

Bebras Australia

Computational Thinking Challenge

2024 Solutions Guide Round 2
Secondary Divisions | **Years 7 to 12**



Bebras Australia

Computational Thinking Challenge

Bebras is an international initiative aiming to promote Computational Thinking skills among students.

Bebras is a fun and engaging computational thinking challenge for students in Years 3 to 12. It is an international challenge that involves over 2.9 million students from 60 countries.

Bebras creates opportunities for students to engage in activities that use and develop their critical and creative thinking and problem-solving skills essential to further learning.

Coding skills are not required to complete the Bebras Challenge. The challenge is open twice a year for three weeks and each round has different questions. Students can participate individually or in teams of up to four.

Lithuanian for beaver, Bebras was the name chosen by the founder of the challenge, Professor Valentina Dagiene from the University of Vilnius, in honour of the animal's collaborative nature and strong work ethic.

The Challenge is coordinated by the International Bebras Committee which meets annually to assess potential questions and share resources. Bebras Australia began in 2014 and was delivered by CSIRO until 2023. From 2024, Bebras Australia is delivered by the Australian Maths Trust.

With support from CSIRO, the AMT provided the Bebras Challenge free of charge for Australian schools in 2024.

To find out more and register for the next challenge, visit www.amt.edu.au



Contents

What is a Solutions Guide?..... v

What is Computational Thinking? vi

Computational Thinking
skills alignment..... vii

Digital Technologies
curriculum key concepts ix

Digital Technologies
key concepts alignment x

2024 Bebras Challenge
Round 2 | Junior Year 7 & 8..... 2

Favourite Drinks.....3

Frog and Mosquito.....5

Maze game.....7

Sprinklers.....9

Gift selection.....12

Robot on the path.....14

BebrasGPT16

Stamp Machines.....18

Sealed Letters.....20

Snail Compress.....22

Watercolour.....24

Jumping Together.....26

Correcting error29

Bebras Ball31

Triangle.....33

2024 Bebras Challenge
Round 2 | Intermediate Year 9 & 1036

Gift selection.....37

Robot on the path.....39

BebrasGPT41

Stamp Machines.....43

Sealed Letters.....45

Snail Compress.....47

Watercolour.....49

Jumping Together.....51

Correcting error54

Bebras Ball56

Triangle.....58

Open It.....60

Channel plan.....64

Palindrome Passwords67

Line Up69

2024 Bebras Challenge
Round 2 | Senior Year 11 & 12 74

BebrasGPT75

Snail Compress.....77

Watercolour.....79

Jumping Together.....81

Correcting error84

Bebras Ball86

Triangle.....88

Open It.....90

Channel plan.....94

Palindrome Passwords97

Mapping the roads.....99

Line Up101

Logs to the warehouse.....106

Walking in the Forest.....109

Painting doors112

2024 Round 2 All 117

What is a Solutions Guide?

Within this Solutions Guide you will find all of the questions and tasks from Round 2 of the Secondary divisions of the 2024 Bebras Australia Computational Thinking Challenge.

On each page you will find a question, answer, an explanation, and background information explaining the skills and key curriculum concepts featured.



What is Computational Thinking?

Computational Thinking is a set of skills that underpin learning within the Digital Technologies classroom. These skills allow students to engage with processes, techniques and digital systems to create improved solutions to address specific problems, opportunities or needs.

Computational Thinking uses a number of skills, including:



DECOMPOSITION

Breaking down problems into smaller, easier parts.



PATTERN RECOGNITION

Using patterns in information to solve problems.



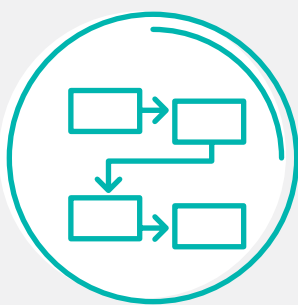
ABSTRACTION

Finding information that is useful and taking away any information that is unhelpful.



MODELLING AND SIMULATION

Trying out different solutions or tracing the path of information to solve problems.



ALGORITHMS

Creating a set of instructions for solving a problem or completing a task.



EVALUATION

Assessing a solution to a problem and using that information again on new problems.

Visit the AMT website for more Bebras resources <https://www.amt.edu.au/bebras>



Computational Thinking skills alignment

		Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Question	Difficulty						
Junior Year 7 & 8							
Favourite Drinks	Easy	○		○		○	○
Frog and Mosquito	Easy	○		○	○	○	○
Maze game	Easy	○		○	○	○	○
Sprinklers	Easy	○	○	○		○	○
Gift selection	Easy	○		○		○	○
Robot on the path	Medium	○		○	○	○	○
BebrasGPT	Medium	○		○		○	○
Stamp Machines	Medium	○	○	○		○	○
Sealed Letters	Medium	○		○			○
Snail Compress	Medium	○		○		○	○
Watercolour	Hard	○		○	○	○	○
Jumping Together	Hard	○	○	○	○		○
Correcting error	Hard	○				○	○
Bebras Ball	Hard	○		○	○	○	○
Triangle	Hard	○	○		○	○	
Intermediate Year 9 & 10							
Gift selection	Easy	○		○		○	○
Robot on the path	Easy	○		○	○	○	○
BebrasGPT	Easy	○		○		○	○
Stamp Machines	Easy	○	○	○		○	○
Sealed Letters	Easy	○		○			○
Snail Compress	Medium	○		○		○	○
Watercolour	Medium	○		○	○	○	○
Jumping Together	Medium	○	○	○	○		○
Correcting error	Medium	○				○	○
Bebras Ball	Medium	○		○	○	○	○
Triangle	Hard	○	○		○	○	
Open It	Hard	○	○	○			○
Channel plan	Hard	○		○	○		○
Palindrome Passwords	Hard	○	○	○		○	○
Line Up	Hard	○		○		○	○

		Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Question	Difficulty						
Senior Year 11 & 12							
BebrasGPT	Easy	○		○		○	○
Snail Compress	Easy	○		○		○	○
Watercolour	Easy	○		○	○	○	○
Jumping Together	Easy	○	○	○	○		○
Correcting error	Easy	○				○	○
Bebras Ball	Medium	○		○	○	○	○
Triangle	Medium	○	○		○	○	
Open It	Medium	○	○	○			○
Channel plan	Medium	○		○	○		○
Palindrome Passwords	Medium	○	○	○		○	○
Mapping the roads	Hard	○		○	○		○
Line Up	Hard	○		○		○	○
Logs to the warehouse	Hard	○	○	○			○
Walking in the Forest	Hard	○		○	○	○	○
Painting doors	Hard	○	○	○		○	○

Digital Technologies curriculum key concepts

Abstraction

Hiding details of an idea, problem or solution that are not relevant, to focus on a manageable number of aspects.

Data Collection

Numerical, categorical, or structured values collected or calculated to create information, e.g. the Census.

Data Representation

How data is represented and structured symbolically for storage and communication, by people and in digital systems.

Data Interpretation

The process of extracting meaning from data. Methods include modelling, statistical analysis, and visualisation.

Specification

Defining a problem precisely and clearly, identifying the requirements, and breaking it down into manageable pieces.

Algorithms

The precise sequence of steps and decisions needed to solve a problem. They often involve iterative (repeated) processes.

Implementation

The automation of an algorithm, typically by writing a computer program (coding) or using appropriate software.

Digital Systems

A system that processes data in binary, made up of hardware, controlled by software, and connected to form networks.

Interactions

Human-Human Interactions: How users use digital systems to communicate and collaborate.

Human-Computer Interactions: How users experience and interface with digital systems.

Impact

Analysing and predicting how existing and created systems meet needs, affect people, and change society and the world.

For more information on the Digital Technologies curriculum, please visit the Australian Curriculum, Assessment and Reporting Authority (ACARA) website: australiancurriculum.edu.au/f-10-curriculum/technologies/digital-technologies




Digital Technologies

key concepts alignment

key concepts alignment		Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Question	Difficulty										
Junior Year 7 & 8											
Favourite Drinks	Easy	○	○	○		○	○	○			
Frog and Mosquito	Easy	○					○	○	○		
Maze game	Easy	○					○	○		○	
Sprinklers	Easy	○		○	○	○	○	○			
Gift selection	Easy	○					○	○	○		
Robot on the path	Medium	○				○	○	○	○		
BebrasGPT	Medium	○					○	○	○		○
Stamp Machines	Medium	○			○		○	○			
Sealed Letters	Medium	○	○			○					○
Snail Compress	Medium	○		○			○	○	○		
Watercolour	Hard	○					○	○		○	
Jumping Together	Hard	○					○	○	○	○	
Correcting error	Hard			○			○	○	○		○
Bebras Ball	Hard	○					○	○	○		
Triangle	Hard		○		○		○		○		
Intermediate Year 9 & 10											
Gift selection	Easy	○					○	○	○		
Robot on the path	Easy	○				○	○	○	○		
BebrasGPT	Easy	○					○	○	○		○
Stamp Machines	Easy	○			○		○	○			
Sealed Letters	Easy	○	○			○					○
Snail Compress	Medium	○		○			○	○	○		
Watercolour	Medium	○					○	○		○	
Jumping Together	Medium	○					○	○	○	○	
Correcting error	Medium			○			○	○	○		○
Bebras Ball	Medium	○					○	○	○		
Triangle	Hard		○		○		○		○		
Open It	Hard	○		○				○			
Channel plan	Hard	○				○	○		○	○	
Palindrome Passwords	Hard	○		○			○	○	○		
Line Up	Hard	○					○	○			

Question	Difficulty	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Senior Year 11 & 12											
BebrasGPT	Easy	○					○	○	○		○
Snail Compress	Easy	○		○			○	○	○		
Watercolour	Easy	○					○	○		○	
Jumping Together	Easy	○					○	○	○	○	
Correcting error	Easy			○			○	○	○		○
Bebras Ball	Medium	○					○	○	○		
Triangle	Medium		○		○		○		○		
Open It	Medium	○		○				○			
Channel plan	Medium	○				○	○		○	○	
Palindrome Passwords	Medium	○		○			○	○	○		
Mapping the roads	Hard	○	○			○	○	○	○	○	○
Line Up	Hard	○					○	○			
Logs to the warehouse	Hard	○					○	○			
Walking in the Forest	Hard	○	○				○	○	○		
Painting doors	Hard	○					○	○	○		

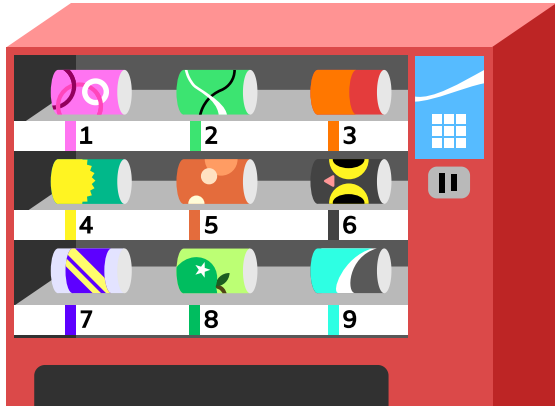


The background is a solid dark blue color. In the upper left corner, there are several overlapping organic, blob-like shapes in lighter shades of blue, ranging from a medium blue to a very light, almost white-blue. These shapes are positioned in the top left and top center of the page.

2024 Bebras Challenge Round 2 | Junior Year 7 & 8

Favourite Drinks

Damian, Nabil and Jasmine always buy their favourite drinks from a vending machine in their work place. Each of them have three favourites from the numbered nine options.



On Monday, only Damian and Nabil worked. Together they bought drinks 1, 2, 3, 5, 8 and 9.

On Tuesday, only Nabil and Jasmine worked. Together they bought drinks 1, 4, 5, 6, 7 and 9.

Task

What drink are their favourites? Put marks in the appropriate cells below by clicking on them.

Click again to remove the mark. Click "Save" when you are done.

	1	2	3	4	5	6	7	8	9
Damian									
Nabil									
Jasmine									

Save

Erase

Explanation

	1	2	3	4	5	6	7	8	9
Damian		✓	✓					✓	
Nabil	✓				✓				✓
Jasmine				✓		✓	✓		

Monday: Damian and Nabil, drinks 1, 2, 3, 5, 8 and 9.

Tuesday: Nabil and Jasmine, drinks 1, 4, 5, 6, 7 and 9.

Since Nabil worked on both days, his favourites must be 1, 5 and 9, as these were repeated on both days.

Accordingly, on Monday, Damian bought drinks 2, 3, 8. On Tuesday, Jasmine bought drinks 4, 6, 7.

Background information

The idea of the task is to be able to analyze complex logical statements and draw correct conclusions based on them. The basic logic operations on the number lists are very important in Informatics. That gives a nice meaning to the basic logical operations (identical to the set operations).

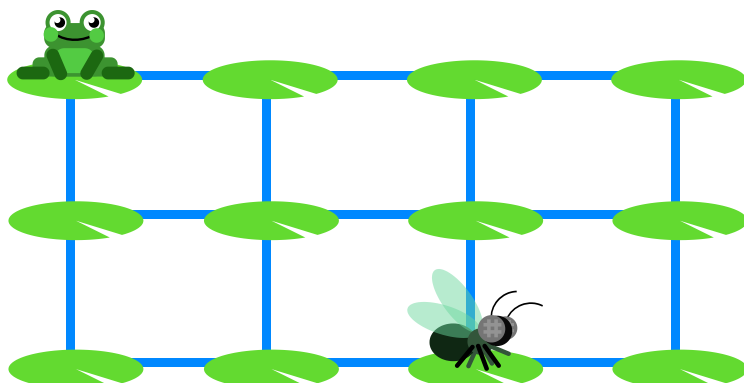
It is sometimes hard to know where to start when you are faced with a complicated problem. In solving this task it is important to find an optimal order of checking logical statements. Breaking the problem into many smaller ones (decomposition) can help you to decide where to start.

Logic is one of the computational thinking skills you use when solving this task. You are given information about nine items bought from a vending machine on two separate days by three people who have three favorite items each. From that information, you have to use *deductive reasoning* to figure out what three items are each person's favorites. It is very helpful to use a *truth table* to solve this type of problem. In this case the truth table has a row for each person and a column for each vending machine item. Your task is to decide for each cell is that particular item that person's favorite. Or in logic we would say: Is the statement "this item is this person's favorite" TRUE or FALSE.

Frog and Mosquito

A frog is sitting on a water lily in the top left corner as shown below. It wants to get to the water lily with the mosquito.

The frog can jump from water lily to water lily along a straight blue line in one jump. It may jump to each water lily only once.



Question

In how many different ways can the frog reach the water lily with the mosquito if it can jump **exactly four times**?

Fill in the number and click "Save" when you are done.

Answer:

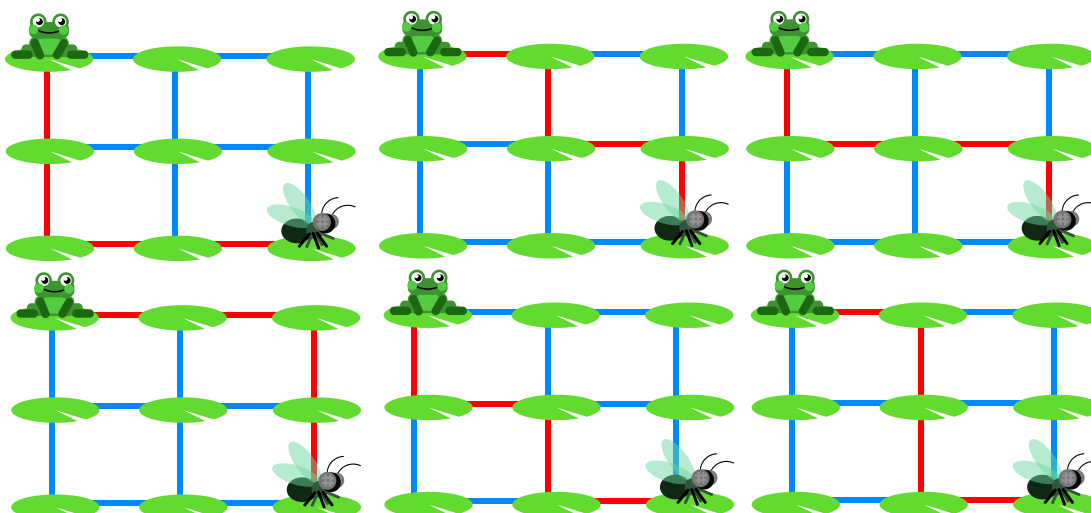
Save

Explanation

Correct answer is 6.

All possible paths that can lead the frog to the water lily with the mosquito along the blue line in four jumps are shown in the six pictures below.

Every other path that brings the frog to the water lily with the mosquito contains more than 4 jumps, or includes diagonal jumps which are forbidden.



Background information

One way to solve this task is to try all possible paths from START to STOP. This is an *algorithm* which uses a technique called *brute force*. It works but it usually takes a long time!

Computer scientists have several different ways to solve this type of task which normally take less time especially in a version of this game that included a much larger terrain with many more squares. The technique used in the explanation of the answer is called *dynamic programming*.

When solving this problem, you might have considered using a different technique. Perhaps you always tried to move to the next square (below or to the right) that would use the least amount of energy in the next step. People call this a *greedy algorithm*. Sometimes greedy algorithms do work but here, this particular idea is incorrect and will not find an optimal path. *Dijkstra's Algorithm* is famous among computer scientists. It combines the ideas of dynamic programming and greedy algorithms and can be used to find optimal paths to all the squares.

Computational thinking is the application of methods and techniques derived from computer science in other areas of life. Dynamic programming is applied to optimization problems. Such problems occur in all areas of life. The situation from the task can be related to the movement of the object in certain directions with the smallest possible loss for a certain parameter. For example, it could be a passenger plane taking off or landing with as little fuel consumption as possible. The task also refers to the algorithms hidden in the background of board games.

Maze game

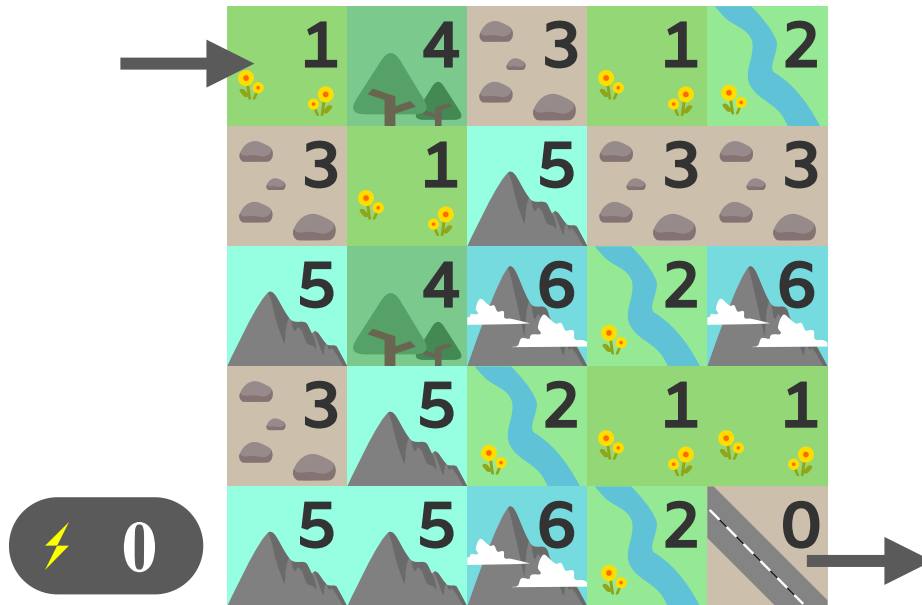
A crocodile is playing a board game. The game is played on a board divided into squares. Numbers in each square show how much energy the crocodile uses when it goes over the square.

The crocodile can only move down and right.

Task

Find the path from the starting arrow to the ending arrow through which the crocodile uses the least amount of energy.

Click on each path section to select it. Press "Save" when you are done.

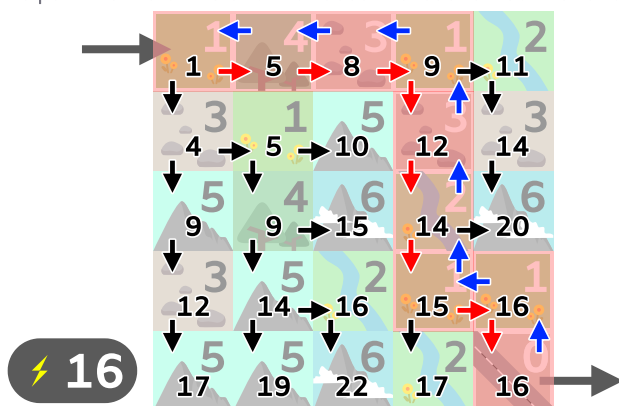


Save

Erase

Explanation

The path from start to end that uses the least amount of energy is highlighted below.



On this path, the amount of energy that the crocodile uses is $1+4+3+1+3+2+1+1+0 = 16$.

We should convince ourselves that 16 is the least possible energy used to go from start to end. To do this, we will call any path between two squares *optimal* if it is a path between them that uses the least possible amount of energy.

In the figure, the black numbers in the bottom of each square show how much energy is used in an optimal path to the square. If these numbers are correct, then we know the correct answer is 16. Why are these numbers correct?

Well, consider any square. Any optimal path to this square must begin with an optimal path to the square just above or just to the left. Therefore, once we know the optimal path to both of these squares, the one between them that requires the least energy determines an optimal path to the square under consideration. This is indicated by the arrows in the figure above. These arrows tell us which direction an optimal path to each square comes from.

We can also calculate how much energy is used along all these optimal paths. For example, look at the fifth square along the path of our answer to this problem (in the second row and fourth column). The amount of energy used in an optimal path to the square above is 9 and the amount of energy used in an optimal path to the square to the left is 10. Since $9 < 10$, this means the optimal path to the third square on our path must come from above and the amount of energy it uses is $9+3=12$. We can use this idea to work right and downwards from the starting square to determine an optimal path from start to end and how much energy the crocodile uses to take this path.

Background information

One way to solve this task is to try all possible paths from START to STOP. This is an *algorithm* which uses a technique called *brute force*. It works but it usually takes a long time!

Computer scientists have several different ways to solve this type of task which normally take less time especially in a version of this game that included a much larger terrain with many more squares. The technique used in the explanation of the answer is called *dynamic programming*.

When solving this problem, you might have considered using a different technique. Perhaps you always tried to move to the next square (below or to the right) that would use the least amount of energy in the next step. People call this a *greedy algorithm*. Sometimes greedy algorithms do work but here, this particular idea is incorrect and will not find an optimal path. *Dijkstra's Algorithm* is famous among computer scientists. It combines the ideas of dynamic programming and greedy algorithms and can be used to find optimal paths to all the squares.

Computational thinking is the application of methods and techniques derived from computer science in other areas of life. Dynamic programming is applied to optimization problems. Such problems occur in all areas of life. The situation from the task can be related to the movement of the object in certain directions with the smallest possible loss for a certain parameter. For example, it could be a passenger plane taking off or landing with as little fuel consumption as possible. The task also refers to the algorithms hidden in the background of board games.

Sprinklers

Amanda is planning to install new sprinklers to water the crops on her farm. The farm land is divided into squares, a map of which is shown below:



The store sells sprinklers that can water 4×4 , 2×2 , or 1×1 squares of land. There are four types of crops on Amanda's farm: apples, corn, grapes, and mushrooms. Each type of crop needs to be watered differently to promote ideal growth. Therefore, if a sprinkler waters a particular patch of land, then all the crops on that patch should be of the same type. Also, each patch is watered by only one sprinkler.

Question

What is the minimum number of sprinklers that Amanda needs in order to water all the crops on her farm?

19

21

23

25

27

Explanation

The correct answer is (D) 25. In order to minimize the number of sprinklers needed to water the farm, the patch of land watered by each sprinkler should be as large as possible — while ensuring that (1) all the crops inside a patch are of the same type, (2) the patches do not overlap, and (3) sprinklers can overlap the patches they water. The figure below shows one way this can be done:



Maximizing the size of the patch for each crop watered by a sprinkler, given the constraint of the size of a patch (4x4, 2x2, 1x1) that different sprinklers can water, does result in minimizing the number of sprinklers needed to water the farm.

Starting with the patch at the top lefthand corner of the farm. For the same crop select the largest patch which can be sprinkled (4x4, 2x2, 1x1). When the appropriate sprinkler has been placed, then move to the next free right patch and select the appropriate sprinkler for crops of a similar type. If the border of the farm is reached, then start the process again with the next row of crops. Repeat this process until all patches have a sprinkler associated with them.

If sprinklers are chosen that water 1x1 patches only, then a minimum of 64 sprinklers would be needed. As the farm can be represented by 64 patches.

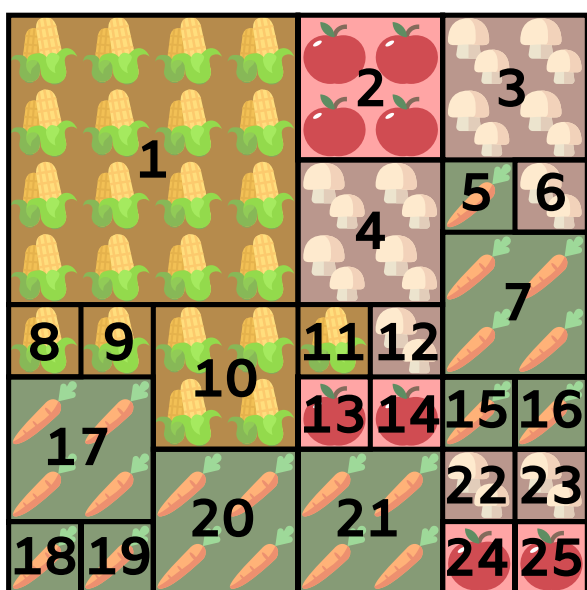
If sprinklers are chosen that water both 2x2 patches and 1x1 patches only. Starting with placing sprinklers that water patches of size 2x2, then placing sprinklers that water 1x1 patches. Then a minimum of 28 sprinklers would be needed. As the farm can be represented by 12 2x2 patches and 16 1x1 patches.

If sprinklers are chosen that water 4x4, 2x2 and 1x1 patches. Starting with placing sprinklers that water patches of size 4x4, then placing sprinklers that water patches of size 2x2, and finally placing sprinklers that watch 1x1 patches. As the farm can be represented by 1 4x4 patch, 8 2x2 patches and 16 1x1 patches. Then a minimum of 25 sprinklers would be needed.

Background information

This challenge is an example of an *optimization problem*, a task with the goal of maximizing or minimizing a target value given a set of restrictions. Several optimization problems can be solved by following the strategy of always selecting the *local optimum* (the best choice at each step) and hoping that doing so will lead to the *global optimum* (the best result in the end). This problem-solving paradigm characterizes *greedy algorithms*. A greedy algorithm is any algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage. In many problems, a greedy strategy does not produce an optimal solution, but a greedy heuristic can yield locally optimal solutions that approximate a globally optimal solution in a reasonable amount of time.

Aside from the greedy algorithm, the solution touches on another informatics concept known as a *quadtree*. A quadtree is a way to represent an image by dividing it into four equal square regions. A region can further be divided into four regions until each region contains exactly one color – similar to how the solution divides the farm into regions until each region encloses exactly one fruit. For example, look at the picture below:



Quadtrees thus allow images to be stored as a collection of regions rather than individual pixels, resulting in smaller storage requirements and faster operations. This idea can also be extended to other two-dimensional data, such as maps, making quadtrees useful in image processing, game development, and navigation, among other applications.

