fit the 3 by 5 crate (or the 3 by 4 crate) in this container.

A container with 1 gap would hold 39 bottles. There are two possibilities for its dimensions. One possibility is 1 by 39 which can hold the 1 by 5 crate but none of the other crates. The other possibility is 3 by 13. The two largest crates would take up 9 rows within such a container. The remaining 4 rows are not enough to place the smallest crate of size 1 by 5.

Therefore 2 gaps is the minimum we can have in any container, and we know this can be achieved.

Connections to Computer Science

This task is an example of the *bin packing* problem. Bin packing is an important practical problem – for instance, loading as many boxes as possible onto a truck, or marking out parking places to allow as many cars as possible to park.

Bin packing is an example of an *optimization problem*: we want to find the best or *optimal* solution, as quickly as possible. Unfortunately, it can be very *inefficient* to find the best solution for bin packing when the number of boxes is large, even if we use very powerful computers. Instead, we can use algorithms that find only *approximate solutions*: these solutions are within some known factor of the optimal solution, e.g., within 50% of the optimal solution. Note that given a choice between taking years of computation to find the best solution, and taking minutes to find an approximate solution, real-world applications suggest to use the approximate solution.

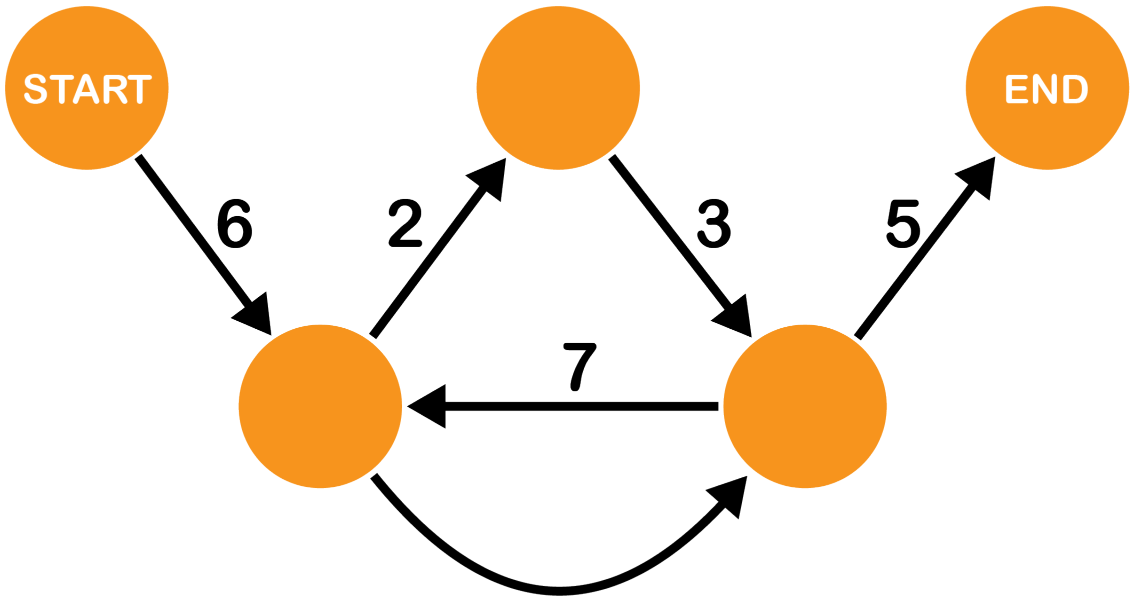
Country of Original Author

Netherlands

Numbers

Story

Birgit is creating numbers using the following diagram:



To create a number they start in the circle labelled START and then they follow arrows until they reach the circle labelled END.

If the arrow they follow is labelled with a digit, they write down that digit as part of their number. One arrow is unlabelled which means Birgit can follow it without writing down any digit.

For example, Birgit can create the number 6235 and the number 67775, among others.

Question

How many different 8-digit numbers can Birgit create?

(A) 5

(B) 9

(C) 12

(D) 14

Answer

(B) 9

Explanation of Answer

Notice that Birgit's numbers must always start with the digit 6 and they must always end with the digit 5, with some 2s, 3s, and 7s in between.

The arrow labelled with the digit 7 together with the unlabelled arrow allow Birgit to add as many 7s to their number as they like immediately after the digit 6.

Similarly, Birgit can add as many 7s as they like immediately before the digit 5.

The arrows labelled with the digits 2 and 3 indicate that Birgit can add the pair 23 to their number, both before and after adding 7s. They can also add multiple pairs of 23, but there must be at least one 7 between pairs. This is because after following the arrow labelled with 3, the only way to move "back" in the diagram is by following the arrow labelled with 7.

Focusing on the number of 23 pairs in Birgit's numbers, let's enumerate all the possible 8-digit numbers Birgit can create.

- No 23 pair: 67777775
- One 23 pair: 62377775, 67237775, 67723775, 67772375, 67777235
- Two 23 pairs (with one 7 between them): 62372375, 67237235
- Two 23 pairs (with two 7s between them): 62377235

It is not possible to have two 23 pairs with three or more 7s between them. This would create a number that is at least 9 digits long.

It is also not possible to have three or more 23 pairs with at least one 7 between them. This would create a number that is at least 10 digits long.

Therefore, Birgit can create 9 different 8-digit numbers.

Connections to Computer Science

The idea behind this problem is that the circles correspond to different *states*, and we can change states by following a *transition* labelled with the appropriate number. In general, this type of object is called a *finite automaton* or *finite state machine (FSM)*.

FSMs show up in more places than you might expect. For example, a traffic light has different states (e.g., red, yellow, or green) and changes between the states based on its environment (e.g., a car is waiting at an intersection). Many devices with buttons (e.g., a coffee machine or automated teller machine) move between states with the press of a button. Careful design of states and transitions allow computer programmers to program devices like this in efficient and reliable ways.

Our task also focusses on generating *words* from the given FSM. These *words* form the *language* specified by the FSM, and are part of the *lexical analysis*, which is part of *tokenizing* input. For example, postal codes and telephone numbers have certain patterns, and those patterns can used in developing a FSM that *recognizes* valid postal codes and telephone numbers.

Country of Original Author

United States