An important skill in computational thinking is *tinkering*. The big picture of this challenge, which is to minimize the number of sprinklers needed to water a farm, can be a bit overwhelming. Hence, it may be helpful to explore some "what-if" questions first. In this challenge, students may start by tinkering with either small or large patches. Asking "What if I start with 2x2 patches?" motivates combining some patches into 4x4 patches and so on. Meanwhile, asking "What if I start with 8x8 patches?" opens the realization that some patches have to be divided into 4x4 patches and so on. Although these insights may not be necessarily structured at first, they are key to formulating the greedy algorithm for this challenge.

Gift selection

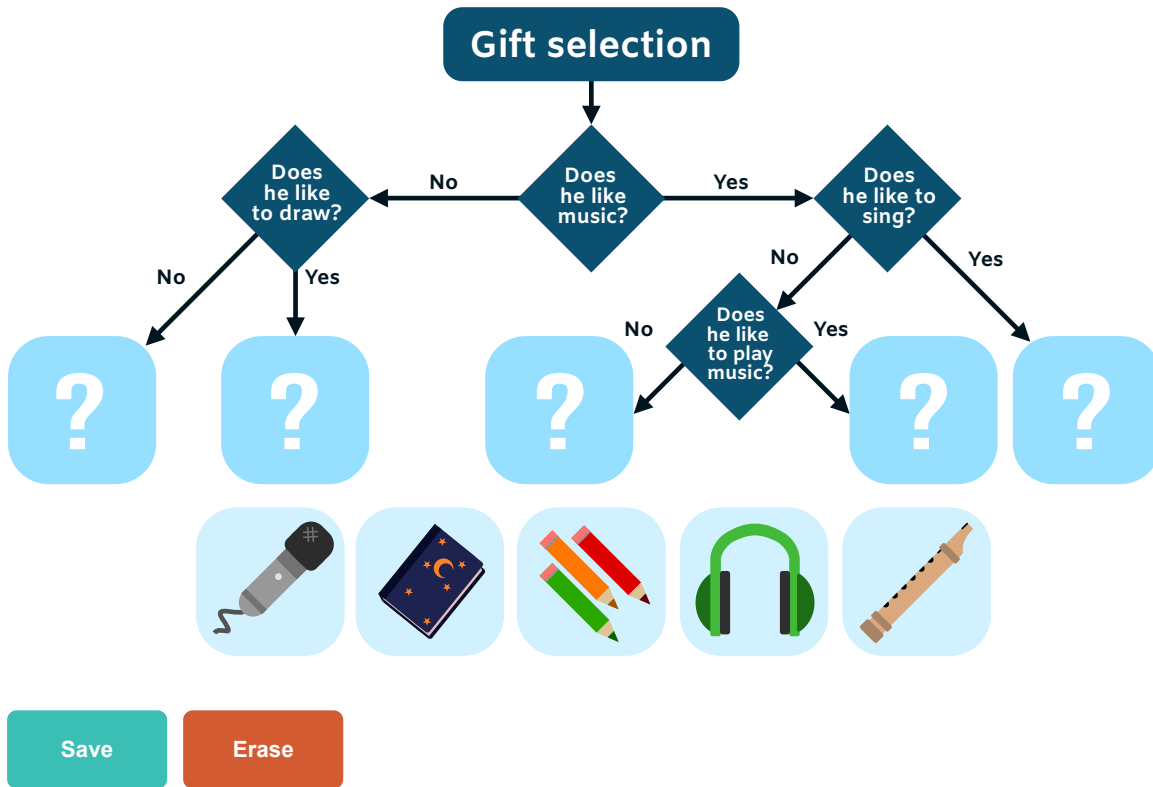
A new student has transferred to Madina's class. They want to surprise him with a gift.

In order to pick the right gift, they ask him a few questions. The flow chart below shows how they ask questions:

Task

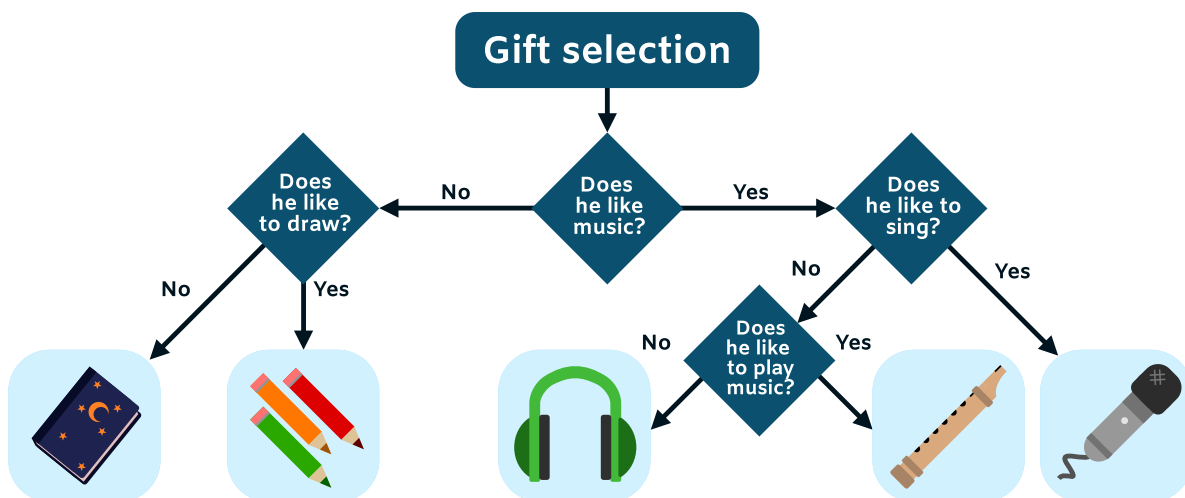
Help Madina's class decide which gifts would best fit the new student's possible answers.

Drag the gifts to the empty boxes and click "Save" when you are done.



Explanation

The following diagram illustrates the correct solution:



If he loves music, they can give him a microphone, recorder, and headphones. If he likes to sing, place the microphone in position E. If he doesn't like to sing but likes to play music, place the recorder in position D and the headphones in position C.

If he doesn't like music but likes to draw, place the colour pencils in position B. Otherwise, place the diary in position A.

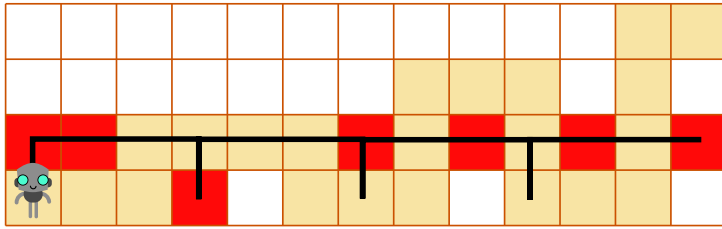
Background information

A decision tree is like a map to help you make decisions (in our example we used the decision tree to place gifts based on new student's answers).

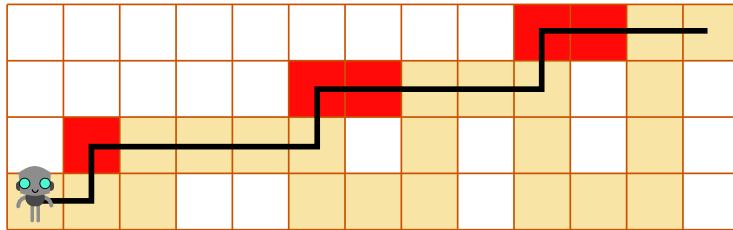
It starts with a key question, and each answer can lead to either more questions, or a final answer.

Computer programs can often be visualised as decision trees. A program can be given an input, and then we use code to specify questions to ask of the input, which ultimately leads with a result that can be output.

If it were to execute commands C) **URRRD**, it would go off the path at **7 squares**:



If it were executing commands D) **RURRR**, it would go off the path at **5 squares**:



Background information

The path of the robot is described by a sequence of 5 commands that the robot repeats. According to the commands, the robot changes its position. Executing a (repeating) sequence of commands is one of the fundamental ideas in programming.

Commands control the robot in this problem by moving it in one of four directions, so-called "absolute control", without the robot turning in that direction.

BebrasGPT

BebrasGPT is a chat bot that has been developed to produce three-word sentences. It does this by predicting the next word based on the previous sequence of words. Each word is chosen one by one, with the next word being chosen based on the probability of it following the current sequence. The tables below show some of these probabilities.

Probabilities for second word:

	"love"	"hate"
"Cats"	0.7	0.3
"Dogs"	0.6	0.4

Probabilities for third word:

	"swimming"	"running"
"Cats love"	0.3	0.8
"Cats hate"	0.9	0.1
"Dogs love"	0.7	0.3
"Dogs hate"	0.1	0.9

For example, if the sentence starts with the word "Cats", the probability of the 3-word sentence being "Cats love running" is 0.56 because:

- the probability of the second word being "love" if the previous word is "Dogs" is 0.7,
- and the probability of the next word being "running" if the previous sequence is "Dogs love" is 0.8,
- so, because the model predicts words one by one, the probability is $0.7 * 0.8 = 0.56$.

Question

If a sentence starts with the word "Dogs" what is the most likely output of BebrasGPT?

"Dogs hate swimming"

"Dogs hate running"

"Dogs love swimming"

"Dogs love running"

Explanation

The correct answer is: "Dogs love swimming."

The probabilities of each sentence are the following:

"Dogs hate swimming", $0.4 * 0.1 = 0.04$

"Dogs hate running", $0.4 * 0.9 = 0.36$

"Dogs love swimming", $0.6 * 0.7 = \mathbf{0.42}$

"Dogs love running", $0.6 * 0.3 = 0.18$

C has the highest probability among the choices.

Background information

This exercise involves an example of an artificial model for language modeling and the production of text. This is a probabilistic model, meaning that each output of the model is based on probabilities. There are many types of models. In this case, the model is producing one word at a time, based on the entire sequence of previous words, which is similar to models like ChatGPT. Of course these models are much, much larger than the model shown in the exercise.

One related important aspect in artificial intelligence and machine learning is the difference between two type of algorithms:

- the learning algorithm,
- and the inference algorithm.

The learning algorithm is used to "train" the model. This corresponds, in our example, to finding which values of the probabilities should be in the table above. In our case, this has already been done. The model is already trained. Usually, this is the most difficult part in terms of theory. For example, ChatGPT was training for many months on several different GPUs, which cost millions of dollars.

The inference algorithm is the one we use after training, to produce the output. For example, when you type the prompt "Hi, ChatGPT. What is machine learning?", the model will take that sequence of words and use something equivalent to a very large table of probabilities to output the answer to the prompt. Although the cost of running the inference algorithm once is much smaller than the training, if the model is used by millions of people everyday (like ChatGPT), the cost of inference is higher than that of training (which was a one time cost).



This task requires algorithmic thinking as the students figure out the next word one at a time, and finding the 3-word sentence with the largest probability. Students are probably familiar with the text predict functions of their smartphones but this task allows them to think through how it works.

Stamp Machines

Paula has the following four machines, that she uses to make designs on paper:



These are the functions of the machines, in no particular order:

- Colour the whole paper black
- Turn the paper 180 degrees
- Stamp  on the paper
- Stamp  on the paper

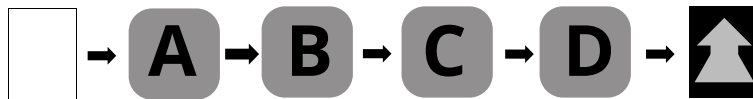
These machines can be used together to make special designs. At each step, a machine can completely change what was done before.

For example, the machine that colours the paper black can cover stamps made on the paper.

Paula initially puts a piece of white paper in the machines, uses the machines in the order shown below, and gets the design



at the end.



Question

Which machine stamps  on the paper?



Explanation

The answer is (D).

There are some conditions to obtain the final paper shown:

- The machine D can't be the one that colours the paper black, or else the final paper would be entirely black.
- Machine A can't be the one that turns the paper 180 degrees, or else the final paper would have an arrow upside down.
- The machine that colours the paper black is before any other that stamps the paper, or else this stamp wouldn't appear.
- The machine that stamps an arrow on the paper is before the machine that turns the paper 180 degrees, and the machine that turns the paper is before the machine that stamps a triangle on the paper.

By all these conditions, we conclude that machines that stamp an arrow on the paper, turn the paper, and stamp a triangle on the paper are in sequence. Then, the machine that colours the paper black is A, and the others B, C, D respectively.

Background information

In this task, each of the machines A, B, C, and D has a specific function and, when associated in different orders, might present different results. Furthermore, the overall result, such as presented in this Task (a black paper with two black figures), depends on a strategic association of the machines. Therefore, students should associate the machines in a sequence that allows them to obtain the expected composition on the paper. They must also consider that each step depends on the previous one – it means the task to be performed by one machine only makes sense if that machine is put after another specific machine. There is a dependency relationship here.

Arranging machines is like programming. For a program to solve a task perfectly, all the statements in the program must be correctly ordered. So, computer scientists must tell the computer what it must do step by step.

Algorithmic Thinking is essential to solve this task. Each machine has one specific function that can influence the other machine's function: for example, if the machine that colors the paper black is put after the machine that stamps a triangle on the paper, the second machine will make the triangle disappear (so this order does not work for this task). The main goal of this task is to discover what position each machine should be put in the sequence A, B, C, D, in order to get the final paper shown and this demands comprehension of how the machines work and how they are used. To some extent, the task is also related to reverse engineering, since the student needs to understand how a piece of the algorithm works based on the input given and output obtained.

Decomposition is also present in this Task. Instead of looking to the global process all at once, students can analyze each step to decide which machine should be placed after another in the association.

Sealed Letters

The Republic of Beaveria has a cabinet full of secret letters.

Among the 16 letters in this cabinet, numbered 1 to 16, 10 had been opened, while the other 6 were still inside their sealed envelopes.

One evening, an enemy spy snuck in and opened one of the sealed letters. However, they forgot to seal it again.

The next morning, the Republic of Beaveria goes into an investigation after noticing that there are now 11 opened letters, as shown in the diagram.

















The Republic's guard does not recall all the details, but they are sure that before the enemy spy sneaked in:

- The total number of opened letters in the C2 and C4 columns was even.
- The total number of opened letters in the C3 and C4 columns was even.
- The total number of opened letters in the R2 and R4 rows was even.
- The total number of opened letters in the R3 and R4 rows was even.

Question

Which letter was opened by the enemy spy?

Click on the letter and press "Save".

	C1	C2	C3	C4
R1	1 	2 	3 	4 
R2	5 	6 	7 	8 
R3	9 	10 	11 	12 
R4	13 	14 	15 	16 

Save

Erase

Explanation

The correct answer is 13.

We follow this line of reasoning:

- There is an even number of opened letters in the C2 and C4 columns combined, matching the guard's recollection. As there is only one letter opened by the spy, this implies that the letter opened by the spy must be either in the C1 or C3 column. There is an even number of opened letters in the C3 and C4 columns combined, matching the guard's recollection. Given the previous statement, this implies that the letter opened by the spy must be in the C1 column.
- There is an odd number of opened letters in the R2 and R4 rows combined, which does not match the guard's recollection. This implies that the letter opened by the spy must be either in the R2 or R4 row.
- There is an odd number of opened letters in the R3 and R4 rows combined, which does not match the guard's recollection. Given the previous statement, this implies that the letter opened by the spy must be in the R4 row.

Therefore, the letter read by the spy is in the C1 column and R4 row, which points to 13.

Background information

This challenge introduces the concept of *error-correcting* codes. Digital data is essentially a sequence of bits: 1s and 0s. When it passes through a network, it can be corrupted due to noise or due to the action of malicious entities. Data stored on DVDs may also become corrupted if these storage devices are scratched. It is thus important to have a scheme that is capable of not only detecting that corruption has occurred (*error detection*) but also correcting the corrupted bits (*error correction*).

The first modern error-correcting code was introduced by Richard Hamming in 1950. This challenge specifically borrows the idea of the [15, 11] *Hamming code*, with the closed and opened letters corresponding to 0s and 1s, respectively. This coding scheme takes an 11-bit data and inserts 4 *parity bits*. These parity bits occupy slots 2, 3, 5, and 9, while the bits of the original data occupy the remaining slots (except slot 1). The parity bits are set so that there is an even number of 1s in the 2nd and 4th columns, 3rd and 4th columns, 2nd and 4th rows, and 3rd and 4th rows. Finally, the bit at slot 1 is set so that there is an even number of 1s in total.

As seen in the presented solution, this scheme allows for the corrupted bit to be identified and subsequently corrected, provided that there is at most 1 corrupted bit. Although more sophisticated error-correcting schemes are necessary to correct 2 or more corrupted bits, the Hamming code's efficiency contributes to its continued usage in NAND flash memory chips embedded in mobile phones and solid-state drives.

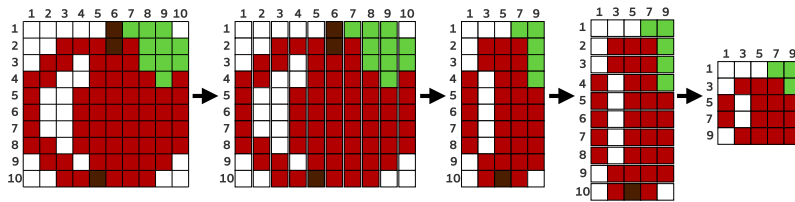
An important skill in computational thinking is *deductive reasoning* – logically deriving conclusions from a set of known facts or *premises*. A common method in deductive reasoning is the process of *elimination*. For instance, in this challenge, knowing that the 2nd and 4th columns combined have an even number of opened letters eliminates the possibility that they have the letter read by the spy, thus narrowing down the choices to the 1st and 3rd columns. This ability to reason in a systematic, top-down manner is a core competency in designing algorithms, tracing code, and debugging programs without resorting to guesswork or committing logical fallacies.

Snail Compress

Snails have a special technique to shrink their captured images.

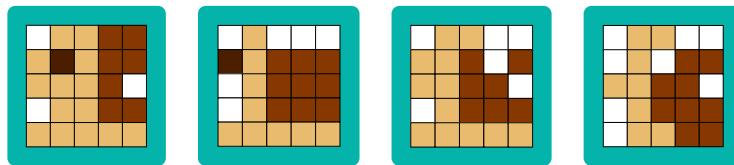
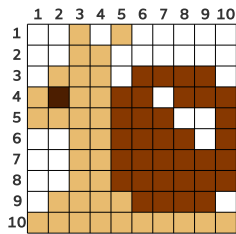
First, they will cut the original image into 10 equally sized strips vertically. Then, they will assemble the odd-numbered vertical strips to create a new image.

Next, they will cut the new image horizontally into 10 equally sized strips. Then, they will assemble the odd-numbered horizontal strips to create a complete shrunken image.



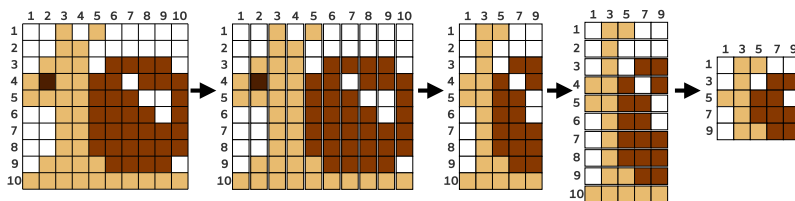
Question

What image will the snails obtain after shrinking the given image using this technique?



Explanation

Following the same procedure as with the picture of the apple, the snail image is compressed in the following way:



First all the even-numbered columns will be erased, then all the even-numbered rows will be erased, which means that only the cells from odd-numbered columns, that are also in odd-numbered rows will remain.

We can see that the correct answer is D, as it is the only image that uses the 9th row for compression of the picture.

Background information

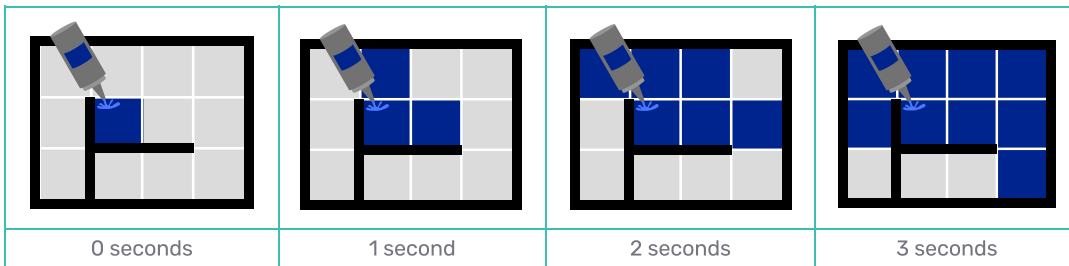
Compression of an image refers to the process of reducing the size of an image file by eliminating or minimising some of the unnecessary data in the image while still maintaining sufficient image quality for use. This process helps to reduce the file size of images, making them easier to store or transmit.

This is lossy compression algorithm, as it completely eliminates some of the rows and columns in the original image. JPG images are also stored using a lossy compression algorithm.

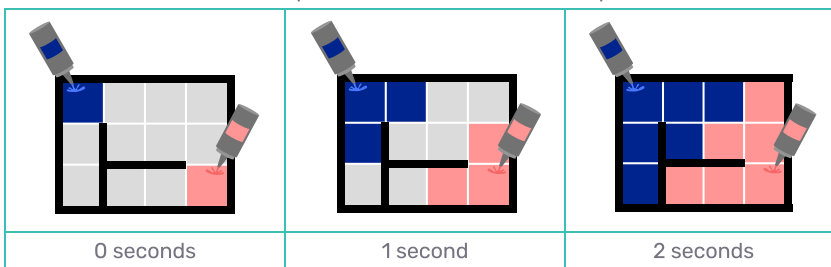
In contrast, a lossless compression algorithm only minimises the data. PNG images are stored using a lossless compression algorithm.

Watercolour

When painters pour watercolour into a maze, the colour will spread to neighbouring empty squares every second. The colour cannot spread through the walls, as you can see below.

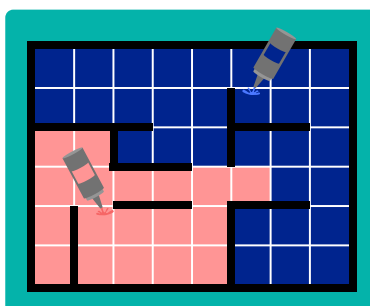
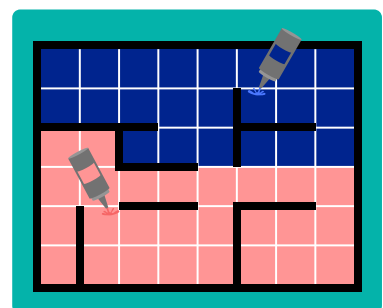
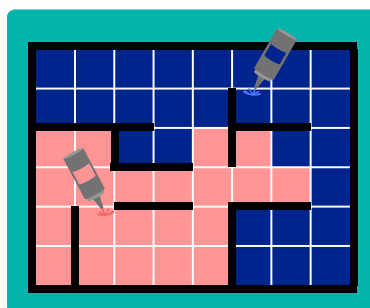
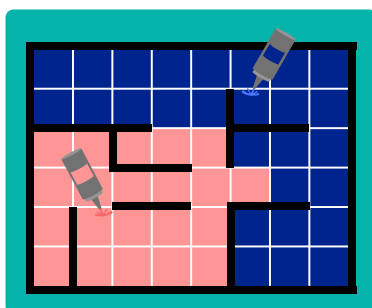
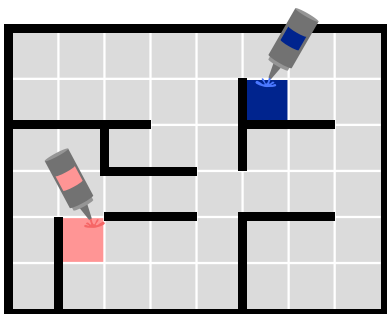


If the painters pour more than one watercolour into the maze, the first colour that reaches a square will fill it completely. When two colours reach a square at the same time, the square takes the darker colour (blue).



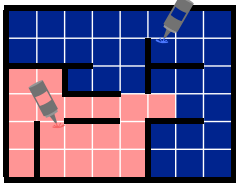
Question

The painters poured two colours into the maze below. What does the maze look like when all the squares are filled with colour?

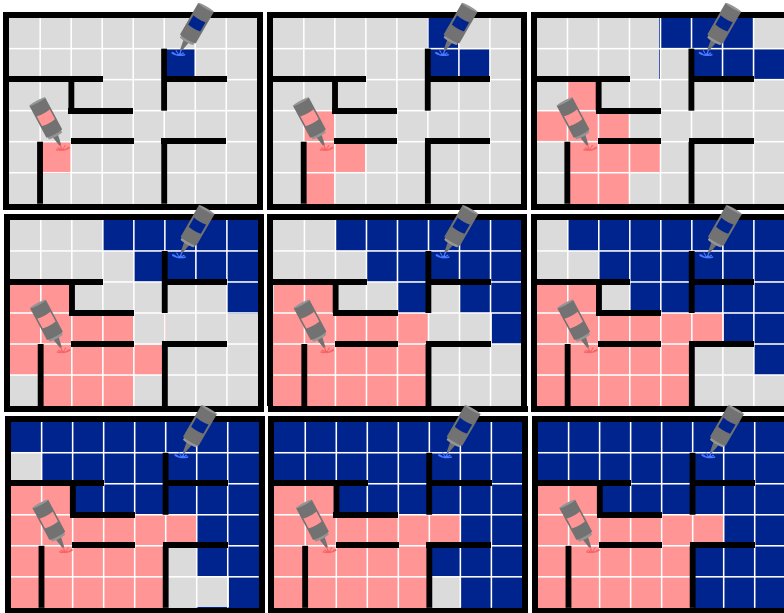


Explanation

The correct answer is



The image below shows the state of the maze second by second:



Background information

The setting of the task resembles a two dimensional array, which basically means a table with rows and columns. This way of representing the data is quite functional to simulate the state of the maze second after second and solve the task.

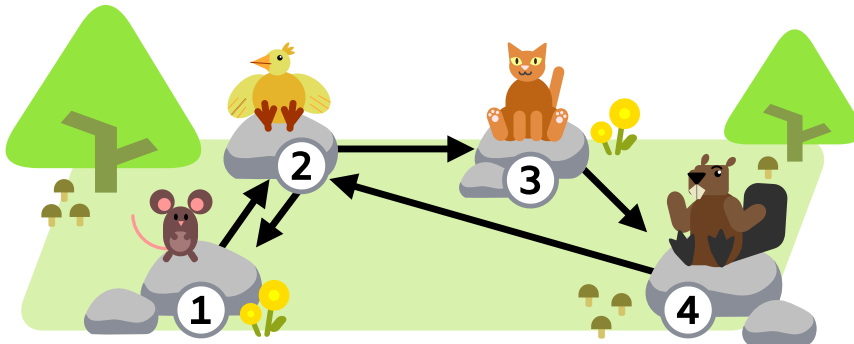
However, it is also possible to represent it as a graph in which each square is connected to its neighbors. This graph allows us to identify quickly which color will reach a square without simulating the whole scenario second by second. This approach using graphs is the same as performing a breadth-first search.

The breadth-first search (BFS) algorithm is one way of searching a tree or graph for a node that meets a set of criteria. It starts at any given node of a graph and visits all nodes at the current depth level before moving on to the nodes at the next depth level. Breadth-first search can be used to solve many problems in graph theory.

Algorithmic thinking is key in solving this tasks since students have to understand how the setting works in order to simulate its situation second by second.

Jumping Together

Beaver lives by rock number 4. In his neighbourhood there are three other rocks where his friends Bird, Cat and Mouse live. The animals have decided to have a party at Beaver's rock. In order to do this the animals can jump from rock to rock, following the arrows, according to the drawing below. Because they all want to be equally tired when the party starts, they all want to jump around and arrive on Beaver's rock having all made the exact same number of jumps (including Beaver).



Question

What is the minimum number of jumps that the animals must make so they all arrive on Beaver's rock with the same amount of jumps?

Fill in the number and click "Save" when you are done.

Answer:

Save

Explanation

By making a single jump, the cat can reach Beaver, but Beaver would be gone, and both the mouse and the bird have not yet arrived.

Please note the following observations:

- Beaver and *mouse* are both one jump away from rock 2. Their second jump could always be the same. So any number that works for Beaver will also work for mouse.
- The lowest numbers for *cat* are 1, 4 and 6.
- The lowest numbers for *bird* are 2, 4, 5 and 6

Looking at it from the point of view of Beaver:

- Beaver could circle around in three jumps, but *cat* cannot do three.
- Beaver could also do five jumps (4 → 2 → 1 → 2 → 3 → 4), but *cat* also cannot do five.

It turns out that every animal can do it in six jumps:

- Beaver can do six jumps (4 → 2 → 3 → 4 → 2 → 3 → 4)
- Cat can do six as well: (3 → 4 → 2 → 1 → 2 → 3 → 4)
- Mouse follows Beaver with six jumps (1 → 2 → 3 → 4 → 2 → 3 → 4)
- Bird can also make six jumps: (2 → 1 → 2 → 1 → 2 → 3 → 4)

Obviously the animals can also do it in more jumps, they could do their loop of 6 jumps and then add another 3, 5, 6, 8, 9, 10, 11, ... jumps. It's interesting to verify that 1, 2, 4 and 7 extra jumps are not possible. Can you find out why?

Background information

This task can be represented as a graph in computer science, where each rock corresponds to a **vertex**, represented by a numbered circle, and the arrows indicate **directed edges** in the graph. The objective is to find a **directed walk** from a starting vertex to the destination vertex.

In this task, it is allowed to visit the same edge multiple times in order to reach the destination vertex. The table below displays the shortest path from each vertex from the start vertex to the destination vertex, along with the length of the shortest path:

Starting vertex	The shortest path to vertex 4	Length of the shortest path
1	1→2→3→4	3
2	2→3→4	2
3	3→4	1
4	-	0

Furthermore, the table below shows the cycles, which represent the shortest path starting from each vertex back to the same vertex:

Starting vertex	Cycle starting from the node	Length of the cycle
1	1→2→1	2
2	2→3→4→2	3
3	3→4→2→3	3
4	4→2→3→4	3

By combining the shortest path from each vertex to the destination and the cycles of the intermediate vertices, all possible directed walks from a vertex to the destination can be enumerated. For example, the table below illustrates the directed walks from each vertex to vertex 4 with a length of 6:

Starting vertex	The directed walk of length 6 to vertex 4 by combining the above paths/cycles
1	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, $4 \rightarrow 2 \rightarrow 3 \rightarrow 4$
2	$2 \rightarrow 1 \rightarrow 2$, $2 \rightarrow 1 \rightarrow 2$, $2 \rightarrow 3 \rightarrow 4$
3	$3 \rightarrow 4$, $4 \rightarrow 2(2 \rightarrow 1 \rightarrow 2) \rightarrow 3 \rightarrow 4$
4	$4 \rightarrow 2 \rightarrow 3 \rightarrow 4$, $4 \rightarrow 2 \rightarrow 3 \rightarrow 4$

This approach is a kind of recursion, which is a concept or process depends on a simpler version of itself. To find the pattern of repeating, the length of the directed walks from starting vertex to the destination can be calculated without needing to illustrate all the solutions.

Correcting error

In one country, a phone number consists of 11 digits. However, people tend to make mistakes when writing them down, so there is a rule to correct ONE mistake. All phone numbers must satisfy the following:

- 8th digit = the last digit of the sum of the first 7 digits (1st-7th).
- 9th digit = the last digit of the sum of the first 4 digits (1st-4th).
- 10th digit = the last digit of the sum of the 1st, 2nd, 5th & 6th digits.
- 11th digit = the last digit of the sum of the 1st, 3rd, 5th & 7th digits.



For example, 12345678046 is a correct phone number, because

- the 8th digit is 8 ($1+2+3+4+5+6+7 = 28$)
- the 9th digit is 0 ($1+2+3+4 = 10$)
- the 10th digit is 4 ($1+2+5+6 = 14$)
- the 11th digit is 6 ($1+3+5+7 = 16$)

Question

Someone made ONE mistake when writing down the phone number 12312316710. What was the correct phone number?

Fill in the number and click "Save" when you are done.

Answer:

Save

Explanation

The correct answer: 12315316710.

The original phone number is 12312316710.

1. First, let's examine the last four digits (8th-11th), which are 6, 7, 1, 0:

- the 8th digit (6) is **incorrect** (because $1+2+3+1+2+3+1 = 13$)
- the 9th digit (7) is correct (because $1+2+3+1 = 7$)
- the 10th digit (1) is **incorrect** (because $1+2+2+3 = 8$)
- the 11th digit (0) is **incorrect** (because $1+3+2+1 = 7$)

As shown above, the last four digits (8th-11th) have 3 incorrect digits, so the one mistake cannot be there. Since the last four digits (8th-11th) are actually based on the first seven digits, the mistake must be in one of the first seven digits (1st-7th).

2. We then examine the first seven digits (1st-7th), which are 1, 2, 3, 1, 2, 3, 1:

- We already know that the 9th digit (which equals to the sum of 1st-4th digits) is correct, so the first four digits (1st-4th) must be correct.
- Let's examine the other three digits (5th-7th). According to the rules:
 - the 5th digit contributes to the 8th, 10th, 11th digits
 - the 6th digit contributes to the 8th and 10th digits
 - the 7th digit contributes to the 8th and 11th digits
- Since **8th, 10th and 11th digits are incorrect**, the mistake must be in the 5th digit.

3. Now let's find out what the 5th digit should be:

- We know that the 5th digit can only be a number between 0 to 9.
- There are 2 ways to find the correct number for the 5th digit:
 - Trial and error: If we try to use the numbers 0 to 9 one by one as the 5th digit, only the number 5 fits the rule and can make the 8th, 10th and 11th digits correct. Therefore, the 5th digit should be 5 (instead of 2).
 - Mathematical comparison: If we compare the 8th, 10th and 11th digits and their sums:

Position	Number	Original Sum	Corrected Sum (to match the Number)	Difference
8th digit	6	13	16	+3
10th digit	1	8	11	+3
11th digit	0	7	10	+3

As shown above, the difference between all pairs of original sum and corrected sum is +3. This means that the original 5th digit (2) should be increased by 3 to become 5 ($2+3=5$).

Lets check this corrected phone number: 1231**5**316710.

- the 8th digit (6) is **now correct** (because $1+2+3+1+5+3+1 = 16$)
- the 9th digit (7) is correct (because $1+2+3+1 = 7$)
- the 10th digit (1) is **now correct** (because $1+2+5+3 = 11$)
- the 11th digit (0) is **now correct** (because $1+3+5+1 = 10$)

Background information

Last 4 digits of the phone number in this task is called error correction code.

In this case, if only ONE mistake is made, we have enough information to identify and correct it, no matter which digit is wrong (including the last 4 digits).

Error correction codes are widely used in transmitting messages, when communication channels are noisy or unreliable, and we don't want to (or can't) repeat the message again.

Bebras Ball

Today is the annual Bebras Ball tournament.

Sixteen players compete in four rounds in order to determine their overall rank from 1st place to 16th place:

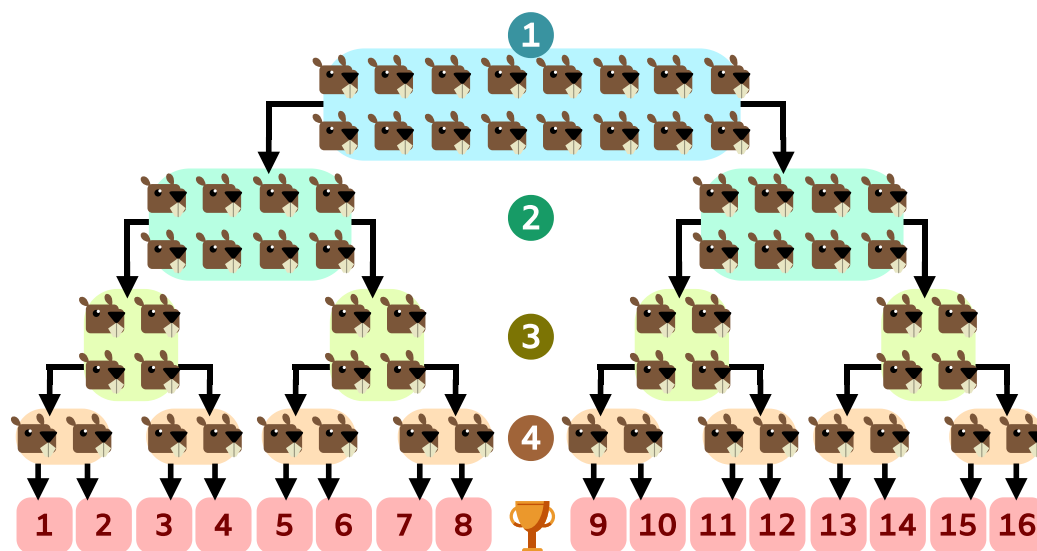
- All sixteen players compete together in round 1, but after each round, the players split up.
- The winning players follow the left arrow to their next round of competition (or final rank).
- The losing players follow the right arrow to their next round of competition (or final rank).

For example, a player who wins during rounds 1 and 2, but loses during rounds 3 and 4, will receive a rank of 4th place.

Question

Noro was a player in the Bebras Ball tournament. If Noro lost during exactly one round, which ranks might they have received?

Select the ranks and press "Save" when you are done.



Save

Erase

Explanation

Correct Answer: 9th, 5th, 3rd, 2nd

Since there are four rounds, and Noro lost during exactly one round, there are four possible scenarios.

Scenario 1: Noro lost during round 1 and won during all others. Thus, Noro would follow the arrows "right, left, left, left" which leads to the rank 9th place.

Scenario 2: Noro lost during round 2 and won during all others. Thus, Noro would follow the arrows "left, right, left, left" which leads to the rank 5th place.

Scenario 3: Noro lost during round 3 and won during all others. Thus, Noro would follow the arrows "left, left, right, left" which leads to the rank 3rd place.

Scenario 4: Noro lost during round 4 and won during all others. Thus, Noro would follow the arrows "left, left, left, right" which leads to the rank 2nd place.

Background information

The diagram in this task, which models the tournament, is a type of structure called a *decision tree*. The top of the tree (or root) is where the decision process begins. From there, we select a branch to follow depending on the answer to a decision question.

In this task, the decision question is “did I win or lose during the round?” Answers of “win” mean we follow the left branch and answers of “lose” mean we follow the right branch.

When we run out of branches we say that we have reached the leaves of the decision tree. The leaves represent the final outcomes. In this task, the final outcomes are the ranks.

If you have ever tried to identify someone’s secret number by making a guess and having them respond with “higher” or “lower”, you have also used a decision tree.

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”, and “fins” are used as decision questions. With repeated training, the program develops enough distinguishing features to correctly classify an image.

One possible way to solve this task includes counting how many rounds Noro would have lost in order to receive each of the 16 ranks, and then selecting a rank if he would have lost exactly once. This process is called evaluation and it means to execute an expression or algorithm and assess the results.

Another approach is to realize that the tournament involves four rounds, so if Noro lost during exactly one round then there are four different times this loss could have occurred: during round one, during round two, during round three, or during round four. Only these four resulting ranks need to be determined. This process is called abstraction and it means to focus in on only the relevant information and ignore the rest.

The decision tree used in this task can also be thought of as a binary tree. Every branch can be associated with a 0 (for “win”) or a 1 (for “lose”). A path from the root of the tree to a leaf would then be a combination of zeroes and ones which represent a binary number. All 4-digit binary numbers with exactly one 1 (meaning four rounds with exactly one “lose”) are 1000, 0100, 0010, and 0001. These binary numbers correspond with the values 8, 4, 2, and 1.

Recall that the answers to this task are 9, 5, 3, and 2. Can you see the connection between the binary numbers and task answers, and can you reason why they differ slightly? Think about what a rank of 1st place would look like in binary.

This process of reframing the task into a more familiar concept (binary numbers in this instance) is called generalisation.

Triangle

June took blocks of letters and stacked them in a triangle, as shown below.



She can find different paths from top to bottom through adjacent blocks in a triangle. Adjacent triangles are indicated by the arrows. She tries to count all the times she takes different paths when reading the same word.

Question

Exactly how many different ways can June read "DONKEY"?

Fill in the number and click "Save" when you are done.

Answer:

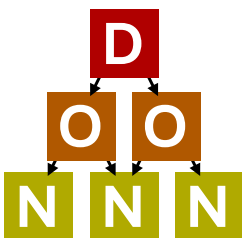
Save

Explanation

June can read "DONKEY" 32 different ways.

To calculate this, you have to take into account the number of paths from each block of letters (as you can see in the picture, there are always two possibilities).

For example, there are 2 different ways to read "DO" starting from the top block. Now, to find different paths to read "DON", there are always only 2 adjacent blocks that can be selected in the 3rd row from the last block of the path that reads "DO". Therefore, there are $2 \times 2 = 4$ different paths to read "DON" starting from the top block.



In addition, you have to take into account the number of possible paths you have to take to choose the next character (there are 5 characters: O, N, K, E and Y).

Therefore, the number of different ways June can read the word "DONKEY" will be 32 ($2 \times 2 \times 2 \times 2 \times 2 = 2^5 = 32$).

Background information

In computer science, recursion is a method of solving a computational problem where the solution depends on solutions to smaller instances of the same problem.

Counting the number of different paths in a triangular structure requires an effort to recognize some rules in a recursive way in simpler and smaller cases.

The task is about reasoning in a structure and finding a proper algorithm to find all possible paths in the triangle.

algorithm - developing a step-by-step solution to the problem, or the rules to follow to solve the problem (that is what has to be done in order to solve this problem)

The solver has to find a rule how number of possible ways increases during the reading process. Another words, solvers have to find out that each step of the algorithm always means exactly twice number of possibilities. (thanks for this point of view to Jiri Vanisek)

The background is a solid dark blue color. In the upper left corner, there are several overlapping organic, blob-like shapes in lighter shades of blue, ranging from a medium blue to a very light, almost white-blue. These shapes are positioned in the top left and top center areas, leaving the rest of the page as a solid dark blue.

2024 Bebras Challenge Round 2 | Intermediate Year 9 & 10

Gift selection

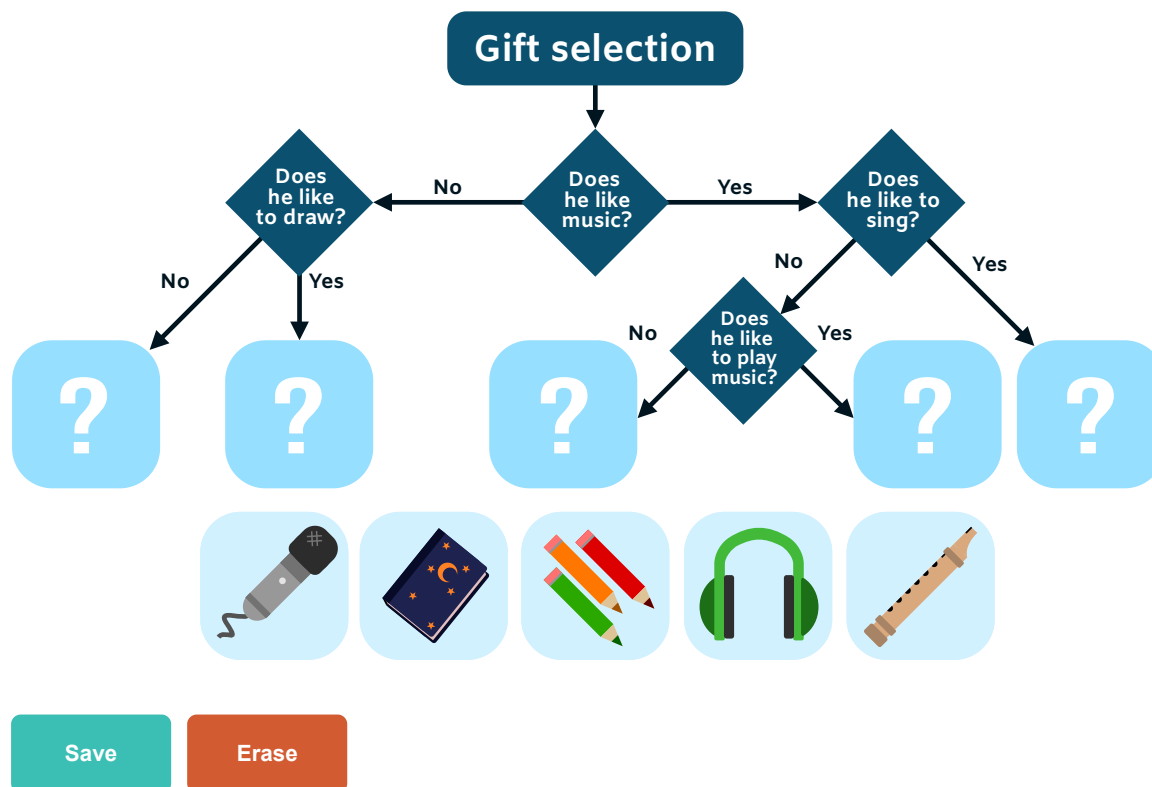
A new student has transferred to Madina's class. They want to surprise him with a gift.

In order to pick the right gift, they ask him a few questions. The flow chart below shows how they ask questions:

Task

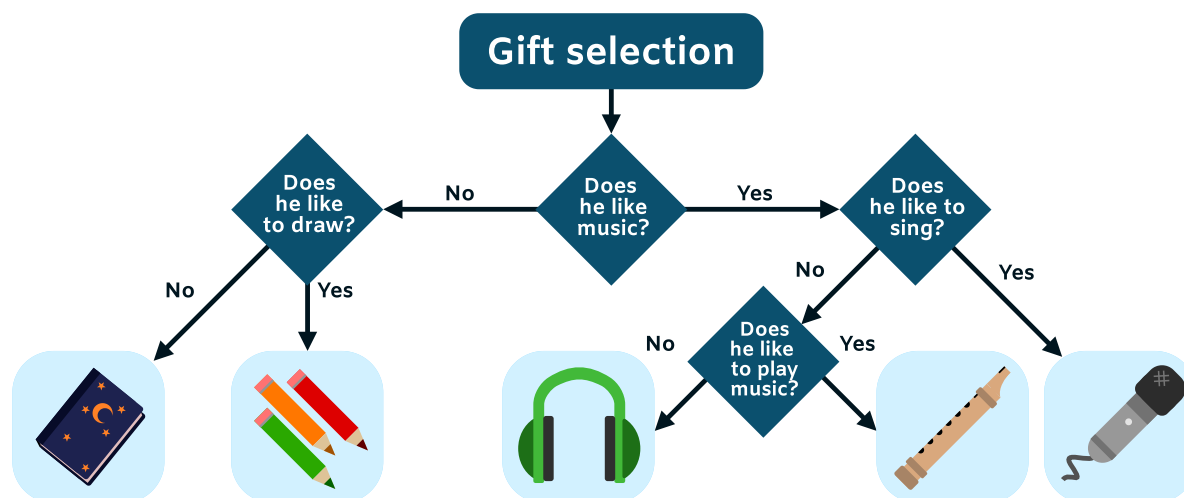
Help Madina's class decide which gifts would best fit the new student's possible answers.

Drag the gifts to the empty boxes and click "Save" when you are done.



Explanation

The following diagram illustrates the correct solution:



If he loves music, they can give him a microphone, recorder, and headphones. If he likes to sing, place the microphone in position E. If he doesn't like to sing but likes to play music, place the recorder in position D and the headphones in position C.

If he doesn't like music but likes to draw, place the colour pencils in position B. Otherwise, place the diary in position A.

Background information

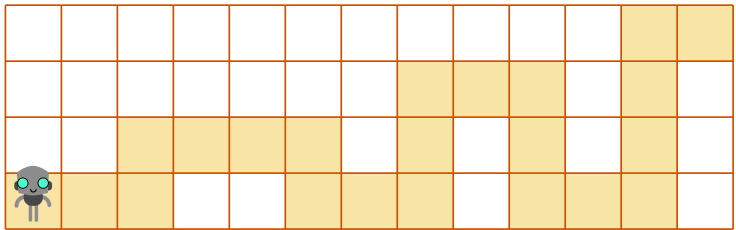
A decision tree is like a map to help you make decisions (in our example we used the decision tree to place gifts based on new student's answers).

It starts with a key question, and each answer can lead to either more questions, or a final answer.

Computer programs can often be visualised as decision trees. A program can be given an input, and then we use code to specify questions to ask of the input, which ultimately leads with a result that can be output.

Robot on the path

We want to program a robot to walk along a path in a park as well as possible. The path is shown in yellow in the picture below. The robot starts at the **left end** of the path, indicated by the little robot image.



The robot knows these **commands**:

- **U** move one square up
- **R** move one square right
- **D** move one square down

The robot's memory holds 5 commands. The robot repeats this sequence until it reaches the right border of the park.

Question

Which sequence of commands will take the robot to the right border of the park while moving over **the fewest number** of white squares?

RRURR

RRRUR

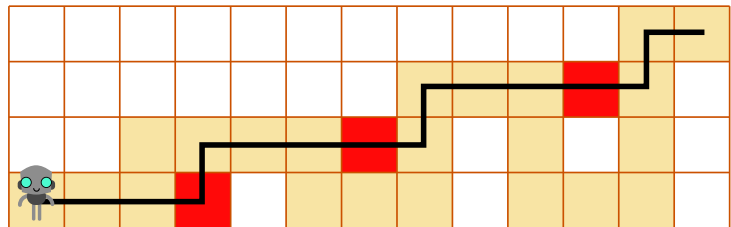
URRRD

RURRR

Explanation

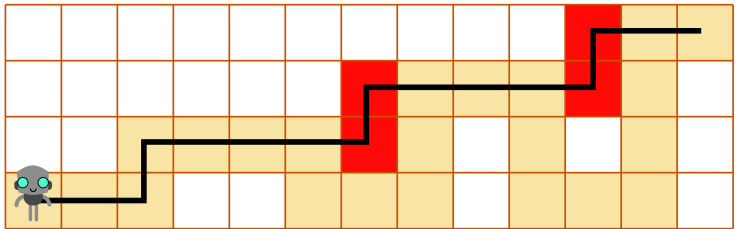
The correct answer is B) **RRRUR**

Here is what happens repeating commands in B) **RRRUR**:

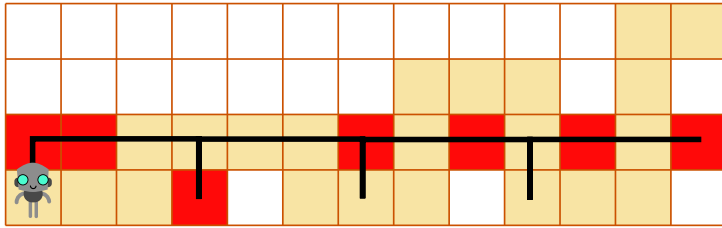


We see that the robot is off the path at **3 squares** (coloured red).

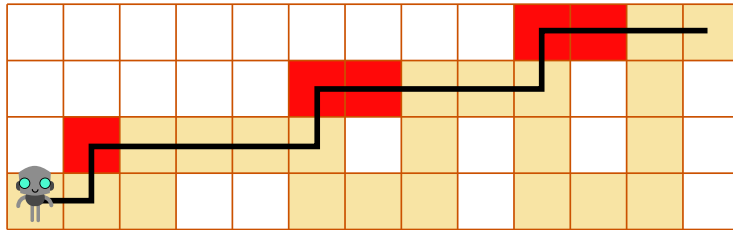
If it were to execute commands A) **RRURR**, it would be off the path at **4 squares**:



If it were to execute commands C) **URRRD**, it would go off the path at **7 squares**:



If it were executing commands D) **RURRR**, it would go off the path at **5 squares**:



Background information

The path of the robot is described by a sequence of 5 commands that the robot repeats. According to the commands, the robot changes its position. Executing a (repeating) sequence of commands is one of the fundamental ideas in programming.

Commands control the robot in this problem by moving it in one of four directions, so-called "absolute control", without the robot turning in that direction.

BebrasGPT

BebrasGPT is a chat bot that has been developed to produce three-word sentences. It does this by predicting the next word based on the previous sequence of words. Each word is chosen one by one, with the next word being chosen based on the probability of it following the current sequence. The tables below show some of these probabilities.

Probabilities for second word:

	"love"	"hate"
"Cats"	0.7	0.3
"Dogs"	0.6	0.4

Probabilities for third word:

	"swimming"	"running"
"Cats love"	0.3	0.8
"Cats hate"	0.9	0.1
"Dogs love"	0.7	0.3
"Dogs hate"	0.1	0.9

For example, if the sentence starts with the word "Cats", the probability of the 3-word sentence being "Cats love running" is 0.56 because:

- the probability of the second word being "love" if the previous word is "Dogs" is 0.7,
- and the probability of the next word being "running" if the previous sequence is "Dogs love" is 0.8,
- so, because the model predicts words one by one, the probability is $0.7 * 0.8 = 0.56$.

Question

If a sentence starts with the word "Dogs" what is the most likely output of BebrasGPT?

"Dogs hate swimming"

"Dogs hate running"

"Dogs love swimming"

"Dogs love running"

Explanation

The correct answer is: "Dogs love swimming."

The probabilities of each sentence are the following:

"Dogs hate swimming", $0.4 * 0.1 = 0.04$

"Dogs hate running", $0.4 * 0.9 = 0.36$

"Dogs love swimming", $0.6 * 0.7 = \mathbf{0.42}$

"Dogs love running", $0.6 * 0.3 = 0.18$

C has the highest probability among the choices.

Background information

This exercise involves an example of an artificial model for language modeling and the production of text. This is a probabilistic model, meaning that each output of the model is based on probabilities. There are many types of models. In this case, the model is producing one word at a time, based on the entire sequence of previous words, which is similar to models like ChatGPT. Of course these models are much, much larger than the model shown in the exercise.

One related important aspect in artificial intelligence and machine learning is the difference between two type of algorithms:

- the learning algorithm,
- and the inference algorithm.

The learning algorithm is used to "train" the model. This corresponds, in our example, to finding which values of the probabilities should be in the table above. In our case, this has already been done. The model is already trained. Usually, this is the most difficult part in terms of theory. For example, ChatGPT was training for many months on several different GPUs, which cost millions of dollars.

The inference algorithm is the one we use after training, to produce the output. For example, when you type the prompt "Hi, ChatGPT. What is machine learning?", the model will take that sequence of words and use something equivalent to a very large table of probabilities to output the answer to the prompt. Although the cost of running the inference algorithm once is much smaller than the training, if the model is used by millions of people everyday (like ChatGPT), the cost of inference is higher than that of training (which was a one time cost).



This task requires algorithmic thinking as the students figure out the next word one at a time, and finding the 3-word sentence with the largest probability. Students are probably familiar with the text predict functions of their smartphones but this task allows them to think through how it works.

Stamp Machines

Paula has the following four machines, that she uses to make designs on paper:



These are the functions of the machines, in no particular order:

- Colour the whole paper black
- Turn the paper 180 degrees
- Stamp  on the paper
- Stamp  on the paper

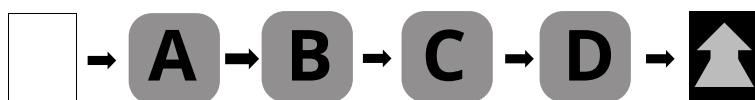
These machines can be used together to make special designs. At each step, a machine can completely change what was done before.

For example, the machine that colours the paper black can cover stamps made on the paper.

Paula initially puts a piece of white paper in the machines, uses the machines in the order shown below, and gets the design



at the end.



Question

Which machine stamps  on the paper?



Explanation

The answer is (D).

There are some conditions to obtain the final paper shown:

- The machine D can't be the one that colours the paper black, or else the final paper would be entirely black.
- Machine A can't be the one that turns the paper 180 degrees, or else the final paper would have an arrow upside down.
- The machine that colours the paper black is before any other that stamps the paper, or else this stamp wouldn't appear.
- The machine that stamps an arrow on the paper is before the machine that turns the paper 180 degrees, and the machine that turns the paper is before the machine that stamps a triangle on the paper.

By all these conditions, we conclude that machines that stamp an arrow on the paper, turn the paper, and stamp a triangle on the paper are in sequence. Then, the machine that colours the paper black is A, and the others B, C, D respectively.

Background information

In this task, each of the machines A, B, C, and D has a specific function and, when associated in different orders, might present different results. Furthermore, the overall result, such as presented in this Task (a black paper with two black figures), depends on a strategic association of the machines. Therefore, students should associate the machines in a sequence that allows them to obtain the expected composition on the paper. They must also consider that each step depends on the previous one – it means the task to be performed by one machine only makes sense if that machine is put after another specific machine. There is a dependency relationship here.

Arranging machines is like programming. For a program to solve a task perfectly, all the statements in the program must be correctly ordered. So, computer scientists must tell the computer what it must do step by step.

Algorithmic Thinking is essential to solve this task. Each machine has one specific function that can influence the other machine's function: for example, if the machine that colors the paper black is put after the machine that stamps a triangle on the paper, the second machine will make the triangle disappear (so this order does not work for this task). The main goal of this task is to discover what position each machine should be put in the sequence A, B, C, D, in order to get the final paper shown and this demands comprehension of how the machines work and how they are used. To some extent, the task is also related to reverse engineering, since the student needs to understand how a piece of the algorithm works based on the input given and output obtained.

Decomposition is also present in this Task. Instead of looking to the global process all at once, students can analyze each step to decide which machine should be placed after another in the association.

Sealed Letters

The Republic of Beaveria has a cabinet full of secret letters.

Among the 16 letters in this cabinet, numbered 1 to 16, 10 had been opened, while the other 6 were still inside their sealed envelopes.

One evening, an enemy spy snuck in and opened one of the sealed letters. However, they forgot to seal it again.

The next morning, the Republic of Beaveria goes into an investigation after noticing that there are now 11 opened letters, as shown in the diagram.

















The Republic's guard does not recall all the details, but they are sure that before the enemy spy sneaked in:

- The total number of opened letters in the C2 and C4 columns was even.
- The total number of opened letters in the C3 and C4 columns was even.
- The total number of opened letters in the R2 and R4 rows was even.
- The total number of opened letters in the R3 and R4 rows was even

Question

Which letter was opened by the enemy spy?

Click on the letter and press "Save".

	C1	C2	C3	C4
R1	1 	2 	3 	4 
R2	5 	6 	7 	8 
R3	9 	10 	11 	12 
R4	13 	14 	15 	16 

Save

Erase

Explanation

The correct answer is 13.

We follow this line of reasoning:

- There is an even number of opened letters in the C2 and C4 columns combined, matching the guard's recollection. As there is only one letter opened by the spy, this implies that the letter opened by the spy must be either in the C1 or C3 column. There is an even number of opened letters in the C3 and C4 columns combined, matching the guard's recollection. Given the previous statement, this implies that the letter opened by the spy must be in the C1 column.
- There is an odd number of opened letters in the R2 and R4 rows combined, which does not match the guard's recollection. This implies that the letter opened by the spy must be either in the R2 or R4 row.
- There is an odd number of opened letters in the R3 and R4 rows combined, which does not match the guard's recollection. Given the previous statement, this implies that the letter opened by the spy must be in the R4 row.

Therefore, the letter read by the spy is in the C1 column and R4 row, which points to 13.

Background information

This challenge introduces the concept of *error-correcting* codes. Digital data is essentially a sequence of bits: 1s and 0s. When it passes through a network, it can be corrupted due to noise or due to the action of malicious entities. Data stored on DVDs may also become corrupted if these storage devices are scratched. It is thus important to have a scheme that is capable of not only detecting that corruption has occurred (*error detection*) but also correcting the corrupted bits (*error correction*).

The first modern error-correcting code was introduced by Richard Hamming in 1950. This challenge specifically borrows the idea of the [15, 11] *Hamming code*, with the closed and opened letters corresponding to 0s and 1s, respectively. This coding scheme takes an 11-bit data and inserts 4 *parity bits*. These parity bits occupy slots 2, 3, 5, and 9, while the bits of the original data occupy the remaining slots (except slot 1). The parity bits are set so that there is an even number of 1s in the 2nd and 4th columns, 3rd and 4th columns, 2nd and 4th rows, and 3rd and 4th rows. Finally, the bit at slot 1 is set so that there is an even number of 1s in total.

As seen in the presented solution, this scheme allows for the corrupted bit to be identified and subsequently corrected, provided that there is at most 1 corrupted bit. Although more sophisticated error-correcting schemes are necessary to correct 2 or more corrupted bits, the Hamming code's efficiency contributes to its continued usage in NAND flash memory chips embedded in mobile phones and solid-state drives.

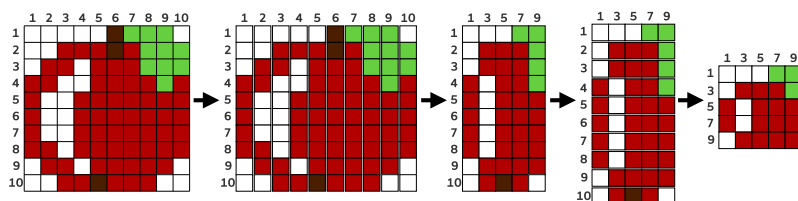
An important skill in computational thinking is *deductive reasoning* – logically deriving conclusions from a set of known facts or *premises*. A common method in deductive reasoning is the process of *elimination*. For instance, in this challenge, knowing that the 2nd and 4th columns combined have an even number of opened letters eliminates the possibility that they have the letter read by the spy, thus narrowing down the choices to the 1st and 3rd columns. This ability to reason in a systematic, top-down manner is a core competency in designing algorithms, tracing code, and debugging programs without resorting to guesswork or committing logical fallacies.

Snail Compress

Snails have a special technique to shrink their captured images.

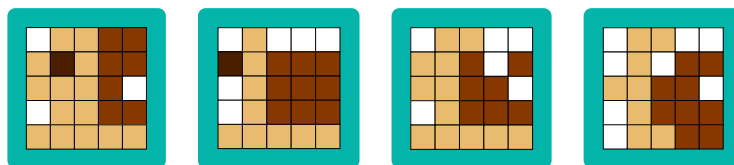
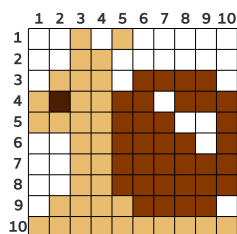
First, they will cut the original image into 10 equally sized strips vertically. Then, they will assemble the odd-numbered vertical strips to create a new image.

Next, they will cut the new image horizontally into 10 equally sized strips. Then, they will assemble the odd-numbered horizontal strips to create a complete shrunken image.



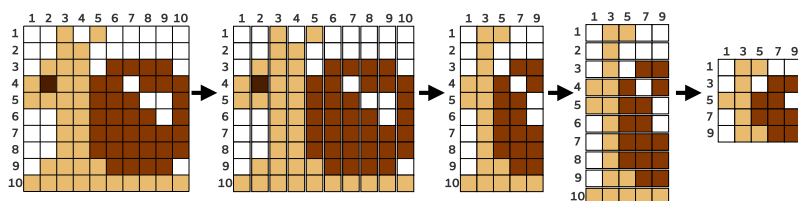
Question

What image will the snails obtain after shrinking the given image using this technique?



Explanation

Following the same procedure as with the picture of the apple, the snail image is compressed in the following way:



First all the even-numbered columns will be erased, then all the even-numbered rows will be erased, which means that only the cells from odd-numbered columns, that are also in odd-numbered rows will remain.

We can see that the correct answer is D, as it is the only image that uses the 9th row for compression of the picture.

Background information

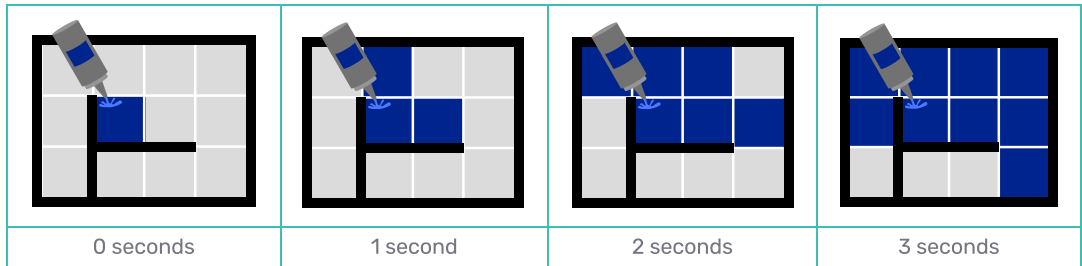
Compression of an image refers to the process of reducing the size of an image file by eliminating or minimising some of the unnecessary data in the image while still maintaining sufficient image quality for use. This process helps to reduce the file size of images, making them easier to store or transmit.

This is lossy compression algorithm, as it completely eliminates some of the rows and columns in the original image. JPG images are also stored using a lossy compression algorithm.

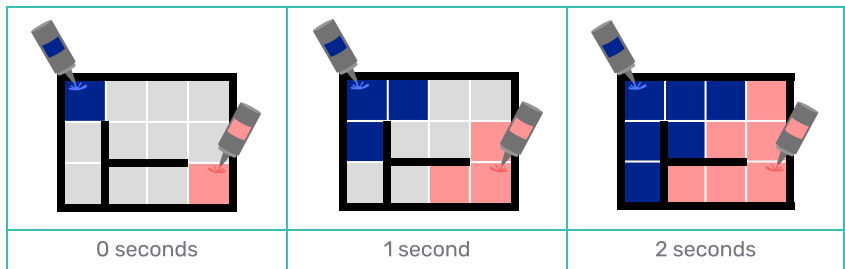
In contrast, a lossless compression algorithm only minimises the data. PNG images are stored using a lossless compression algorithm.

Watercolour

When painters pour watercolour into a maze, the colour will spread to neighbouring empty squares every second. The colour cannot spread through the walls, as you can see below.

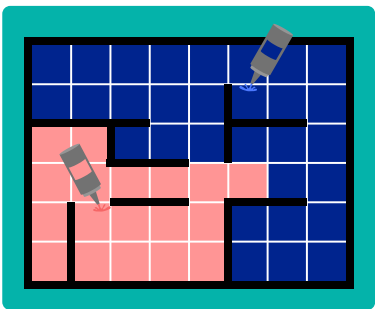
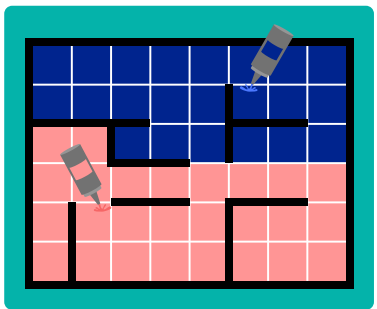
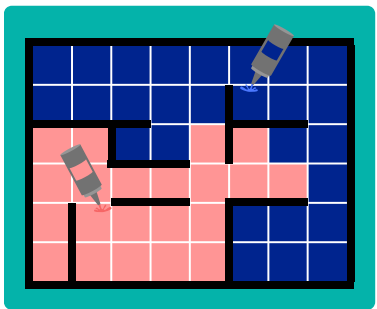
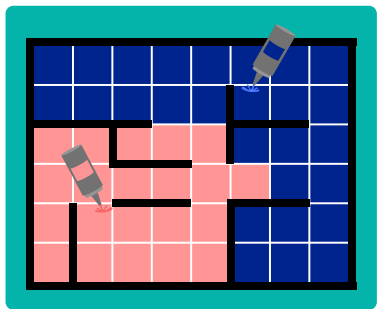
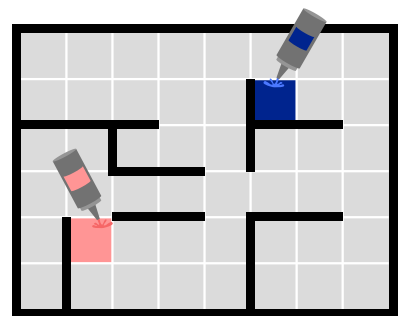


If the painters pour more than one watercolour into the maze, the first colour that reaches a square will fill it completely. When two colours reach a square at the same time, the square takes the darker colour (blue).



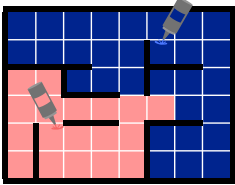
Question

The painters poured two colours into the maze below. What does the maze look like when all the squares are filled with colour?

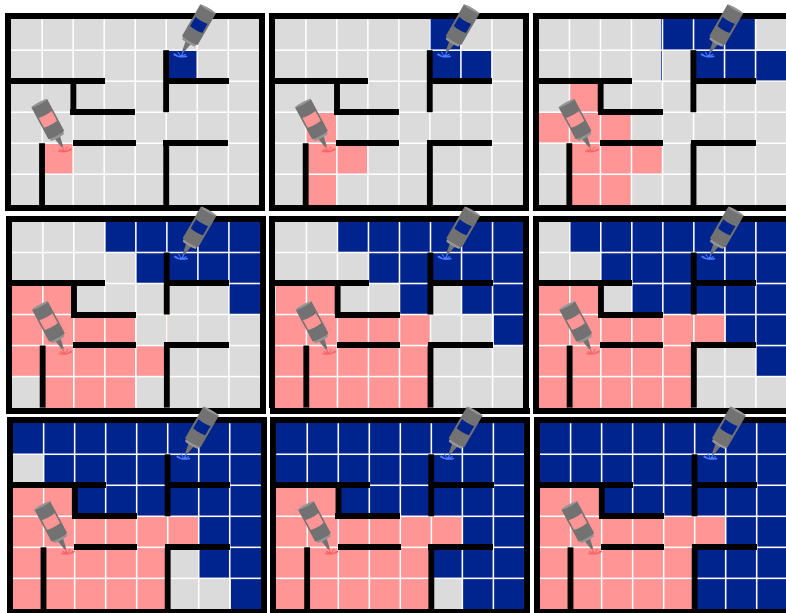


Explanation

The correct answer is



The image below shows the state of the maze second by second:



Background information

The setting of the task resembles a two dimensional array, which basically means a table with rows and columns. This way of representing the data is quite functional to simulate the state of the maze second after second and solve the task.

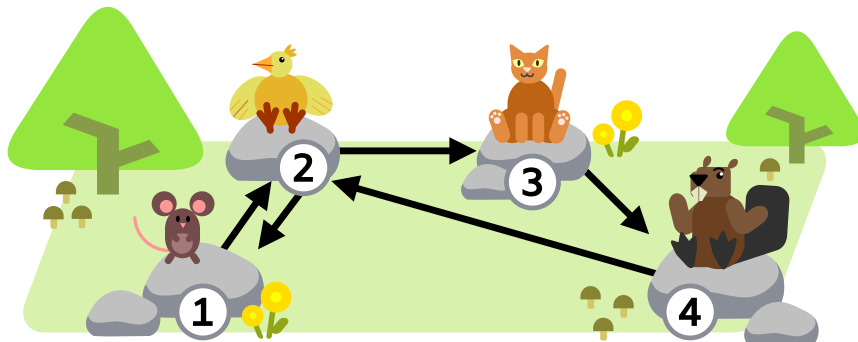
However, it is also possible to represent it as a graph in which each square is connected to its neighbors. This graph allows us to identify quickly which color will reach a square without simulating the whole scenario second by second. This approach using graphs is the same as performing a breadth-first search.

The breadth-first search (BFS) algorithm is one way of searching a tree or graph for a node that meets a set of criteria. It starts at any given node of a graph and visits all nodes at the current depth level before moving on to the nodes at the next depth level. Breadth-first search can be used to solve many problems in graph theory.

Algorithmic thinking is key in solving this tasks since students have to understand how the setting works in order to simulate its situation second by second.

Jumping Together

Beaver lives by rock number 4. In his neighbourhood there are three other rocks where his friends Bird, Cat and Mouse live. The animals have decided to have a party at Beaver's rock. In order to do this the animals can jump from rock to rock, following the arrows, according to the drawing below. Because they all want to be equally tired when the party starts, they all want to jump around and arrive on Beaver's rock having all made the exact same number of jumps (including Beaver).



Question

What is the minimum number of jumps that the animals must make so they all arrive on Beaver's rock with the same amount of jumps?

Fill in the number and click "Save" when you are done.

Answer:

Save

Explanation

By making a single jump, the cat can reach Beaver, but Beaver would be gone, and both the mouse and the bird have not yet arrived.

Please note the following observations:

- Beaver and *mouse* are both one jump away from rock 2. Their second jump could always be the same. So any number that works for Beaver will also work for mouse.
- The lowest numbers for *cat* are 1, 4 and 6.
- The lowest numbers for *bird* are 2, 4, 5 and 6

Looking at it from the point of view of Beaver:

- Beaver could circle around in three jumps, but *cat* cannot do three.
- Beaver could also do five jumps (4 → 2 → 1 → 2 → 3 → 4), but *cat* also cannot do five.

It turns out that every animal can do it in six jumps:

- Beaver can do six jumps (4 → 2 → 3 → 4 → 2 → 3 → 4)
- Cat can do six as well: (3 → 4 → 2 → 1 → 2 → 3 → 4)
- Mouse follows Beaver with six jumps (1 → 2 → 3 → 4 → 2 → 3 → 4)
- Bird can also make six jumps: (2 → 1 → 2 → 1 → 2 → 3 → 4)

Obviously the animals can also do it in more jumps, they could do their loop of 6 jumps and then add another 3, 5, 6, 8, 9, 10, 11, ... jumps. It's interesting to verify that 1, 2, 4 and 7 extra jumps are not possible. Can you find out why?

Background information

This task can be represented as a graph in computer science, where each rock corresponds to a **vertex**, represented by a numbered circle, and the arrows indicate **directed edges** in the graph. The objective is to find a **directed walk** from a starting vertex to the destination vertex.

In this task, it is allowed to visit the same edge multiple times in order to reach the destination vertex. The table below displays the shortest path from each vertex from the start vertex to the destination vertex, along with the length of the shortest path:

Starting vertex	The shortest path to vertex 4	Length of the shortest path
1	1→2→3→4	3
2	2→3→4	2
3	3→4	1
4	-	0

Furthermore, the table below shows the cycles, which represent the shortest path starting from each vertex back to the same vertex:

Starting vertex	Cycle starting from the node	Length of the cycle
1	1→2→1	2
2	2→3→4→2	3
3	3→4→2→3	3
4	4→2→3→4	3

By combining the shortest path from each vertex to the destination and the cycles of the intermediate vertices, all possible directed walks from a vertex to the destination can be enumerated. For example, the table below illustrates the directed walks from each vertex to vertex 4 with a length of 6:

Starting vertex	The directed walk of length 6 to vertex 4 by combining the above paths/cycles
1	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, $4 \rightarrow 2 \rightarrow 3 \rightarrow 4$
2	$2 \rightarrow 1 \rightarrow 2$, $2 \rightarrow 1 \rightarrow 2$, $2 \rightarrow 3 \rightarrow 4$
3	$3 \rightarrow 4$, $4 \rightarrow 2(2 \rightarrow 1 \rightarrow 2) \rightarrow 3 \rightarrow 4$
4	$4 \rightarrow 2 \rightarrow 3 \rightarrow 4$, $4 \rightarrow 2 \rightarrow 3 \rightarrow 4$

This approach is a kind of recursion, which is a concept or process depends on a simpler version of itself. To find the pattern of repeating, the length of the directed walks from starting vertex to the destination can be calculated without needing to illustrate all the solutions.

Correcting error

In one country, a phone number consists of 11 digits. However, people tend to make mistakes when writing them down, so there is a rule to correct ONE mistake. All phone numbers must satisfy the following:

- 8th digit = the last digit of the sum of the first 7 digits (1st-7th).
- 9th digit = the last digit of the sum of the first 4 digits (1st-4th).
- 10th digit = the last digit of the sum of the 1st, 2nd, 5th & 6th digits.
- 11th digit = the last digit of the sum of the 1st, 3rd, 5th & 7th digits.



For example, 12345678046 is a correct phone number, because

- the 8th digit is 8 ($1+2+3+4+5+6+7 = 28$)
- the 9th digit is 0 ($1+2+3+4 = 10$)
- the 10th digit is 4 ($1+2+5+6 = 14$)
- the 11th digit is 6 ($1+3+5+7 = 16$)

Question

Someone made ONE mistake when writing down the phone number 12312316710. What was the correct phone number?

Fill in the number and click "Save" when you are done.

Answer:

Save

Explanation

The correct answer: 12315316710.

The original phone number is 12312316710.

1. First, let's examine the last four digits (8th-11th), which are 6, 7, 1, 0:

- the 8th digit (6) is **incorrect** (because $1+2+3+1+2+3+1 = 13$)
- the 9th digit (7) is correct (because $1+2+3+1 = 7$)
- the 10th digit (1) is **incorrect** (because $1+2+2+3 = 8$)
- the 11th digit (0) is **incorrect** (because $1+3+2+1 = 7$)

As shown above, the last four digits (8th-11th) have 3 incorrect digits, so the one mistake cannot be there. Since the last four digits (8th-11th) are actually based on the first seven digits, the mistake must be in one of the first seven digits (1st-7th).

2. We then examine the first seven digits (1st-7th), which are 1, 2, 3, 1, 2, 3, 1:

- We already know that the 9th digit (which equals to the sum of 1st-4th digits) is correct, so the first four digits (1st-4th) must be correct.
- Let's examine the other three digits (5th-7th). According to the rules:
 - the 5th digit contributes to the 8th, 10th, 11th digits
 - the 6th digit contributes to the 8th and 10th digits
 - the 7th digit contributes to the 8th and 11th digits
- Since **8th, 10th and 11th digits are incorrect**, the mistake must be in the 5th digit.

3. Now let's find out what the 5th digit should be:

- We know that the 5th digit can only be a number between 0 to 9.
- There are 2 ways to find the correct number for the 5th digit:
 - Trial and error: If we try to use the numbers 0 to 9 one by one as the 5th digit, only the number 5 fits the rule and can make the 8th, 10th and 11th digits correct. Therefore, the 5th digit should be 5 (instead of 2).
 - Mathematical comparison: If we compare the 8th, 10th and 11th digits and their sums:

Position	Number	Original Sum	Corrected Sum (to match the Number)	Difference
8th digit	6	13	16	+3
10th digit	1	8	11	+3
11th digit	0	7	10	+3

As shown above, the difference between all pairs of original sum and corrected sum is +3. This means that the original 5th digit (2) should be increased by 3 to become 5 ($2+3=5$).

Lets check this corrected phone number: 1231**5**316710.

- the 8th digit (6) is **now correct** (because $1+2+3+1+5+3+1 = 16$)
- the 9th digit (7) is correct (because $1+2+3+1 = 7$)
- the 10th digit (1) is **now correct** (because $1+2+5+3 = 11$)
- the 11th digit (0) is **now correct** (because $1+3+5+1 = 10$)

Background information

Last 4 digits of the phone number in this task is called error correction code.

In this case, if only ONE mistake is made, we have enough information to identify and correct it, no matter which digit is wrong (including the last 4 digits).

Error correction codes are widely used in transmitting messages, when communication channels are noisy or unreliable, and we don't want to (or can't) repeat the message again.

Bebras Ball

Today is the annual Bebras Ball tournament.

Sixteen players compete in four rounds in order to determine their overall rank from 1st place to 16th place:

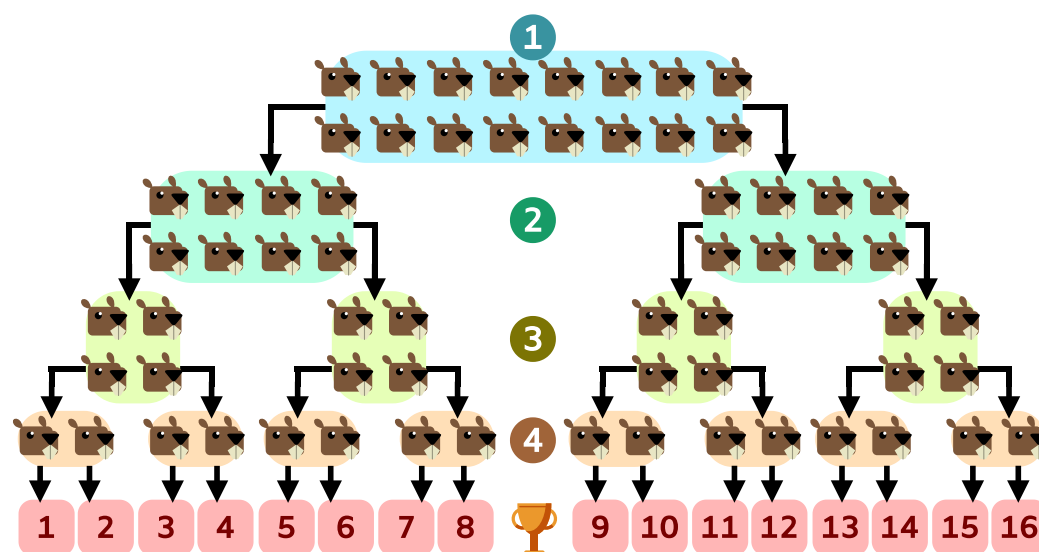
- All sixteen players compete together in round 1, but after each round, the players split up.
- The winning players follow the left arrow to their next round of competition (or final rank).
- The losing players follow the right arrow to their next round of competition (or final rank).

For example, a player who wins during rounds 1 and 2, but loses during rounds 3 and 4, will receive a rank of 4th place.

Question

Noro was a player in the Bebras Ball tournament. If Noro lost during exactly one round, which ranks might they have received?

Select the ranks and press "Save" when you are done.



Save

Erase

Explanation

Correct Answer: 9th, 5th, 3rd, 2nd

Since there are four rounds, and Noro lost during exactly one round, there are four possible scenarios.

Scenario 1: Noro lost during round 1 and won during all others. Thus, Noro would follow the arrows "right, left, left, left" which leads to the rank 9th place.

Scenario 2: Noro lost during round 2 and won during all others. Thus, Noro would follow the arrows "left, right, left, left" which leads to the rank 5th place.

Scenario 3: Noro lost during round 3 and won during all others. Thus, Noro would follow the arrows "left, left, right, left" which leads to the rank 3rd place.

Scenario 4: Noro lost during round 4 and won during all others. Thus, Noro would follow the arrows "left, left, left, right" which leads to the rank 2nd place.

Background information

The diagram in this task, which models the tournament, is a type of structure called a *decision tree*. The top of the tree (or root) is where the decision process begins. From there, we select a branch to follow depending on the answer to a decision question.

In this task, the decision question is “did I win or lose during the round?” Answers of “win” mean we follow the left branch and answers of “lose” mean we follow the right branch.

When we run out of branches we say that we have reached the leaves of the decision tree. The leaves represent the final outcomes. In this task, the final outcomes are the ranks.

If you have ever tried to identify someone’s secret number by making a guess and having them respond with “higher” or “lower”, you have also used a decision tree.

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”, and “fins” are used as decision questions. With repeated training, the program develops enough distinguishing features to correctly classify an image.

One possible way to solve this task includes counting how many rounds Noro would have lost in order to receive each of the 16 ranks, and then selecting a rank if he would have lost exactly once. This process is called evaluation and it means to execute an expression or algorithm and assess the results.

Another approach is to realize that the tournament involves four rounds, so if Noro lost during exactly one round then there are four different times this loss could have occurred: during round one, during round two, during round three, or during round four. Only these four resulting ranks need to be determined. This process is called abstraction and it means to focus in on only the relevant information and ignore the rest.

The decision tree used in this task can also be thought of as a binary tree. Every branch can be associated with a 0 (for “win”) or a 1 (for “lose”). A path from the root of the tree to a leaf would then be a combination of zeroes and ones which represent a binary number. All 4-digit binary numbers with exactly one 1 (meaning four rounds with exactly one “lose”) are 1000, 0100, 0010, and 0001. These binary numbers correspond with the values 8, 4, 2, and 1.

Recall that the answers to this task are 9, 5, 3, and 2. Can you see the connection between the binary numbers and task answers, and can you reason why they differ slightly? Think about what a rank of 1st place would look like in binary.

This process of reframing the task into a more familiar concept (binary numbers in this instance) is called generalisation.

Triangle

June took blocks of letters and stacked them in a triangle, as shown below.



She can find different paths from top to bottom through adjacent blocks in a triangle. Adjacent triangles are indicated by the arrows. She tries to count all the times she takes different paths when reading the same word.

Question

Exactly how many different ways can June read "DONKEY"?

Fill in the number and click "Save" when you are done.

Answer:

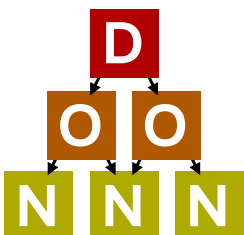
Save

Explanation

June can read "DONKEY" 32 different ways.

To calculate this, you have to take into account the number of paths from each block of letters (as you can see in the picture, there are always two possibilities).

For example, there are 2 different ways to read "DO" starting from the top block. Now, to find different paths to read "DON", there are always only 2 adjacent blocks that can be selected in the 3rd row from the last block of the path that reads "DO". Therefore, there are $2 \times 2 = 4$ different paths to read "DON" starting from the top block.



In addition, you have to take into account the number of possible paths you have to take to choose the next character (there are 5 characters: O, N, K, E and Y).

Therefore, the number of different ways June can read the word "DONKEY" will be 32 ($2 \times 2 \times 2 \times 2 \times 2 = 2^5 = 32$).

Background information

In computer science, recursion is a method of solving a computational problem where the solution depends on solutions to smaller instances of the same problem.

Counting the number of different paths in a triangular structure requires an effort to recognize some rules in a recursive way in simpler and smaller cases.

The task is about reasoning in a structure and finding a proper algorithm to find all possible paths in the triangle.

algorithm - developing a step-by-step solution to the problem, or the rules to follow to solve the problem (that is what has to be done in order to solve this problem)





The solver has to find a rule how number of possible ways increases during the reading process. Another words, solvers have to find out that each step of the algorithm always means exactly twice number of possibilities. (thanks for this point of view to Jiri Vanisek)

Open It

A spy has to unlock a safe by lighting up the correct number combination.

On each move, the spy can rotate the arrow clockwise for either 3 or 4 steps. The arrow toggles the light of the number it lands on. If the number's light was off, the light is turned on. If the number's light was on, the light is turned off.

For example, this is what happens if the spy makes 3 moves, each rotating by 4 steps:

Starting position	After 1st move (4 steps)	After 2nd move (4 steps)	After 3rd move (4 steps)
			

Question

To open the safe, the spy needs to light up only 7 and 8 (no other numbers should be lit).

What is the minimum number of moves the spy needs to do in order to light up only 7 and 8 from the starting position shown above?

3 moves

4 moves

5 moves

6 moves

7 moves





Explanation

The correct answer is **4 moves**.

One way to start solving this task is by realizing that the numbers 8 and 7 can both be reached by making two moves from the starting position. To reach 7, we move it twice by 3 steps. To reach 8, we move it once by 4 steps, then by 3.

Furthermore, if we are on 8, we can reach 7 in two moves, but not the other way round: reaching 8 from 7 would require more moves and complicate the solution. This is an indication that first trying to reach 8 first and then 7 looks like a promising option, which could maybe be completed in just 4 moves. But of course, with 4 moves, we also modify the state of the intermediary numbers of which we land. So, in order to keep only 7 and 8 lit and no other number, we must ensure that we land on the same intermediary number both when first go for 8, and then when we later reach 7.

This works if we move like this: first, 3 steps (we light up 4), then 4 steps (we light up 8), then 4 steps again (we turn off 4), and finally 3 more steps to light up 7.

1st move (3 steps)	2nd move (4 steps)	3rd move (4 steps)	4th move (3 steps)
			

Here is a systematic list of all possible sequences of moves, which can confirm that 4 is the optimal solution.

If the spy only makes 1 move:

1st move	Numbers lit
3 steps	4
4 steps	5

Using 1 move, the spy can light up only one number.

If the spy makes 2 moves:

1st move	2nd move	Numbers landed on	Numbers lit at the end
3 steps	3 steps	4, 7	4, 7
3 steps	4 steps	4, 8	4, 8
4 steps	3 steps	5, 8	5, 8
4 steps	4 steps	5, 1	5, 1

Using 2 moves, the spy can light up two numbers, but those 2 numbers are not 7 and 8.

If the spy makes 3 moves:

1st move	2nd move	3rd move	Numbers landed on	Numbers lit at the end
3 steps	3 steps	3 steps	4, 7, 2	4, 7, 2
3 steps	3 steps	4 steps	4, 7, 3	4, 7, 3
3 steps	4 steps	3 steps	4, 8, 3	4, 8, 3
3 steps	4 steps	4 steps	4, 8, 4	8
4 steps	3 steps	3 steps	5, 8, 3	5, 8, 3
4 steps	3 steps	4 steps	5, 8, 4	5, 8, 4
4 steps	4 steps	3 steps	5, 1, 4	5, 1, 4
4 steps	4 steps	4 steps	5, 1, 5	1

Using 3 moves, the spy can only light up either one or three numbers. It is impossible to light up 7 and 8.

If the spy makes 4 moves:

1st move	2nd move	3rd move	4th move	Numbers landed on	Numbers lit at the end
3 steps	3 steps	3 steps	3 steps	4, 7, 2, 5	4, 7, 2, 5
3 steps	3 steps	3 steps	4 steps	4, 7, 2, 6	4, 7, 2, 6
3 steps	3 steps	4 steps	3 steps	4, 7, 3, 6	4, 7, 3, 6
3 steps	3 steps	4 steps	4 steps	4, 7, 3, 7	4, 3
3 steps	4 steps	3 steps	3 steps	4, 8, 3, 6	4, 8, 3, 6
3 steps	4 steps	3 steps	4 steps	4, 8, 3, 7	4, 8, 3, 7
3 steps	4 steps	4 steps	3 steps	4, 8, 4, 7	8, 7
3 steps	4 steps	4 steps	4 steps	4, 8, 4, 8	-
4 steps	3 steps	3 steps	3 steps	5, 8, 3, 6	5, 8, 3, 6
4 steps	3 steps	3 steps	4 steps	5, 8, 3, 7	5, 8, 3, 7
4 steps	3 steps	4 steps	3 steps	5, 8, 4, 7	5, 8, 4, 7
4 steps	3 steps	4 steps	4 steps	5, 8, 4, 8	5, 4
4 steps	4 steps	3 steps	3 steps	5, 1, 4, 7	5, 1, 4, 7
4 steps	4 steps	3 steps	4 steps	5, 1, 4, 8	5, 1, 4, 8
4 steps	4 steps	4 steps	3 steps	5, 1, 5, 8	1, 8
4 steps	4 steps	4 steps	4 steps	5, 1, 5, 1	-

Therefore, the minimum number of moves the spy needs to do in order to light up only 7 and 8, from the starting position, is 4 moves.

Background information

In this task, we are given rules (instructions) that need to be repeated multiple times. However, we don't know how many times each rule (instruction) needs to be used to reach the goal.

One approach is to write a program that generates all possible sequences of moves (a total of 4 in this task) and checks the output for each case. Computer scientists refer to this as a *brute-force search*. However, take note that this method becomes impractical when we don't know the number of repetitions or when the number of possibilities overwhelms our memory.

In this particular case, it is similar to the "trial and error" method, where we attempt to find the solution by iteratively following the rules. It's important to note that the first solution found through trial and error doesn't always yield the optimal solution, unless we start from the most optimal implementation of the instruction, in this case starting from using only 1 move and then increasing the number of moves from there.

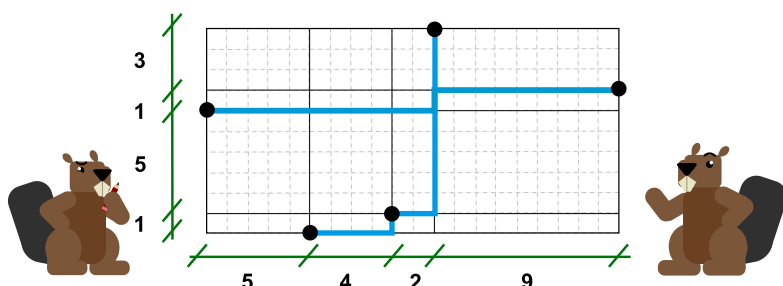
In informatics, number systems play a crucial role in representing and manipulating data. The circular arrangement of numbers in the task, similar to a clock, represents a specific number system. Computers use different number systems, such as binary (base-2), decimal (base-10), or hexadecimal (base-16), to represent and process information. The numbers in the task are like the digits on a clock, but instead of going from 1 to 12, they go from 1 to 8.

Understanding number systems and their representations is fundamental in computer science and helps computers store, process, and transmit data efficiently.

Channel plan

Maurice, an architect, presents his plan for the construction of a water channel to connect the five dwellings of a village. The dwellings are indicated on the map with black circles.

Maurice finds it nice to follow the horizontal and vertical lines that cross each of the dwellings. On his 20x10 drawing he indicates the layout of the planned channel in blue, and reports the distances of channel sections on the axes.



Jasmin notes that this channel is a total of 36 units long... however, to work less, it could be made shorter!

Question

How long is the shortest channel made up of horizontal and vertical segments that can connect the five dwellings?

Fill in the number and click "Save" when done.

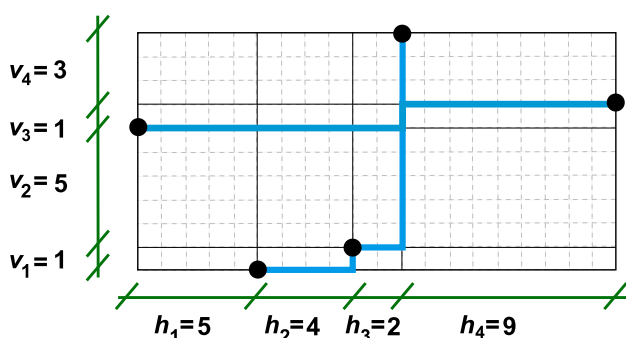
Answer:

Save

Explanation

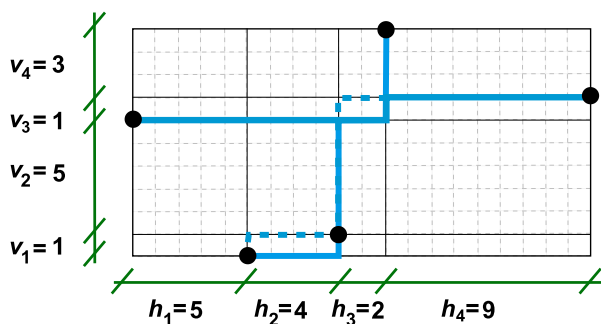
The correct answer is 34.

Let's consider the lengths of the horizontal ($h_1 = 5$, $h_2 = 4$, $h_3 = 2$, $h_4 = 9$) and vertical ($v_1 = 1$, $v_2 = 5$, $v_3 = 1$, $v_4 = 3$) sections, each of which must be "covered" by at least one channel stretch; therefore, the minimum length cannot be less than 30. (Note that this goal could be achieved if the dwellings were arranged on at most one horizontal and one vertical line.)

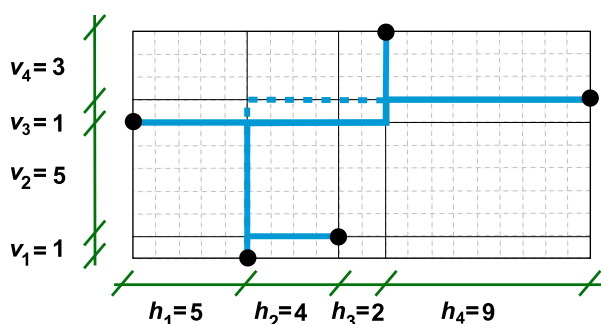


In Maurice's project (see figure above), sections h_2 and h_3 are covered twice, and indeed the overall length of the channel is $30 + 4 + 2 = 36$.

To get an optimal solution just move the channel stretch covering section v_2 as shown in the next figure (where dotted segments show equivalent alternatives to run across two sides of a rectangle), so that only section h_2 is covered twice, and the total length is $30 + 4 = 34$.



Moving the channel stretch covering section v_2 further to the left, two other optimal solutions are obtained:



Clearly, covering section v_2 twice is not convenient, since $v_2 > h_2$. The remaining possible locations for the only (vertical) stretch covering section v_2 are at the left and right ends of the map; in these cases, at least section h_1 or section h_4 (both greater than h_2) would be covered twice, respectively. This shows that the minimum overall length is 34, with h_2 covered twice.

Background information

Let's see how this task fits into computational geometry, a branch of computer science devoted to the study of algorithms which can be stated in terms of geometry; indeed, it is one of the oldest fields of computing.

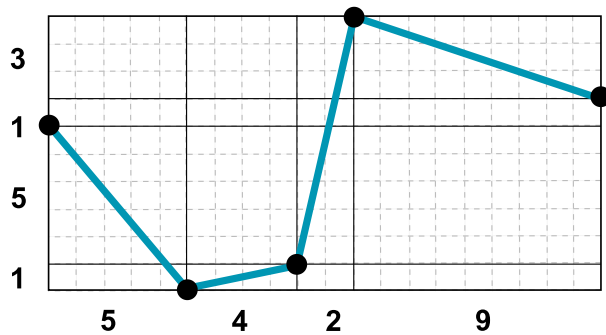
Let's start with this aim: we want to connect some given points of the plane to each other, by means of a tree, a structure composed of several segments (the edges of the tree), so that between any two of the given points there is only one path joining them. Of course, the segments' end points (the vertices of the tree) include the given points, and possibly other points, called Steiner points (Jakob Steiner was a Swiss mathematician who lived in the nineteenth century).

The rectilinear Steiner tree problem is to find a tree of minimum total length composed solely of horizontal and vertical line segments with respect to a predetermined Cartesian reference system. This problem arises in wire routing, the planning of connections between the thousands of components of a VLSI system; indeed, in the design of electronic (VLSI) circuits, wires are often constrained to run only in vertical and horizontal directions. The problem of determining the minimum total wire length is difficult, precisely NP-complete (Garey and Johnson, 1977).

On this topic, a lot of papers and entire books have been written, as well as efficiently solvable special cases and heuristic or approximate algorithms giving near-optimal solutions have been developed.

In 1966, Maurice Hanan proved that there exists an optimal rectilinear Steiner tree with all Steiner points chosen from the intersection points of horizontal and vertical lines drawn from the given points, precisely the case proposed in this task; of course, the resulting grid, called Hanan grid, depends on the chosen Cartesian reference system.

A remark, to conclude: if the beavers had simply wanted to connect the dwellings of their village, without introducing other intersection points, they would have had to find a minimal spanning tree; in the present case, it would result as shown in the next figure, with a total length of about 30.64 units, and therefore with a saving of about 10%.



To solve the minimal spanning tree problem in general, several efficient algorithms are known (e.g., Kruskal and Prim-Jarník).

This task requires the abilities to configure a scenario in different ways, respecting the constraints imposed and choosing objects from a given set to achieve a certain goal, and to see how far the search for a solution can go. It stimulates looking at a problem under different constraints and using heuristic reasoning to discover a solution, compare different solutions, and understand how close a solution can be to an optimal one.

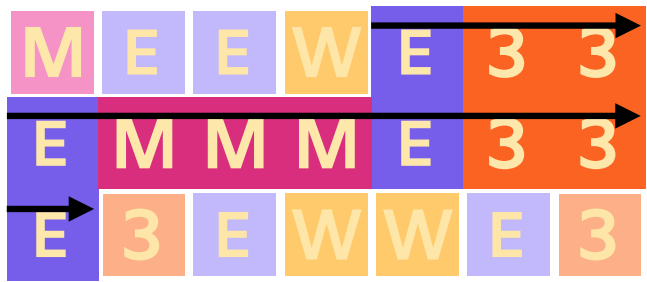
This task can be useful to illustrate how a change in requirements (from channels directly connecting dwellings, as in spanning tree problem, to channels formed by orthogonal stretches) can make the optimization problem (much) more difficult to solve. Provided the number of dwellings is small, you can proceed as suggested in the Answer Explanation section, in order to find a solution and try to improve it, with different combinations. You can then compare your solution with the one provided by a software package such as *GeoSteiner*.

Palindrome Passwords

Hannah and Otto are planning a meeting for their club. The secret meeting time is encoded within a code block of letters and numbers. The code block is read in rows starting from the top, each row from left to right, as if they formed a single sequence.

The secret meeting time is the length of the longest palindrome within that code block. A palindrome is a sequence of letters and numbers that reads exactly the same forwards and backwards.

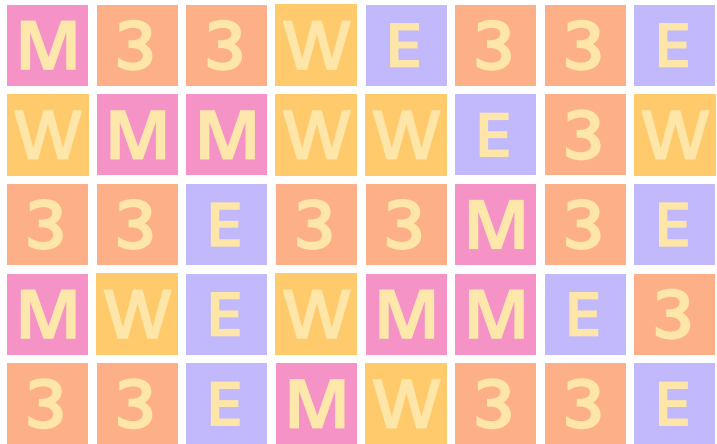
For example, the code block from last week is shown below. The longest palindrome is "E33EMMME33E", which has a length of 11. Therefore, the secret meeting time was 11:00.



Question

Here is the code block for next week's meeting. Which letters and numbers make up the palindrome that indicates the secret meeting time?

Click on the blocks to select them and click "Save" when you are done.



Save

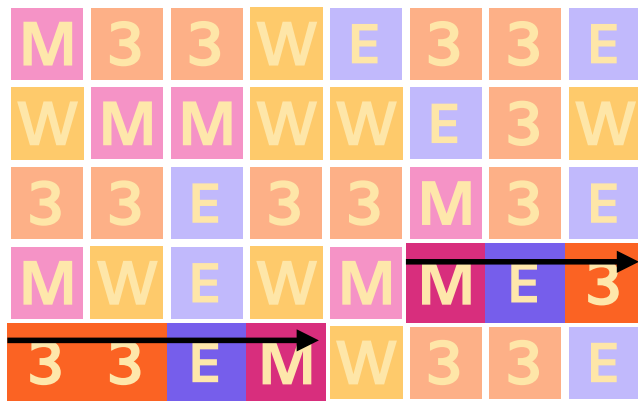
Erase

Explanation

The longest palindrome is 7 blocks long. Therefore the secret meeting time is 7:00.



There are quite a few palindromes in this week's image, but **M E 3 3 3 E M** is the longest. It contains the longest set of blocks in a row, which are the same when read both forwards and backwards.



Background information

Making and sharing codes so that only certain people can read them is a type of **encryption**.

Encryption is used in computer science to protect information, so that it cannot be accessed and understood by everyone who sees it. When people try to read encrypted data without the key to understanding it, it can just look like a scrambled mess of information.

For example, when we buy something online, an algorithm is used to encrypt our payment information. This helps to protect sensitive information from hackers. Encryption is similar to putting a lock on a diary, so that only you and the people you trust can read it.

This question helps us to develop and use search algorithms. We can simply go through all possible paths to check which ones are palindromes, but this takes a very long time. We can instead develop strategies, such as starting with all the 2 and 3 length palindromes, and growing them by adding start and end positions. This helps us find a solution much more quickly.

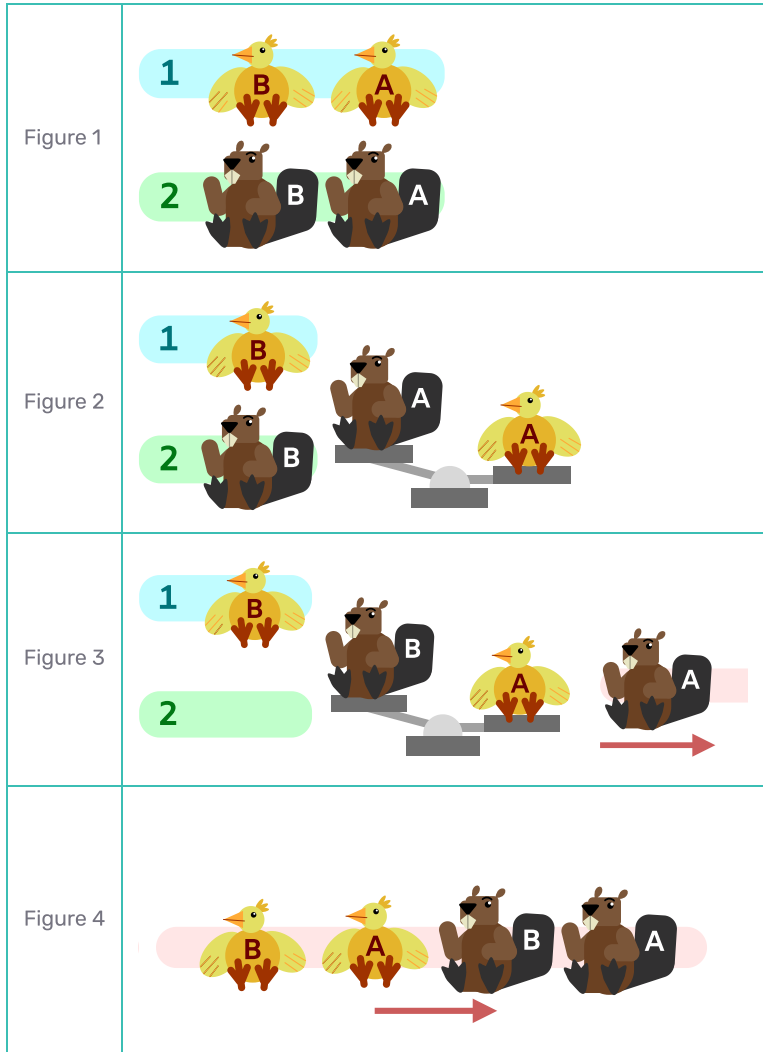
In looking for palindromes, this question also makes us use our pattern recognition skills. Palindromes always follow a pattern, and once we know what the features of this pattern are, it is easier for us to seek them out.

Line Up

The Riverland Wrestling Competition is today. All animals from all teams participate in order based on their weight. To line them all up correctly, first, animals from each team line up based on their weight, the lightest on the right and the heaviest on the left. Then, these lines are merged into one, two lines at a time, following the steps below, until all lines are merged:

- 1. The first animal in each of the two lines steps on the scales.
- 2. The lighter one of the two queues up at the end of the new line.
- 3. The heavier animal stays on the scales while a new animal from the opposing team steps onto the other side of the scales.
- 4. The animals keep repeating Step 2 and 3 until all animals are merged into one new line.
- 5. When only animals of the same team are left to be weighed, they simply join the new line in the order that they are in.

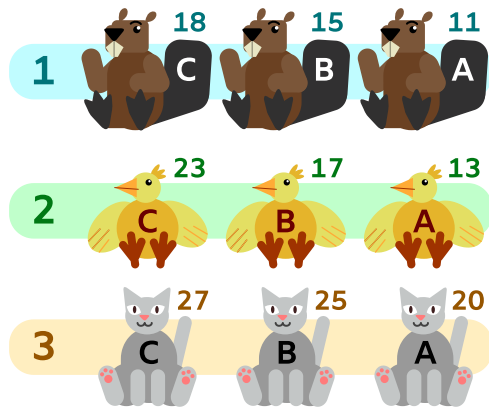
The figures below show how two lines merge into one. The red arrow indicates the direction of the front of the line.



It took two weight comparisons to merge these two lines (figure 2 and 3) into one line (figure 4).

Question

Three teams are lined up, ready to be weighed, as shown below. The numbers above the animals indicate their weight. Which two teams will take the highest number of weight comparisons to merge?



team 1 and team 2

team 2 and team 3

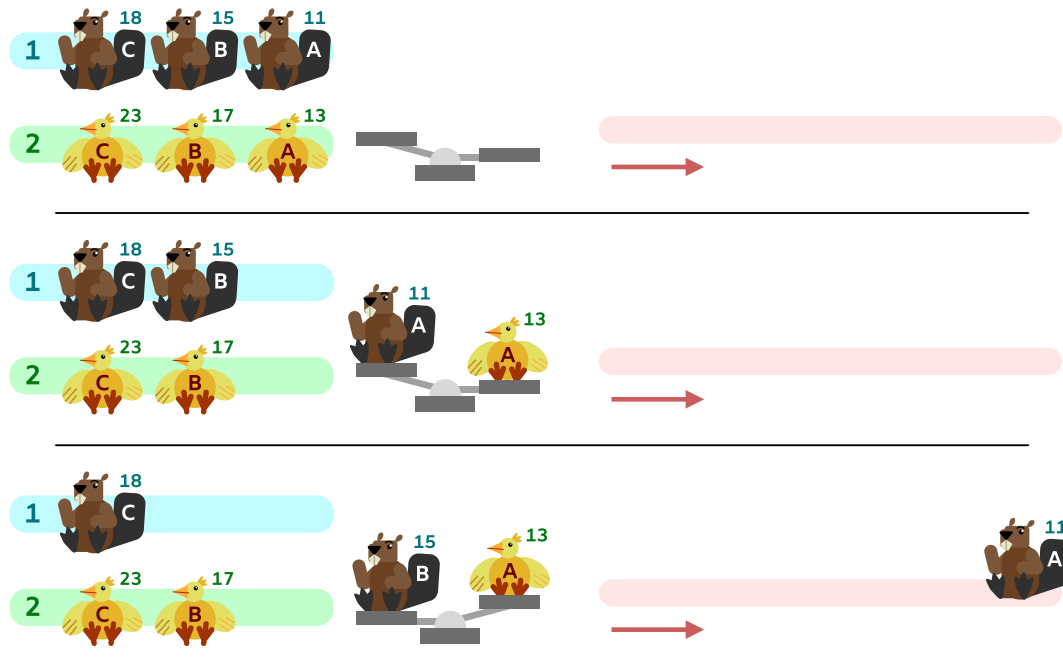
team 1 and team 3

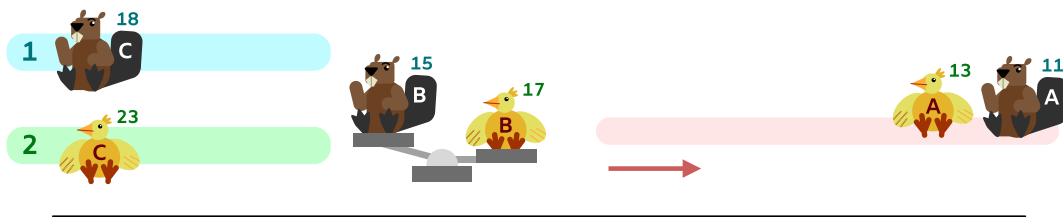
they all take the same number of weight comparisons to merge

Explanation

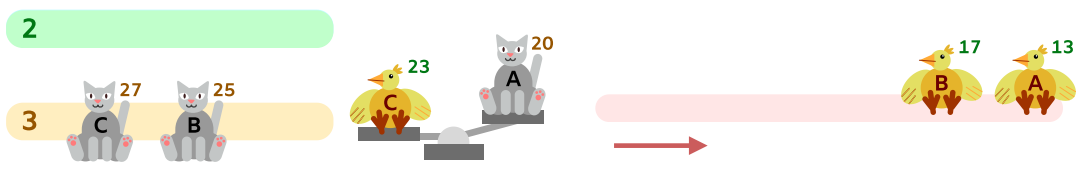
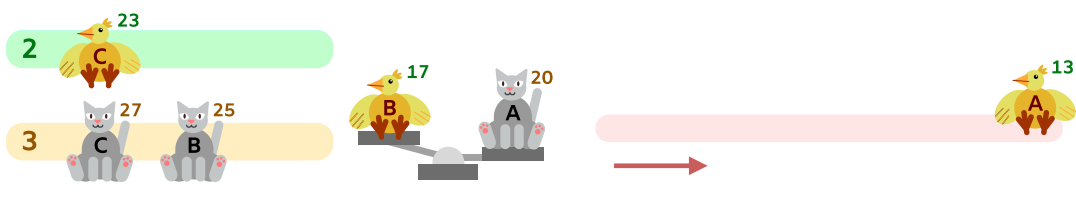
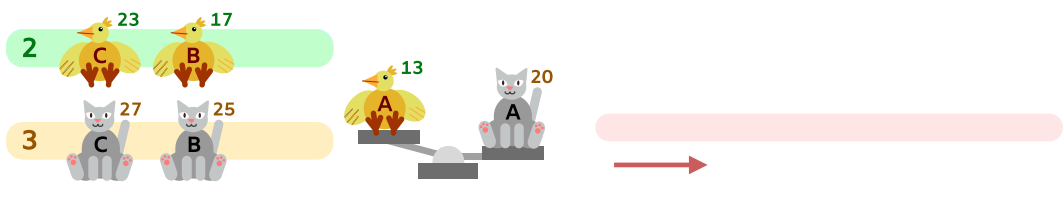
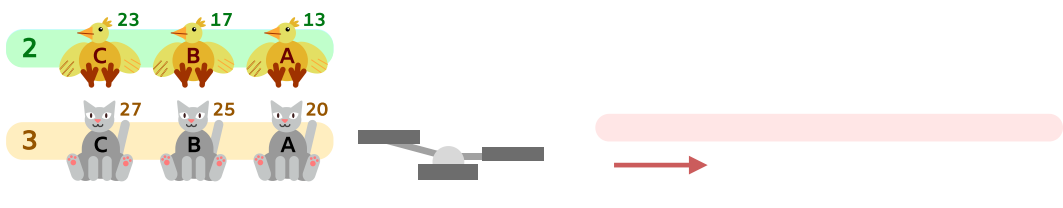
The correct answer is A.

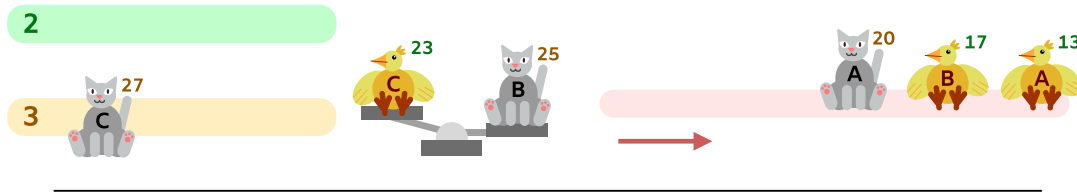
It takes five comparisons to merge team 1 and team 2 as shown below:



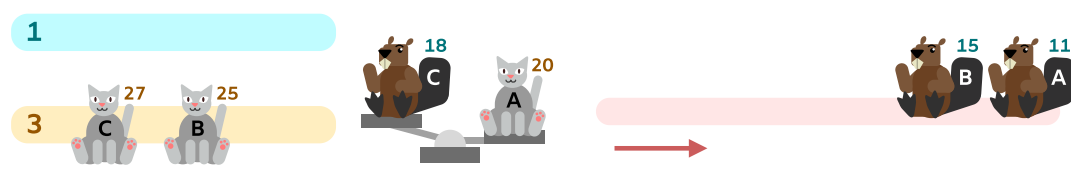
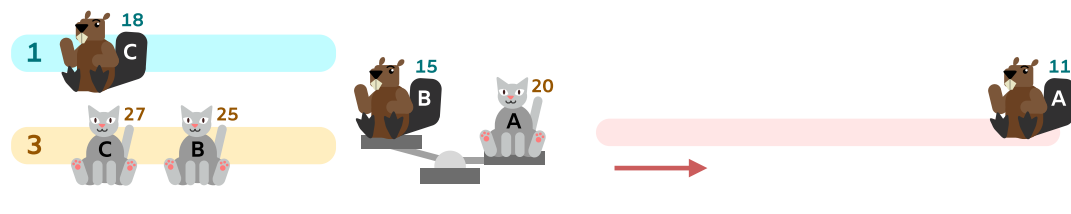
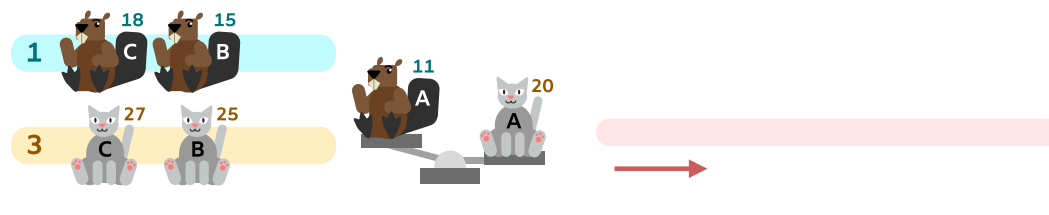
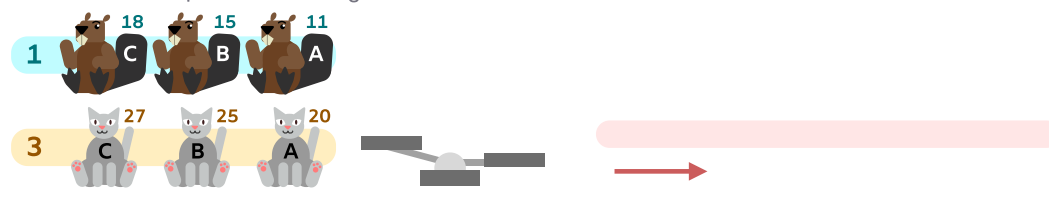


It takes three comparisons to merge team 2 and team 3 as shown below:





It takes four comparisons to merge team 1 and team 3 as shown below:



Actually it is not necessary to execute the merge sorting algorithm on the three pairs of teams; it suffices to observe that:

1. the lowest number of comparisons are needed when the heaviest animal in one line is lighter than the lightest animal in the other line;
2. the highest number of comparisons are needed when each animal in one line is heavier than the animal in the same position in the other line, but lighter than the next animal in the other line; that is, for each animal, the next higher weight animal is in the other line.

Team 1 and Team 3 have property 1. Team 1 and Team 2 have property 2, and are therefore the ones that take the highest number of comparisons.

Background information

Computer scientists use sorting algorithms to put a group of elements into order. One of the sorting algorithms is **merge sort**, which is the algorithm used in this task to sort all animals participating in the contest. This algorithm is based on a technique called **Divide and Conquer** by computer scientists.

Divide and Conquer breaks down a problem (like sorting all animals) into smaller sub-problems (like sorting subgroups of animals), solves each sub-problem, and then combines the solutions to solve the original problem. When applicable, this technique can boost the efficiency of algorithms.

To solve this task, students need to use **algorithmic thinking** to understand how two lines are merged and apply the algorithm to merge teams.

When students try to calculate the number of comparisons of merging any two lines, they can use **abstraction** to figure out what are the factors that affect the total number of comparisons: in this case if the weights increase alternating in the two lines or the weights in one line are all smaller than the weights in the other line.

The background is a solid dark blue color. In the upper left corner, there are several overlapping organic, blob-like shapes in lighter shades of blue, ranging from a medium blue to a very light, almost white-blue. These shapes are positioned in the top left and top center of the page.

2024 Bebras Challenge Round 2 | Senior Year 11 & 12

BebrasGPT

BebrasGPT is a chat bot that has been developed to produce three-word sentences. It does this by predicting the next word based on the previous sequence of words. Each word is chosen one by one, with the next word being chosen based on the probability of it following the current sequence. The tables below show some of these probabilities.

Probabilities for second word:

	"love"	"hate"
"Cats"	0.7	0.3
"Dogs"	0.6	0.4

Probabilities for third word:

	"swimming"	"running"
"Cats love"	0.3	0.8
"Cats hate"	0.9	0.1
"Dogs love"	0.7	0.3
"Dogs hate"	0.1	0.9

For example, if the sentence starts with the word "Cats", the probability of the 3-word sentence being "Cats love running" is 0.56 because:

- the probability of the second word being "love" if the previous word is "Dogs" is 0.7,
- and the probability of the next word being "running" if the previous sequence is "Dogs love" is 0.8,
- so, because the model predicts words one by one, the probability is $0.7 * 0.8 = 0.56$.

Question

If a sentence starts with the word "Dogs" what is the most likely output of BebrasGPT?

"Dogs hate swimming"

"Dogs hate running"

"Dogs love swimming"

"Dogs love running"

Explanation

The correct answer is: "Dogs love swimming."

The probabilities of each sentence are the following:

"Dogs hate swimming", $0.4 * 0.1 = 0.04$

"Dogs hate running", $0.4 * 0.9 = 0.36$

"Dogs love swimming", $0.6 * 0.7 = \mathbf{0.42}$

"Dogs love running", $0.6 * 0.3 = 0.18$

C has the highest probability among the choices.

Background information

This exercise involves an example of an artificial model for language modeling and the production of text. This is a probabilistic model, meaning that each output of the model is based on probabilities. There are many types of models. In this case, the model is producing one word at a time, based on the entire sequence of previous words, which is similar to models like ChatGPT. Of course these models are much, much larger than the model shown in the exercise.

One related important aspect in artificial intelligence and machine learning is the difference between two type of algorithms:

- the learning algorithm,
- and the inference algorithm.

The learning algorithm is used to "train" the model. This corresponds, in our example, to finding which values of the probabilities should be in the table above. In our case, this has already been done. The model is already trained. Usually, this is the most difficult part in terms of theory. For example, ChatGPT was training for many months on several different GPUs, which cost millions of dollars.

The inference algorithm is the one we use after training, to produce the output. For example, when you type the prompt "Hi, ChatGPT. What is machine learning?", the model will take that sequence of words and use something equivalent to a very large table of probabilities to output the answer to the prompt. Although the cost of running the inference algorithm once is much smaller than the training, if the model is used by millions of people everyday (like ChatGPT), the cost of inference is higher than that of training (which was a one time cost).

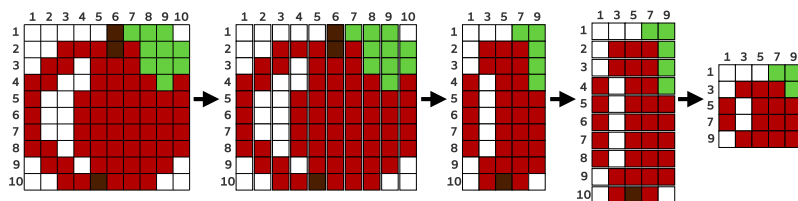
This task requires algorithmic thinking as the students figure out the next word one at a time, and finding the 3-word sentence with the largest probability. Students are probably familiar with the text predict functions of their smartphones but this task allows them to think through how it works.

Snail Compress

Snails have a special technique to shrink their captured images.

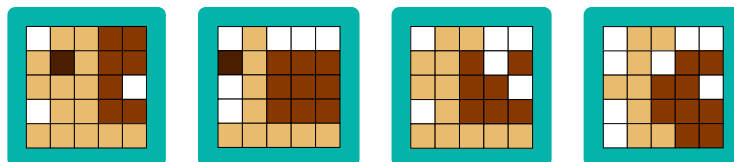
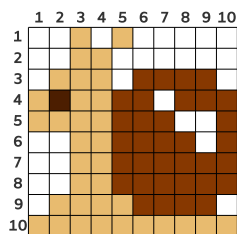
First, they will cut the original image into 10 equally sized strips vertically. Then, they will assemble the odd-numbered vertical strips to create a new image.

Next, they will cut the new image horizontally into 10 equally sized strips. Then, they will assemble the odd-numbered horizontal strips to create a complete shrunken image.



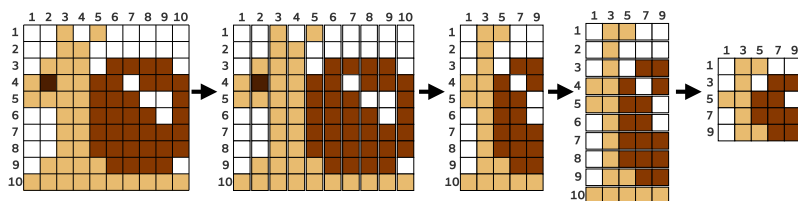
Question

What image will the snails obtain after shrinking the given image using this technique?



Explanation

Following the same procedure as with the picture of the apple, the snail image is compressed in the following way:



First all the even-numbered columns will be erased, then all the even-numbered rows will be erased, which means that only the cells from odd-numbered columns, that are also in odd-numbered rows will remain.

We can see that the correct answer is D, as it is the only image that uses the 9th row for compression of the picture.

Background information

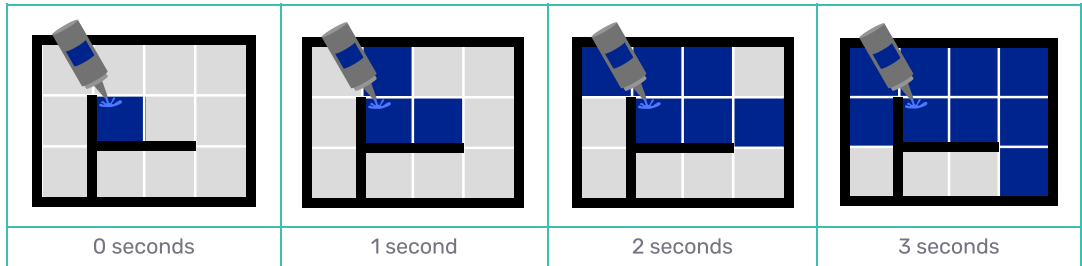
Compression of an image refers to the process of reducing the size of an image file by eliminating or minimising some of the unnecessary data in the image while still maintaining sufficient image quality for use. This process helps to reduce the file size of images, making them easier to store or transmit.

This is lossy compression algorithm, as it completely eliminates some of the rows and columns in the original image. JPG images are also stored using a lossy compression algorithm.

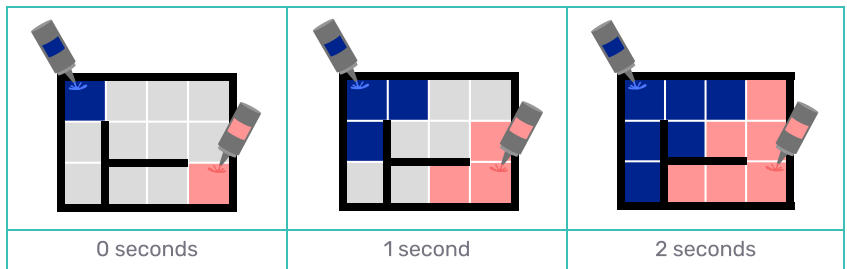
In contrast, a lossless compression algorithm only minimises the data. PNG images are stored using a lossless compression algorithm.

Watercolour

When painters pour watercolour into a maze, the colour will spread to neighbouring empty squares every second. The colour cannot spread through the walls, as you can see below.

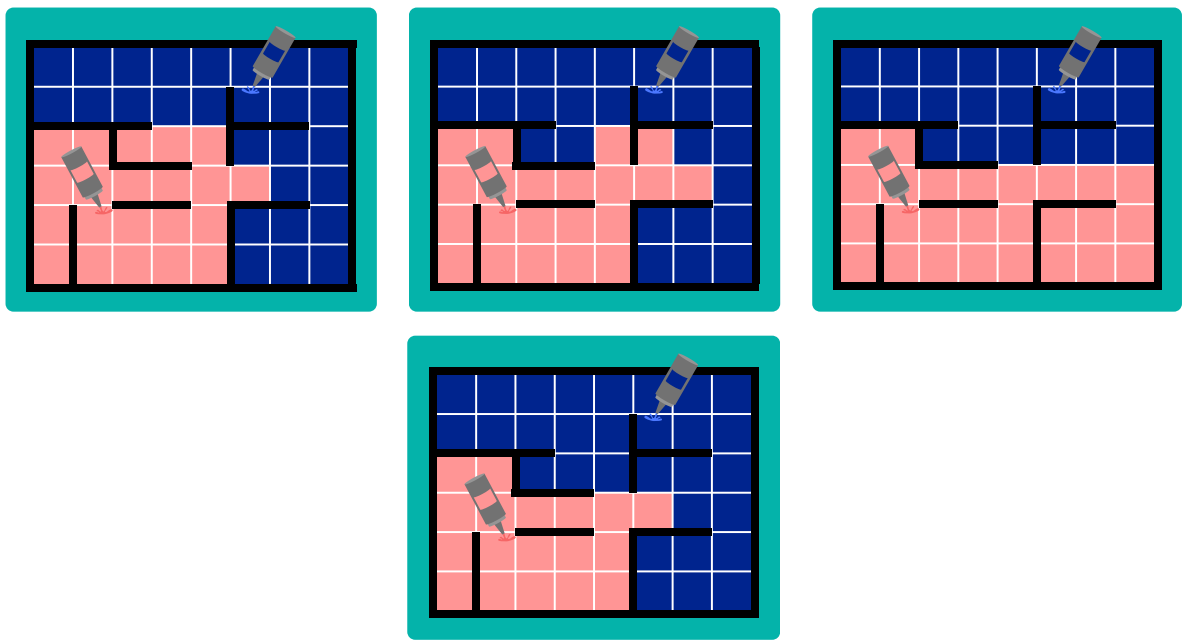
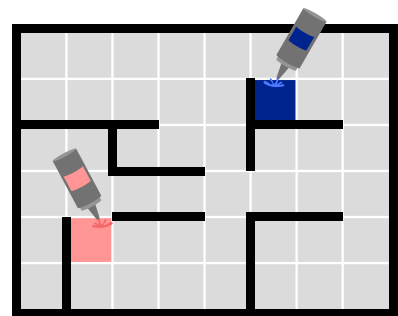


If the painters pour more than one watercolour into the maze, the first colour that reaches a square will fill it completely. When two colours reach a square at the same time, the square takes the darker colour (blue).



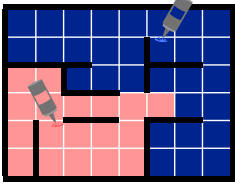
Question

The painters poured two colours into the maze below. What does the maze look like when all the squares are filled with colour?

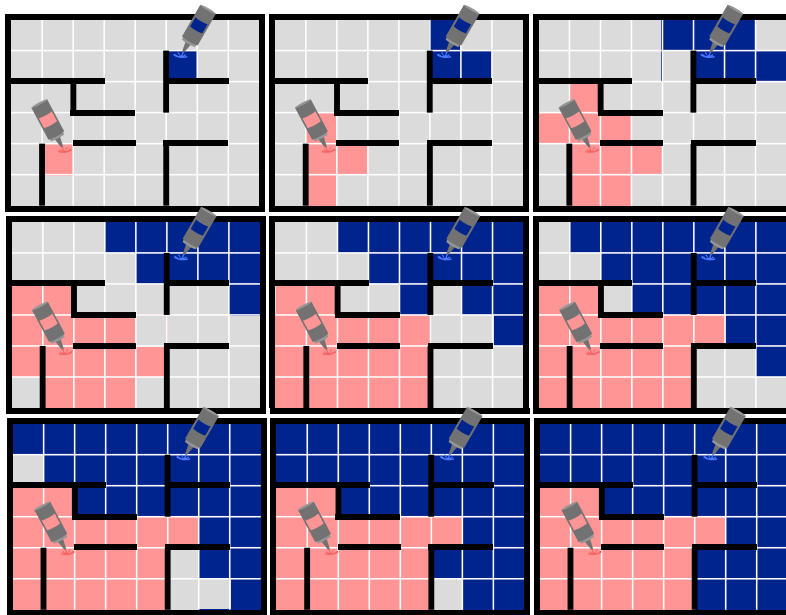


Explanation

The correct answer is



The image below shows the state of the maze second by second:



Background information

The setting of the task resembles a two dimensional array, which basically means a table with rows and columns. This way of representing the data is quite functional to simulate the state of the maze second after second and solve the task.

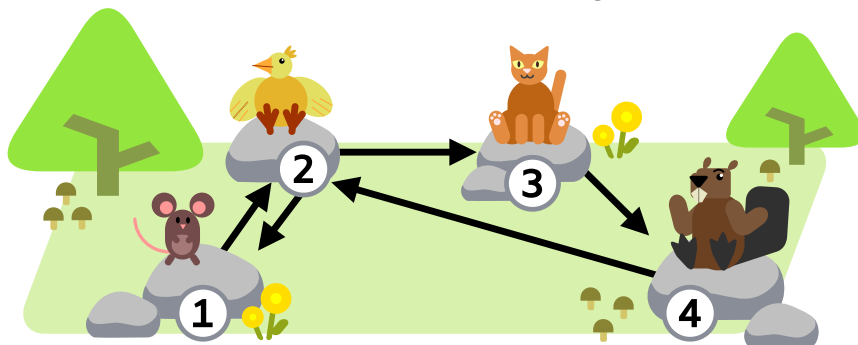
However, it is also possible to represent it as a graph in which each square is connected to its neighbors. This graph allows us to identify quickly which color will reach a square without simulating the whole scenario second by second. This approach using graphs is the same as performing a breadth-first search.

The breadth-first search (BFS) algorithm is one way of searching a tree or graph for a node that meets a set of criteria. It starts at any given node of a graph and visits all nodes at the current depth level before moving on to the nodes at the next depth level. Breadth-first search can be used to solve many problems in graph theory.

Algorithmic thinking is key in solving this tasks since students have to understand how the setting works in order to simulate its situation second by second.

Jumping Together

Beaver lives by rock number 4. In his neighbourhood there are three other rocks where his friends Bird, Cat and Mouse live. The animals have decided to have a party at Beaver's rock. In order to do this the animals can jump from rock to rock, following the arrows, according to the drawing below. Because they all want to be equally tired when the party starts, they all want to jump around and arrive on Beaver's rock having all made the exact same number of jumps (including Beaver).



Question

What is the minimum number of jumps that the animals must make so they all arrive on Beaver's rock with the same amount of jumps?

Fill in the number and click "Save" when you are done.

Answer:

Save

Explanation

By making a single jump, the cat can reach Beaver, but Beaver would be gone, and both the mouse and the bird have not yet arrived.

Please note the following observations:

- Beaver and *mouse* are both one jump away from rock 2. Their second jump could always be the same. So any number that works for Beaver will also work for mouse.
- The lowest numbers for *cat* are 1, 4 and 6.
- The lowest numbers for *bird* are 2, 4, 5 and 6

Looking at it from the point of view of Beaver:

- Beaver could circle around in three jumps, but *cat* cannot do three.
- Beaver could also do five jumps (4 → 2 → 1 → 2 → 3 → 4), but *cat* also cannot do five.

It turns out that every animal can do it in six jumps:

- Beaver can do six jumps (4 → 2 → 3 → 4 → 2 → 3 → 4)
- Cat can do six as well: (3 → 4 → 2 → 1 → 2 → 3 → 4)
- Mouse follows Beaver with six jumps (1 → 2 → 3 → 4 → 2 → 3 → 4)
- Bird can also make six jumps: (2 → 1 → 2 → 1 → 2 → 3 → 4)

Obviously the animals can also do it in more jumps, they could do their loop of 6 jumps and then add another 3, 5, 6, 8, 9, 10, 11, ... jumps. It's interesting to verify that 1, 2, 4 and 7 extra jumps are not possible. Can you find out why?

Background information

This task can be represented as a graph in computer science, where each rock corresponds to a **vertex**, represented by a numbered circle, and the arrows indicate **directed edges** in the graph. The objective is to find a **directed walk** from a starting vertex to the destination vertex.

In this task, it is allowed to visit the same edge multiple times in order to reach the destination vertex. The table below displays the shortest path from each vertex from the start vertex to the destination vertex, along with the length of the shortest path:

Starting vertex	The shortest path to vertex 4	Length of the shortest path
1	1→2→3→4	3
2	2→3→4	2
3	3→4	1
4	-	0

Furthermore, the table below shows the cycles, which represent the shortest path starting from each vertex back to the same vertex:

Starting vertex	Cycle starting from the node	Length of the cycle
1	1→2→1	2
2	2→3→4→2	3
3	3→4→2→3	3
4	4→2→3→4	3

By combining the shortest path from each vertex to the destination and the cycles of the intermediate vertices, all possible directed walks from a vertex to the destination can be enumerated. For example, the table below illustrates the directed walks from each vertex to vertex 4 with a length of 6:

Starting vertex	The directed walk of length 6 to vertex 4 by combining the above paths/cycles
1	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, $4 \rightarrow 2 \rightarrow 3 \rightarrow 4$
2	$2 \rightarrow 1 \rightarrow 2$, $2 \rightarrow 1 \rightarrow 2$, $2 \rightarrow 3 \rightarrow 4$
3	$3 \rightarrow 4$, $4 \rightarrow 2(2 \rightarrow 1 \rightarrow 2) \rightarrow 3 \rightarrow 4$
4	$4 \rightarrow 2 \rightarrow 3 \rightarrow 4$, $4 \rightarrow 2 \rightarrow 3 \rightarrow 4$

This approach is a kind of recursion, which is a concept or process depends on a simpler version of itself. To find the pattern of repeating, the length of the directed walks from starting vertex to the destination can be calculated without needing to illustrate all the solutions.

Correcting error

In one country, a phone number consists of 11 digits. However, people tend to make mistakes when writing them down, so there is a rule to correct ONE mistake. All phone numbers must satisfy the following:

- 8th digit = the last digit of the sum of the first 7 digits (1st-7th).
- 9th digit = the last digit of the sum of the first 4 digits (1st-4th).
- 10th digit = the last digit of the sum of the 1st, 2nd, 5th & 6th digits.
- 11th digit = the last digit of the sum of the 1st, 3rd, 5th & 7th digits.



For example, 12345678046 is a correct phone number, because

- the 8th digit is 8 ($1+2+3+4+5+6+7 = 28$)
- the 9th digit is 0 ($1+2+3+4 = 10$)
- the 10th digit is 4 ($1+2+5+6 = 14$)
- the 11th digit is 6 ($1+3+5+7 = 16$)

Question

Someone made ONE mistake when writing down the phone number 12312316710. What was the correct phone number?

Fill in the number and click "Save" when you are done.

Answer:

Save

Explanation

The correct answer: 12315316710.

The original phone number is 12312316710.

1. First, let's examine the last four digits (8th-11th), which are 6, 7, 1, 0:

- the 8th digit (6) is **incorrect** (because $1+2+3+1+2+3+1 = 13$)
- the 9th digit (7) is correct (because $1+2+3+1 = 7$)
- the 10th digit (1) is **incorrect** (because $1+2+2+3 = 8$)
- the 11th digit (0) is **incorrect** (because $1+3+2+1 = 7$)

As shown above, the last four digits (8th-11th) have 3 incorrect digits, so the one mistake cannot be there. Since the last four digits (8th-11th) are actually based on the first seven digits, the mistake must be in one of the first seven digits (1st-7th).

2. We then examine the first seven digits (1st-7th), which are 1, 2, 3, 1, 2, 3, 1:

- We already know that the 9th digit (which equals to the sum of 1st-4th digits) is correct, so the first four digits (1st-4th) must be correct.
- Let's examine the other three digits (5th-7th). According to the rules:
 - the 5th digit contributes to the 8th, 10th, 11th digits
 - the 6th digit contributes to the 8th and 10th digits
 - the 7th digit contributes to the 8th and 11th digits
- Since **8th, 10th and 11th digits are incorrect**, the mistake must be in the 5th digit.

3. Now let's find out what the 5th digit should be:

- We know that the 5th digit can only be a number between 0 to 9.
- There are 2 ways to find the correct number for the 5th digit:
 - Trial and error: If we try to use the numbers 0 to 9 one by one as the 5th digit, only the number 5 fits the rule and can make the 8th, 10th and 11th digits correct. Therefore, the 5th digit should be 5 (instead of 2).
 - Mathematical comparison: If we compare the 8th, 10th and 11th digits and their sums:

Position	Number	Original Sum	Corrected Sum (to match the Number)	Difference
8th digit	6	13	16	+3
10th digit	1	8	11	+3
11th digit	0	7	10	+3

As shown above, the difference between all pairs of original sum and corrected sum is +3. This means that the original 5th digit (2) should be increased by 3 to become 5 ($2+3=5$).

Lets check this corrected phone number: 1231**5**316710.

- the 8th digit (6) is **now correct** (because $1+2+3+1+5+3+1 = 16$)
- the 9th digit (7) is correct (because $1+2+3+1 = 7$)
- the 10th digit (1) is **now correct** (because $1+2+5+3 = 11$)
- the 11th digit (0) is **now correct** (because $1+3+5+1 = 10$)

Background information

Last 4 digits of the phone number in this task is called error correction code.

In this case, if only ONE mistake is made, we have enough information to identify and correct it, no matter which digit is wrong (including the last 4 digits).

Error correction codes are widely used in transmitting messages, when communication channels are noisy or unreliable, and we don't want to (or can't) repeat the message again.

Bebras Ball

Today is the annual Bebras Ball tournament.

Sixteen players compete in four rounds in order to determine their overall rank from 1st place to 16th place:

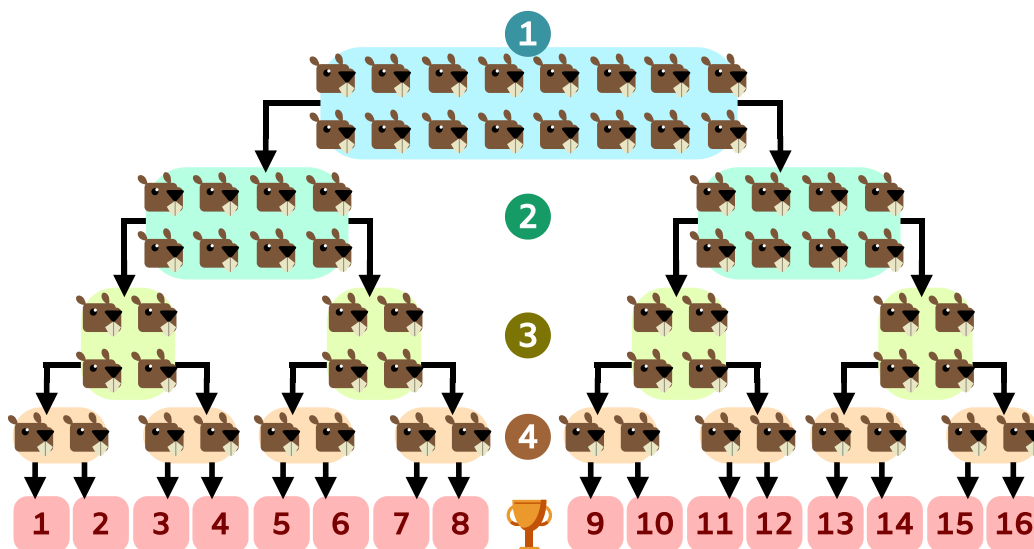
- All sixteen players compete together in round 1, but after each round, the players split up.
- The winning players follow the left arrow to their next round of competition (or final rank).
- The losing players follow the right arrow to their next round of competition (or final rank).

For example, a player who wins during rounds 1 and 2, but loses during rounds 3 and 4, will receive a rank of 4th place.

Question

Noro was a player in the Bebras Ball tournament. If Noro lost during exactly one round, which ranks might they have received?

Select the ranks and press "Save" when you are done.



Save

Erase

Explanation

Correct Answer: 9th, 5th, 3rd, 2nd

Since there are four rounds, and Noro lost during exactly one round, there are four possible scenarios.

Scenario 1: Noro lost during round 1 and won during all others. Thus, Noro would follow the arrows "right, left, left, left" which leads to the rank 9th place.

Scenario 2: Noro lost during round 2 and won during all others. Thus, Noro would follow the arrows "left, right, left, left" which leads to the rank 5th place.

Scenario 3: Noro lost during round 3 and won during all others. Thus, Noro would follow the arrows "left, left, right, left" which leads to the rank 3rd place.

Scenario 4: Noro lost during round 4 and won during all others. Thus, Noro would follow the arrows "left, left, left, right" which leads to the rank 2nd place.

Background information

The diagram in this task, which models the tournament, is a type of structure called a *decision tree*. The top of the tree (or root) is where the decision process begins. From there, we select a branch to follow depending on the answer to a decision question.

In this task, the decision question is “did I win or lose during the round?” Answers of “win” mean we follow the left branch and answers of “lose” mean we follow the right branch.

When we run out of branches we say that we have reached the leaves of the decision tree. The leaves represent the final outcomes. In this task, the final outcomes are the ranks.

If you have ever tried to identify someone’s secret number by making a guess and having them respond with “higher” or “lower”, you have also used a decision tree.

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”, and “fins” are used as decision questions. With repeated training, the program develops enough distinguishing features to correctly classify an image.

One possible way to solve this task includes counting how many rounds Noro would have lost in order to receive each of the 16 ranks, and then selecting a rank if he would have lost exactly once. This process is called evaluation and it means to execute an expression or algorithm and assess the results.

Another approach is to realize that the tournament involves four rounds, so if Noro lost during exactly one round then there are four different times this loss could have occurred: during round one, during round two, during round three, or during round four. Only these four resulting ranks need to be determined. This process is called abstraction and it means to focus in on only the relevant information and ignore the rest.

The decision tree used in this task can also be thought of as a binary tree. Every branch can be associated with a 0 (for “win”) or a 1 (for “lose”). A path from the root of the tree to a leaf would then be a combination of zeroes and ones which represent a binary number. All 4-digit binary numbers with exactly one 1 (meaning four rounds with exactly one “lose”) are 1000, 0100, 0010, and 0001. These binary numbers correspond with the values 8, 4, 2, and 1.

Recall that the answers to this task are 9, 5, 3, and 2. Can you see the connection between the binary numbers and task answers, and can you reason why they differ slightly? Think about what a rank of 1st place would look like in binary.

This process of reframing the task into a more familiar concept (binary numbers in this instance) is called generalisation.

Triangle

June took blocks of letters and stacked them in a triangle, as shown below.



She can find different paths from top to bottom through adjacent blocks in a triangle. Adjacent triangles are indicated by the arrows. She tries to count all the times she takes different paths when reading the same word.

Question

Exactly how many different ways can June read "DONKEY"?

Fill in the number and click "Save" when you are done.

Answer:

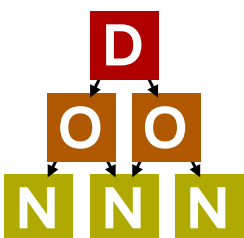
Save

Explanation

June can read "DONKEY" 32 different ways.

To calculate this, you have to take into account the number of paths from each block of letters (as you can see in the picture, there are always two possibilities).

For example, there are 2 different ways to read "DO" starting from the top block. Now, to find different paths to read "DON", there are always only 2 adjacent blocks that can be selected in the 3rd row from the last block of the path that reads "DO". Therefore, there are $2 \times 2 = 4$ different paths to read "DON" starting from the top block.



In addition, you have to take into account the number of possible paths you have to take to choose the next character (there are 5 characters: O, N, K, E and Y).

Therefore, the number of different ways June can read the word "DONKEY" will be 32 ($2 \times 2 \times 2 \times 2 \times 2 = 2^5 = 32$).

Background information

In computer science, recursion is a method of solving a computational problem where the solution depends on solutions to smaller instances of the same problem.

Counting the number of different paths in a triangular structure requires an effort to recognize some rules in a recursive way in simpler and smaller cases.

The task is about reasoning in a structure and finding a proper algorithm to find all possible paths in the triangle.





algorithm - developing a step-by-step solution to the problem, or the rules to follow to solve the problem (that is what has to be done in order to solve this problem)

The solver has to find a rule how number of possible ways increases during the reading process. Another words, solvers have to find out that each step of the algorithm always means exactly twice number of possibilities. (thanks for this point of view to Jiri Vanisek)

Open It

On each move, the spy can rotate the arrow clockwise for either 3 or 4 steps. The arrow toggles the light of the number it lands on. If the number's light was off, the light is turned on. If the number's light was on, the light is turned off.

For example, this is what happens if the spy makes 3 moves, each rotating by 4 steps:

Starting position	After 1st move (4 steps)	After 2nd move (4 steps)	After 3rd move (4 steps)
			

Question

To open the safe, the spy needs to light up only 7 and 8 (no other numbers should be lit).

What is the minimum number of moves the spy needs to do in order to light up only 7 and 8 from the starting position shown above?

☐ 3 moves

☐ 4 moves

☐ 5 moves

☐ 6 moves

☐ 7 moves





Explanation

The correct answer is **4 moves**.

One way to start solving this task is by realizing that the numbers 8 and 7 can both be reached by making two moves from the starting position. To reach 7, we move it twice by 3 steps. To reach 8, we move it once by 4 steps, then by 3.

Furthermore, if we are on 8, we can reach 7 in two moves, but not the other way round: reaching 8 from 7 would require more moves and complicate the solution. This is an indication that first trying to reach 8 first and then 7 looks like a promising option, which could maybe be completed in just 4 moves. But of course, with 4 moves, we also modify the state of the intermediary numbers of which we land. So, in order to keep only 7 and 8 lit and no other number, we must ensure that we land on the same intermediary number both when first go for 8, and then when we later reach 7.

This works if we move like this: first, 3 steps (we light up 4), then 4 steps (we light up 8), then 4 steps again (we turn off 4), and finally 3 more steps to light up 7.

1st move (3 steps)	2nd move (4 steps)	3rd move (4 steps)	4th move (3 steps)
			

Here is a systematic list of all possible sequences of moves, which can confirm that 4 is the optimal solution.

If the spy only makes 1 move:

1st move	Numbers lit
3 steps	4
4 steps	5

Using 1 move, the spy can light up only one number.

If the spy makes 2 moves:

1st move	2nd move	Numbers landed on	Numbers lit at the end
3 steps	3 steps	4, 7	4, 7
3 steps	4 steps	4, 8	4, 8
4 steps	3 steps	5, 8	5, 8
4 steps	4 steps	5, 1	5, 1

Using 2 moves, the spy can light up two numbers, but those 2 numbers are not 7 and 8.

If the spy makes 3 moves:

1st move	2nd move	3rd move	Numbers landed on	Numbers lit at the end
3 steps	3 steps	3 steps	4, 7, 2	4, 7, 2
3 steps	3 steps	4 steps	4, 7, 3	4, 7, 3
3 steps	4 steps	3 steps	4, 8, 3	4, 8, 3
3 steps	4 steps	4 steps	4, 8, 4	8
4 steps	3 steps	3 steps	5, 8, 3	5, 8, 3
4 steps	3 steps	4 steps	5, 8, 4	5, 8, 4
4 steps	4 steps	3 steps	5, 1, 4	5, 1, 4
4 steps	4 steps	4 steps	5, 1, 5	1

Using 3 moves, the spy can only light up either one or three numbers. It is impossible to light up 7 and 8.

If the spy makes 4 moves:

1st move	2nd move	3rd move	4th move	Numbers landed on	Numbers lit at the end
3 steps	3 steps	3 steps	3 steps	4, 7, 2, 5	4, 7, 2, 5
3 steps	3 steps	3 steps	4 steps	4, 7, 2, 6	4, 7, 2, 6
3 steps	3 steps	4 steps	3 steps	4, 7, 3, 6	4, 7, 3, 6
3 steps	3 steps	4 steps	4 steps	4, 7, 3, 7	4, 3
3 steps	4 steps	3 steps	3 steps	4, 8, 3, 6	4, 8, 3, 6
3 steps	4 steps	3 steps	4 steps	4, 8, 3, 7	4, 8, 3, 7
3 steps	4 steps	4 steps	3 steps	4, 8, 4, 7	8, 7
3 steps	4 steps	4 steps	4 steps	4, 8, 4, 8	-
4 steps	3 steps	3 steps	3 steps	5, 8, 3, 6	5, 8, 3, 6
4 steps	3 steps	3 steps	4 steps	5, 8, 3, 7	5, 8, 3, 7
4 steps	3 steps	4 steps	3 steps	5, 8, 4, 7	5, 8, 4, 7
4 steps	3 steps	4 steps	4 steps	5, 8, 4, 8	5, 4
4 steps	4 steps	3 steps	3 steps	5, 1, 4, 7	5, 1, 4, 7
4 steps	4 steps	3 steps	4 steps	5, 1, 4, 8	5, 1, 4, 8
4 steps	4 steps	4 steps	3 steps	5, 1, 5, 8	1, 8
4 steps	4 steps	4 steps	4 steps	5, 1, 5, 1	-

Therefore, the minimum number of moves the spy needs to do in order to light up only 7 and 8, from the starting position, is 4 moves.

Background information

In this task, we are given rules (instructions) that need to be repeated multiple times. However, we don't know how many times each rule (instruction) needs to be used to reach the goal.

One approach is to write a program that generates all possible sequences of moves (a total of 4 in this task) and checks the output for each case. Computer scientists refer to this as a *brute-force search*. However, take note that this method becomes impractical when we don't know the number of repetitions or when the number of possibilities overwhelms our memory.

In this particular case, it is similar to the "trial and error" method, where we attempt to find the solution by iteratively following the rules. It's important to note that the first solution found through trial and error doesn't always yield the optimal solution, unless we start from the most optimal implementation of the instruction, in this case starting from using only 1 move and then increasing the number of moves from there.

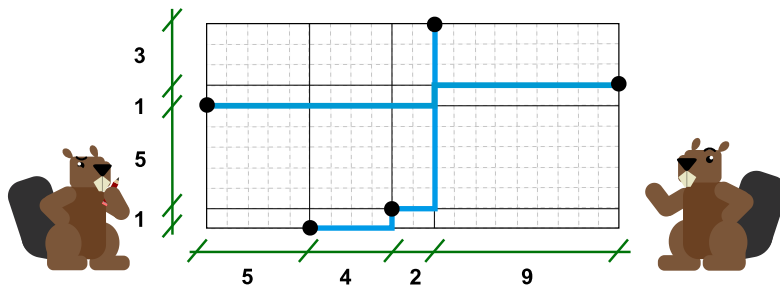
In informatics, number systems play a crucial role in representing and manipulating data. The circular arrangement of numbers in the task, similar to a clock, represents a specific number system. Computers use different number systems, such as binary (base-2), decimal (base-10), or hexadecimal (base-16), to represent and process information. The numbers in the task are like the digits on a clock, but instead of going from 1 to 12, they go from 1 to 8.

Understanding number systems and their representations is fundamental in computer science and helps computers store, process, and transmit data efficiently.

Channel plan

Maurice, an architect, presents his plan for the construction of a water channel to connect the five dwellings of a village. The dwellings are indicated on the map with black circles.

Maurice finds it nice to follow the horizontal and vertical lines that cross each of the dwellings. On his 20x10 drawing he indicates the layout of the planned channel in blue, and reports the distances of channel sections on the axes.



Jasmin notes that this channel is a total of 36 units long... however, to work less, it could be made shorter!

Question

How long is the shortest channel made up of horizontal and vertical segments that can connect the five dwellings?

Fill in the number and click "Save" when done.

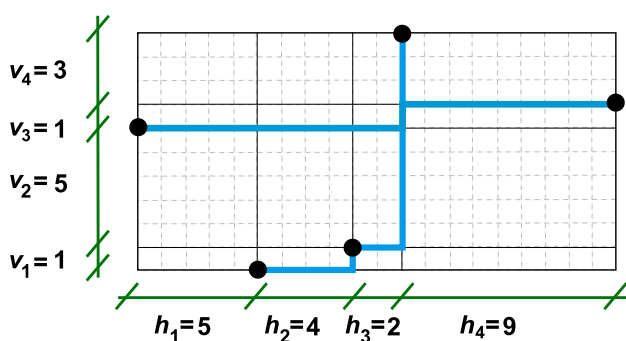
Answer:

Save

Explanation

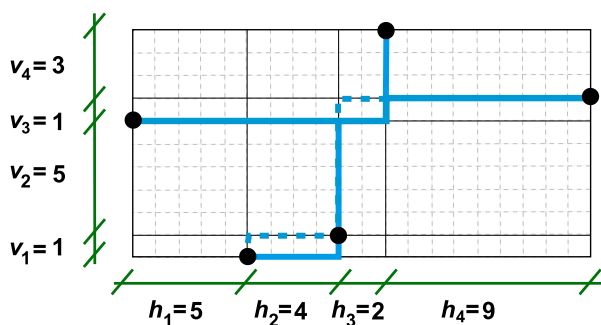
The correct answer is 34.

Let's consider the lengths of the horizontal ($h_1 = 5$, $h_2 = 4$, $h_3 = 2$, $h_4 = 9$) and vertical ($v_1 = 1$, $v_2 = 5$, $v_3 = 1$, $v_4 = 3$) sections, each of which must be "covered" by at least one channel stretch; therefore, the minimum length cannot be less than 30. (Note that this goal could be achieved if the dwellings were arranged on at most one horizontal and one vertical line.)

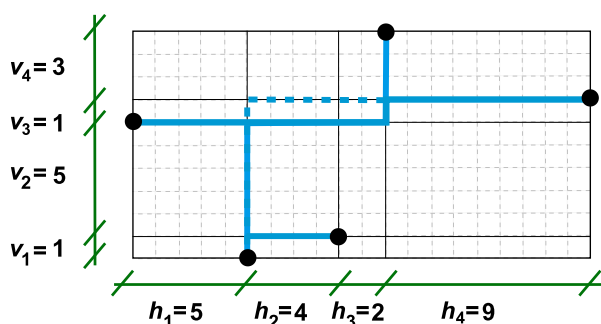


In Maurice's project (see figure above), sections h_2 and h_3 are covered twice, and indeed the overall length of the channel is $30 + 4 + 2 = 36$.

To get an optimal solution just move the channel stretch covering section v_2 as shown in the next figure (where dotted segments show equivalent alternatives to run across two sides of a rectangle), so that only section h_2 is covered twice, and the total length is $30 + 4 = 34$.



Moving the channel stretch covering section v_2 further to the left, two other optimal solutions are obtained:



Clearly, covering section v_2 twice is not convenient, since $v_2 > h_2$. The remaining possible locations for the only (vertical) stretch covering section v_2 are at the left and right ends of the map; in these cases, at least section h_1 or section h_4 (both greater than h_2) would be covered twice, respectively. This shows that the minimum overall length is 34, with h_2 covered twice.

Background information

Let's see how this task fits into computational geometry, a branch of computer science devoted to the study of algorithms which can be stated in terms of geometry; indeed, it is one of the oldest fields of computing.

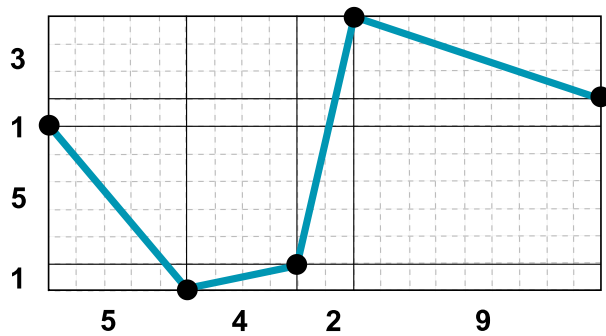
Let's start with this aim: we want to connect some given points of the plane to each other, by means of a tree, a structure composed of several segments (the edges of the tree), so that between any two of the given points there is only one path joining them. Of course, the segments' end points (the vertices of the tree) include the given points, and possibly other points, called Steiner points (Jakob Steiner was a Swiss mathematician who lived in the nineteenth century).

The rectilinear Steiner tree problem is to find a tree of minimum total length composed solely of horizontal and vertical line segments with respect to a predetermined Cartesian reference system. This problem arises in wire routing, the planning of connections between the thousands of components of a VLSI system; indeed, in the design of electronic (VLSI) circuits, wires are often constrained to run only in vertical and horizontal directions. The problem of determining the minimum total wire length is difficult, precisely NP-complete (Garey and Johnson, 1977).

On this topic, a lot of papers and entire books have been written, as well as efficiently solvable special cases and heuristic or approximate algorithms giving near-optimal solutions have been developed.

In 1966, Maurice Hanan proved that there exists an optimal rectilinear Steiner tree with all Steiner points chosen from the intersection points of horizontal and vertical lines drawn from the given points, precisely the case proposed in this task; of course, the resulting grid, called Hanan grid, depends on the chosen Cartesian reference system.

A remark, to conclude: if the beavers had simply wanted to connect the dwellings of their village, without introducing other intersection points, they would have had to find a minimal spanning tree; in the present case, it would result as shown in the next figure, with a total length of about 30.64 units, and therefore with a saving of about 10%.



To solve the minimal spanning tree problem in general, several efficient algorithms are known (e.g., Kruskal and Prim-Jarník).

This task requires the abilities to configure a scenario in different ways, respecting the constraints imposed and choosing objects from a given set to achieve a certain goal, and to see how far the search for a solution can go. It stimulates looking at a problem under different constraints and using heuristic reasoning to discover a solution, compare different solutions, and understand how close a solution can be to an optimal one.

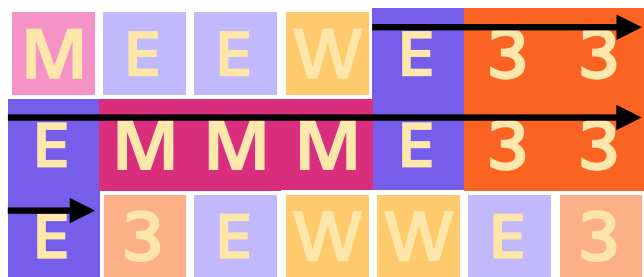
This task can be useful to illustrate how a change in requirements (from channels directly connecting dwellings, as in spanning tree problem, to channels formed by orthogonal stretches) can make the optimization problem (much) more difficult to solve. Provided the number of dwellings is small, you can proceed as suggested in the Answer Explanation section, in order to find a solution and try to improve it, with different combinations. You can then compare your solution with the one provided by a software package such as *GeoSteiner*.

Palindrome Passwords

Hannah and Otto are planning a meeting for their club. The secret meeting time is encoded within a code block of letters and numbers. The code block is read in rows starting from the top, each row from left to right, as if they formed a single sequence.

The secret meeting time is the length of the longest palindrome within that code block. A palindrome is a sequence of letters and numbers that reads exactly the same forwards and backwards.

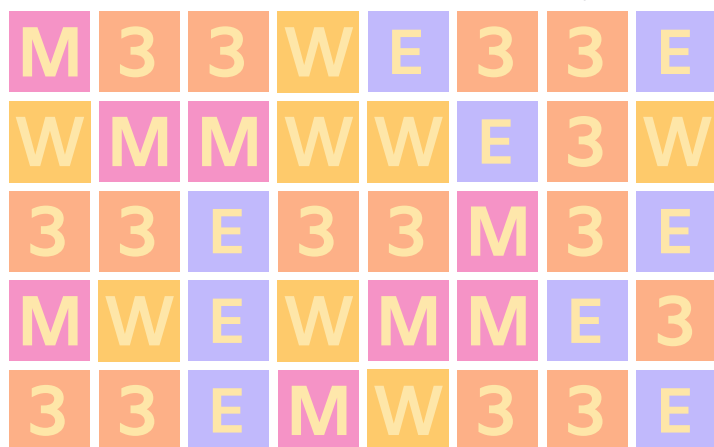
For example, the code block from last week is shown below. The longest palindrome is "E33EMMME33E", which has a length of 11. Therefore, the secret meeting time was 11:00.



Question

Here is the code block for next week's meeting. Which letters and numbers make up the palindrome that indicates the secret meeting time?

Click on the blocks to select them and click "Save" when you are done.



Save

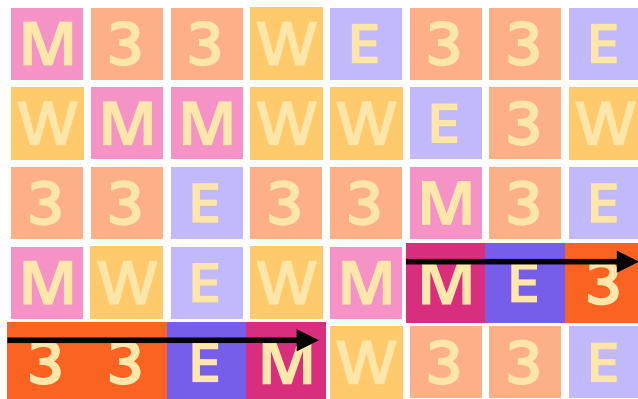
Erase

Explanation

The longest palindrome is 7 blocks long. Therefore the secret meeting time is 7:00.



There are quite a few palindromes in this week's image, but **M E 3 3 3 E M** is the longest. It contains the longest set of blocks in a row, which are the same when read both forwards and backwards.



Background information

Making and sharing codes so that only certain people can read them is a type of **encryption**.

Encryption is used in computer science to protect information, so that it cannot be accessed and understood by everyone who sees it. When people try to read encrypted data without the key to understanding it, it can just look like a scrambled mess of information.

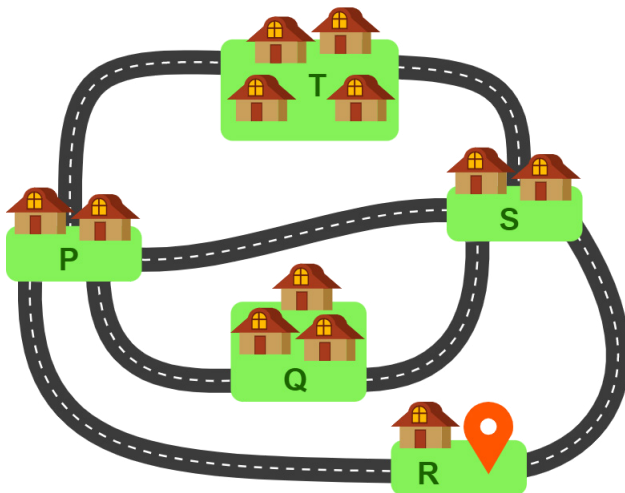
For example, when we buy something online, an algorithm is used to encrypt our payment information. This helps to protect sensitive information from hackers. Encryption is similar to putting a lock on a diary, so that only you and the people you trust can read it.

This question helps us to develop and use search algorithms. We can simply go through all possible paths to check which ones are palindromes, but this takes a very long time. We can instead develop strategies, such as starting with all the 2 and 3 length palindromes, and growing them by adding start and end positions. This helps us find a solution much more quickly.

In looking for palindromes, this question also makes us use our pattern recognition skills. Palindromes always follow a pattern, and once we know what the features of this pattern are, it is easier for us to seek them out.

Mapping the roads

A company that digitises maps is collecting pictures of every road that connects the villages displayed on the map below. Due to time constraints, they only have seven hours to capture these images. They can choose any route that enables them to accomplish this task, bearing in mind that it takes exactly one hour to capture all necessary pictures of a road.



Question

Starting from village R, how many different routes could the car use, to complete the map pictures in exactly 7 hours?

6

8

10

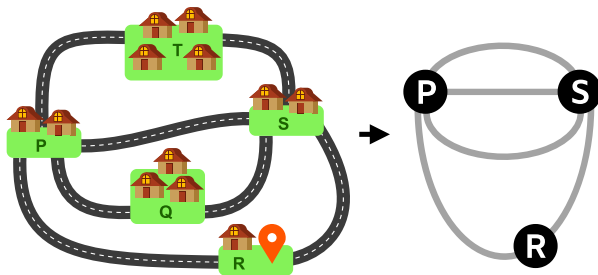
12

Explanation

The correct answer is: there are 12 different routes.

The car must cross each path exactly once in order to pick up all the pictures in seven hours.

As a result, the only places where there is a choice of which path to take are the starting house (R) and the houses (P and S) connected to more than two paths. We can simplify the problem statement by building an abstract model that ignores houses Q and T. Below is such an abstract model of the paths



We are told we have to start from R, and because we need to hop along each path exactly once, this means we must finish at R also.

Let's first consider moving rightwards from R. The only choices of routes are how many different ways to get from S to P that visit each of the three paths between S and P. The answer to this is six.

To understand this, let's give names to the three paths: top, middle, bottom. We can start with any one of the three, but after choosing one we only have a choice of either of the two others, and after making that choice there is no choice for the third path. So the number of choices is $3 \times 2 \times 1 = 6$, as shown here:

top, middle, bottom	middle, top, bottom	bottom, top, middle
top, bottom, middle	middle, bottom, top	bottom, middle, top.

So, starting from R and moving rightwards, there are six ways to get back to R by travelling along each path exactly once. What about starting from R and moving leftwards? The answer is also six, and in fact, each of the possible routes is just the reverse of the first six routes.

This gives a total of 12 possible routes that visit each of the seven edges exactly once. Listing them all out, here are the 12 different routes the Easter Bunny could use:

$R \rightarrow S \rightarrow T \rightarrow P \rightarrow S \rightarrow Q \rightarrow P \rightarrow R$ $R \rightarrow P \rightarrow Q \rightarrow S \rightarrow P \rightarrow T \rightarrow S \rightarrow R$
 $R \rightarrow S \rightarrow T \rightarrow P \rightarrow Q \rightarrow S \rightarrow P \rightarrow R$ $R \rightarrow P \rightarrow S \rightarrow Q \rightarrow P \rightarrow T \rightarrow S \rightarrow R$
 $R \rightarrow S \rightarrow P \rightarrow T \rightarrow S \rightarrow Q \rightarrow P \rightarrow R$ $R \rightarrow P \rightarrow Q \rightarrow S \rightarrow T \rightarrow P \rightarrow S \rightarrow R$
 $R \rightarrow S \rightarrow P \rightarrow Q \rightarrow S \rightarrow T \rightarrow P \rightarrow R$ $R \rightarrow P \rightarrow T \rightarrow S \rightarrow Q \rightarrow P \rightarrow S \rightarrow R$
 $R \rightarrow S \rightarrow Q \rightarrow P \rightarrow S \rightarrow T \rightarrow P \rightarrow R$ $R \rightarrow P \rightarrow T \rightarrow S \rightarrow P \rightarrow Q \rightarrow S \rightarrow R$
 $R \rightarrow S \rightarrow Q \rightarrow P \rightarrow T \rightarrow S \rightarrow P \rightarrow R$ $R \rightarrow P \rightarrow S \rightarrow T \rightarrow P \rightarrow Q \rightarrow S \rightarrow R$

Background information

A graph consists of a set of vertices or nodes (usually depicted as points or small circles) and a set of edges. Each edge joins two vertices. A sequence of connected edges in a graph is called a path. In this problem, the houses are the vertices, and the roads that connect the villages are the edges.

The graph in this task is special, since it is connected (there is a path from any vertex to any other vertex by following edges of the graph) and no vertices have an odd number of edges connected to them. For graphs that have these two special properties, there is a way to follow all the edges in the graph exactly once, and return to the starting vertex, which is called an Eulerian circuit. Eulerian circuits were first discussed by Leonhard Euler while solving the famous Seven Bridges of Königsberg problem in 1736. We can find Eulerian circuits by applying Fleury's algorithm or Hierholzer's algorithm. A delivery driver might like their route to an Eulerian circuit so they don't have to retrace their previous steps.

Algorithms: This task requires one to write (or invent) an algorithm. In this task we are given a graph and must systematically find all of the ways to traverse through each edge exactly once, keeping track of all of them in order to obtain an accurate count.

Abstraction: This task illustrates creating an abstract model. In order to more efficiently find the number of different routes one must recognise that a simplified graph that ignores vertices T and Q is sufficient.

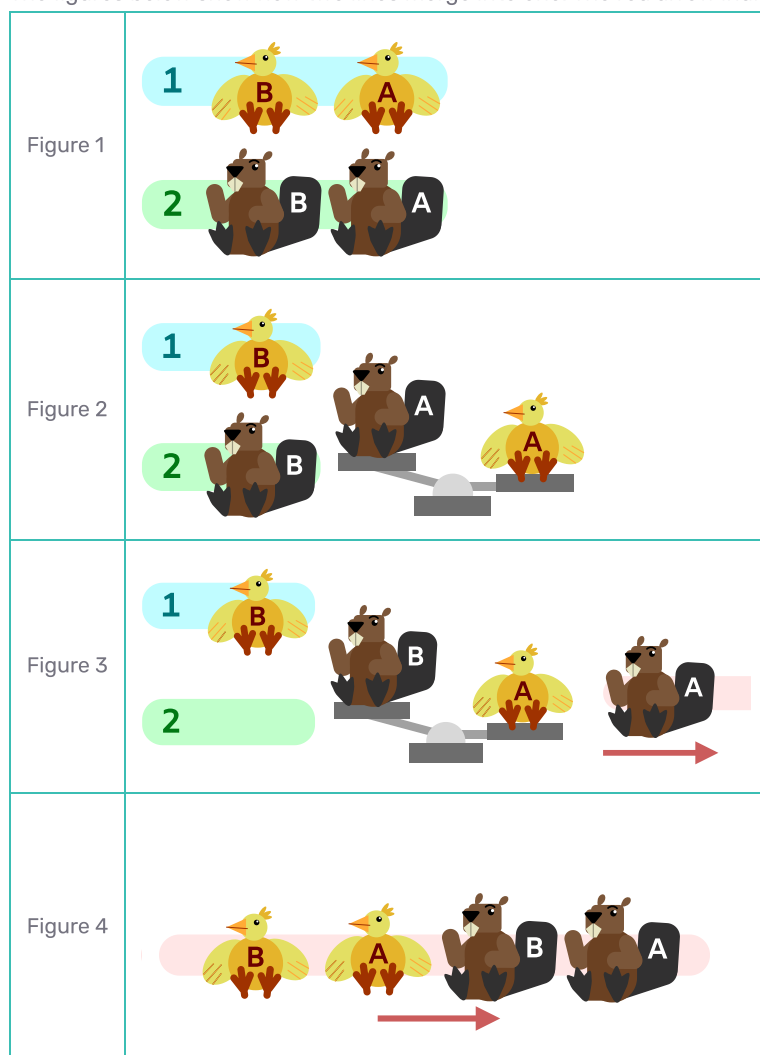
Decomposition: This task illustrates breaking a problem into parts. We can separate the two problems of going left or right from house R, and then solve independently the problem of going between S and P. Putting our partial solutions together gives us the final answer.

Line Up

The Riverland Wrestling Competition is today. All animals from all teams participate in order based on their weight. To line them all up correctly, first, animals from each team line up based on their weight, the lightest on the right and the heaviest on the left. Then, these lines are merged into one, two lines at a time, following the steps below, until all lines are merged:

1. The first animal in each of the two lines steps on the scales.
2. The lighter one of the two queues up at the end of the new line.
3. The heavier animal stays on the scales while a new animal from the opposing team steps onto the other side of the scales.
4. The animals keep repeating Step 2 and 3 until all animals are merged into one new line.
5. When only animals of the same team are left to be weighed, they simply join the new line in the order that they are in.

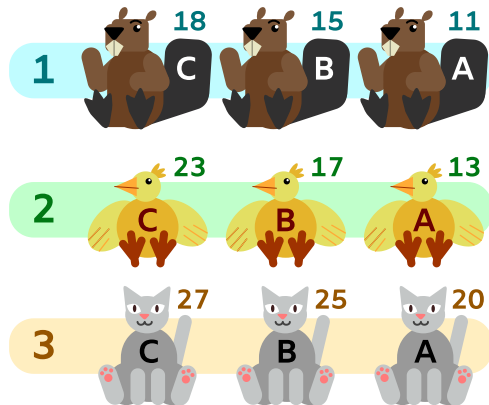
The figures below show how two lines merge into one. The red arrow indicates the direction of the front of the line.



It took two weight comparisons to merge these two lines (figure 2 and 3) into one line (figure 4).

Question

Three teams are lined up, ready to be weighed, as shown below. The numbers above the animals indicate their weight. Which two teams will take the highest number of weight comparisons to merge?



team 1 and team 2

team 2 and team 3

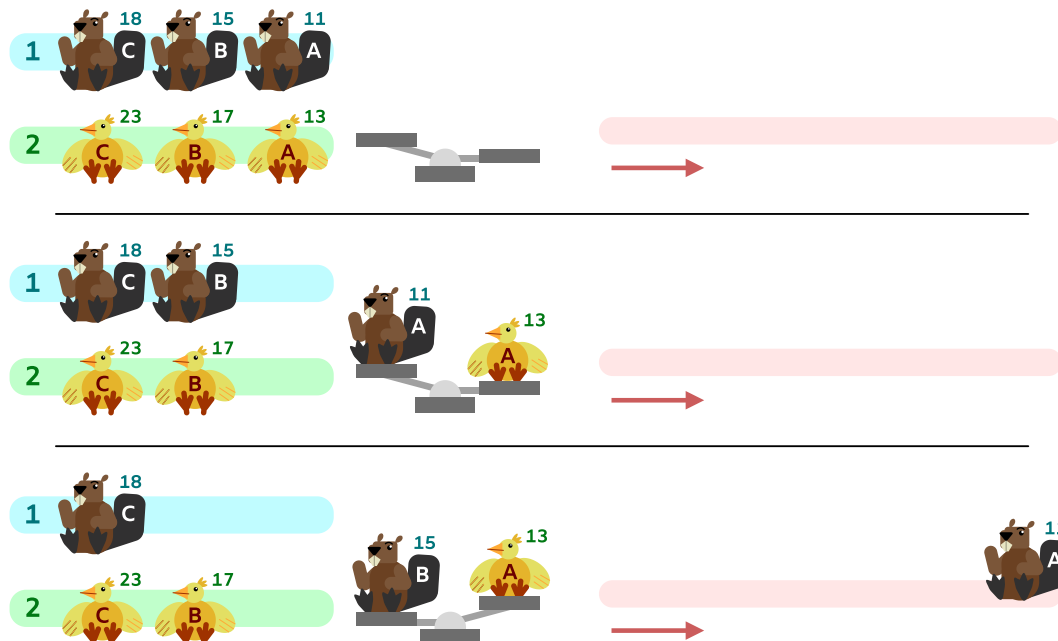
team 1 and team 3

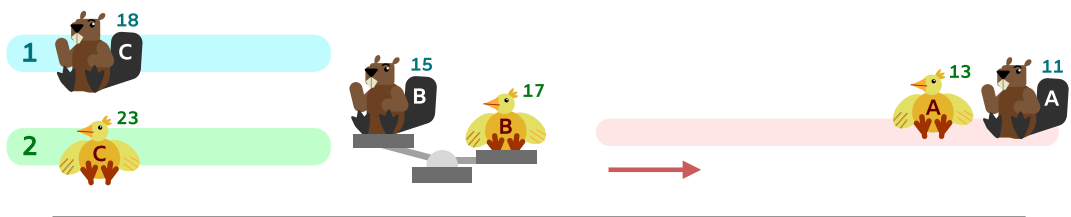
they all take the same number of weight comparisons to merge

Explanation

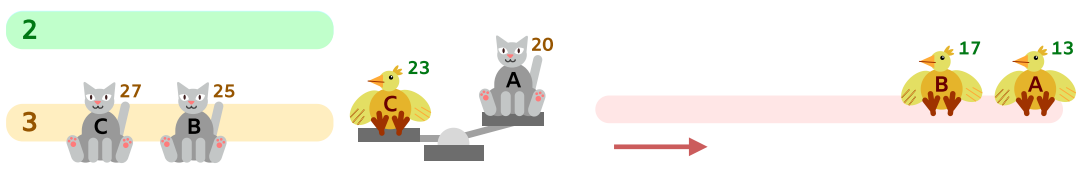
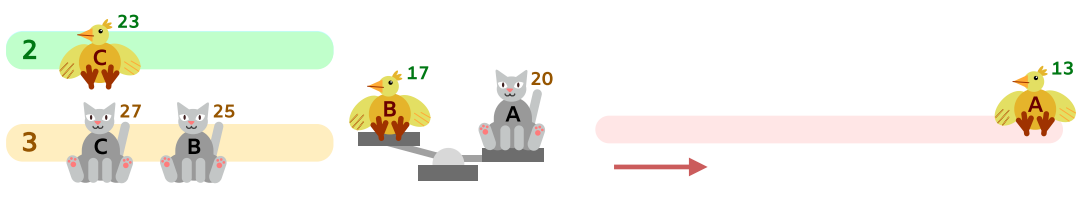
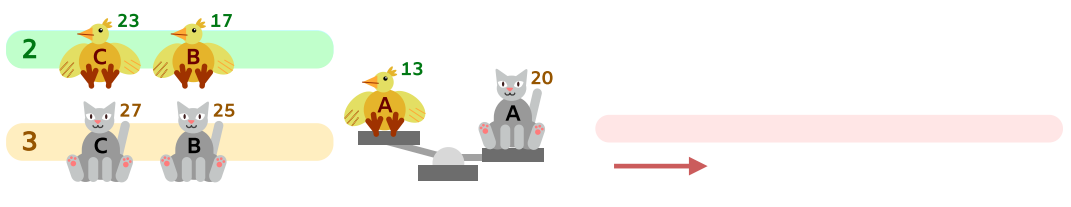
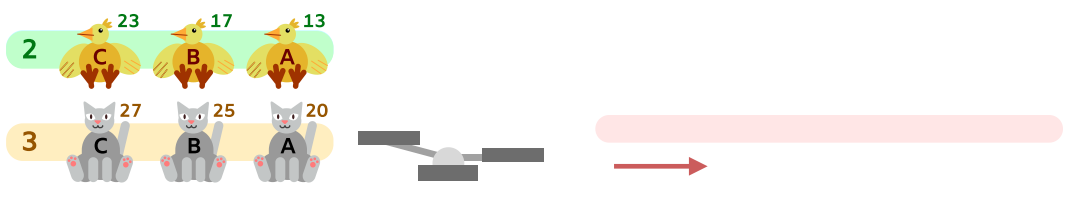
The correct answer is A.

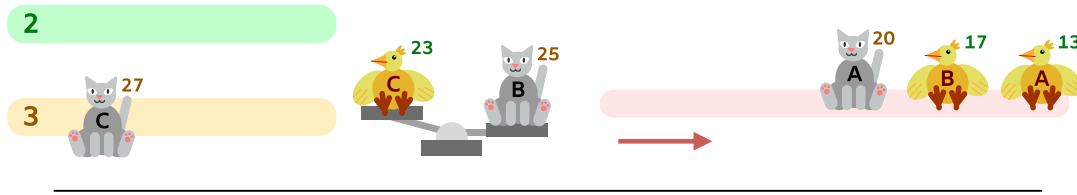
It takes five comparisons to merge team 1 and team 2 as shown below:



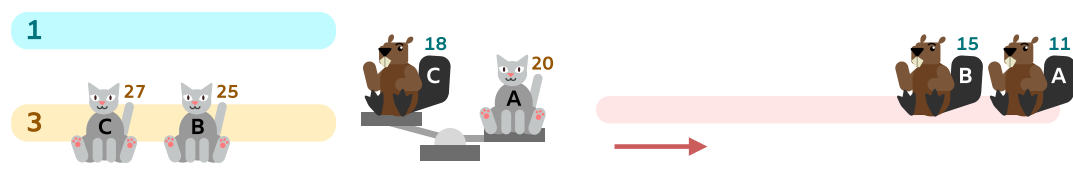
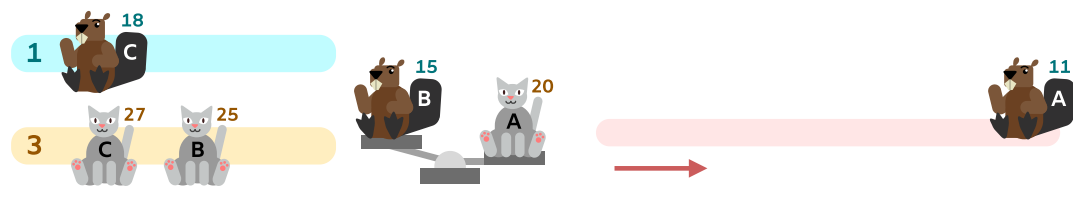
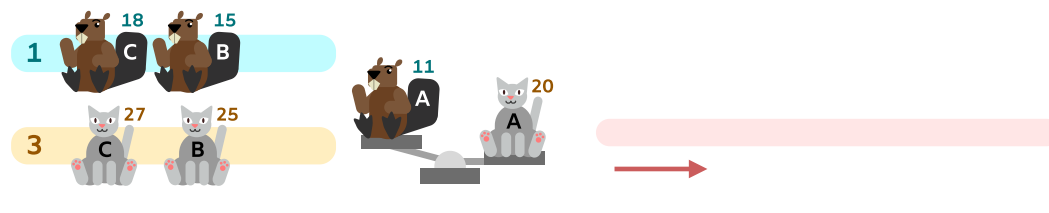
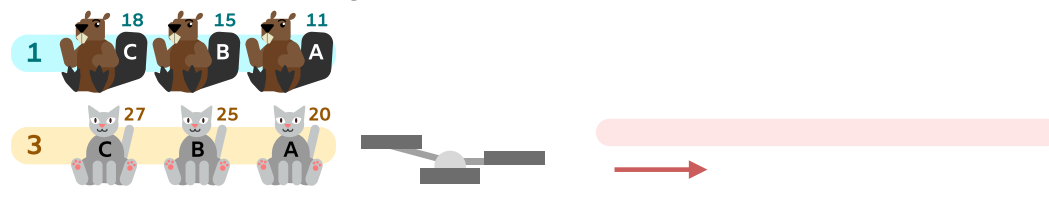


It takes three comparisons to merge team 2 and team 3 as shown below:





It takes four comparisons to merge team 1 and team 3 as shown below:



Actually it is not necessary to execute the merge sorting algorithm on the three pairs of teams; it suffices to observe that:

1. the lowest number of comparisons are needed when the heaviest animal in one line is lighter than the lightest animal in the other line;
2. the highest number of comparisons are needed when each animal in one line is heavier than the animal in the same position in the other line, but lighter than the next animal in the other line; that is, for each animal, the next higher weight animal is in the other line.

Team 1 and Team 3 have property 1. Team 1 and Team 2 have property 2, and are therefore the ones that take the highest number of comparisons.

Background information

Computer scientists use sorting algorithms to put a group of elements into order. One of the sorting algorithms is **merge sort**, which is the algorithm used in this task to sort all animals participating in the contest. This algorithm is based on a technique called **Divide and Conquer** by computer scientists.

Divide and Conquer breaks down a problem (like sorting all animals) into smaller sub-problems (like sorting subgroups of animals), solves each sub-problem, and then combines the solutions to solve the original problem. When applicable, this technique can boost the efficiency of algorithms.

To solve this task, students need to use **algorithmic thinking** to understand how two lines are merged and apply the algorithm to merge teams.

When students try to calculate the number of comparisons of merging any two lines, they can use **abstraction** to figure out what are the factors that affect the total number of comparisons: in this case if the weights increase alternating in the two lines or the weights in one line are all smaller than the weights in the other line.

Logs to the warehouse

Amy and Bob have to carry logs – one at a time! – from the river bank to the warehouse on the left. To make their job more enjoyable, they decide to create a game and take turns at work.

Whoever is on duty selects a log and carries it to the left, until they comes across another log or reach the warehouse.

Whoever is forced to carry the last remaining log to the warehouse loses the game.



A winning strategy is one that can be played and always results in a win, no matter how the opponent plays.

Today they have to carry four logs, arranged as shown, and Amy is first on duty.

Question

Only one of the following statements is true: which one?

Amy has a winning strategy and must start by carrying one of the two logs from A to the warehouse.

Amy has a winning strategy and must start by carrying the log in B next to the two logs in A.

Amy has a winning strategy and must start by carrying the log in C next to the log in B.

None from the Amy's strategies from the options 1, 2, 3 are winning.

Explanation

The correct answer is: Amy has a winning strategy and must start by carrying the log in B next to the two logs in A.

Let's call "position" the sequence of numbers representing, from left to right, how many logs are in each group outside the warehouse, from nearest to farthest from the warehouse. Play moves from one position to another, and it is not possible to go back to a previous position. In our case, the initial position is represented by the sequence 211 (for simplicity, we can omit the use of a separator between one number and the next, since there are only four logs here).

If Amy carries the log in B next to the two logs in A, then the new position is 31, and the game can continue in two ways:

a) Bob carries one log from A to the warehouse (so position 21 is reached), Amy carries the log in C next to the two logs in A (so position 3 is reached), Bob has no choice but to carry one of the three logs to the warehouse, Amy carries one of the remaining two, and the last log is up to Bob!

b) Bob carries the log in C next to the three logs in A (so position 4 is reached), Amy carries one log to the warehouse (so position 3 is reached), and the continuation is as above.

The correctness of this answer (2) implies that the last answer (4) is wrong.

If Amy starts by carrying one of the two logs from A to the warehouse (answer 1), then the new position is 111, from which Bob carries the log in C next to the log in B reaching position 12 (this is the only right move Bob can make to win, as we'll see); now the game can continue in two ways:

- Amy carries one log in B next to the log in A reaching position 21, the game continues as in case a) but with the two players' roles reversed, so Amy loses.

- Amy carries the log in A to the warehouse reaching position 2, and the continuation is forced.

If Amy starts by carrying the log in C next to the log in B (answer 3), then the new position is 22, from which Bob carries one log from A to the warehouse reaching position 12, as above.

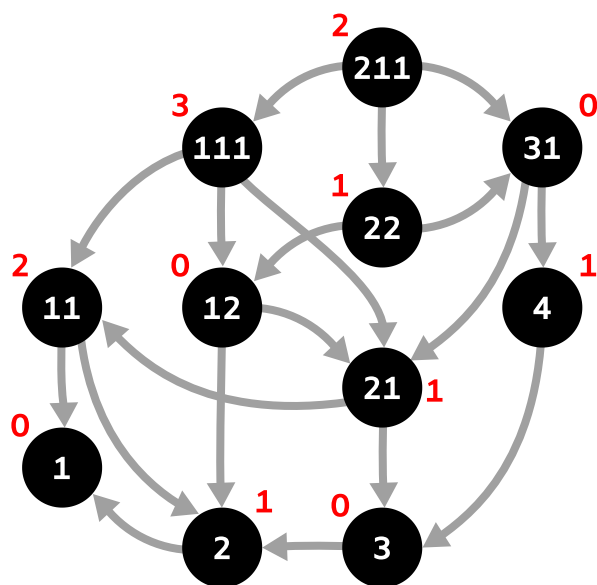
Background information

This task is an instance of an impartial combinatorial game (since the two players have the same set of legal moves from any given position); moreover, there will always be a winner, since a final draw is impossible here.

To represent the transitions from each position to another, and therefore get an overview of all the possible developments of a game starting from a given initial position, we build a graph, where each node stands for a position. From each node X we draw an arc towards each node Y representing a position reachable from X by one move (and in this case we say that Y is adjacent to X). So this graph is a directed graph, and it is also acyclic for our game since a player cannot go back to a previous position.

In our case, it is suitable to assume 1 as the final position: whoever, by his last move, leaves this position to the opponent will win the game, since the only action that the opponent can perform is to carry the only remaining log to the warehouse.

By way of example, let's build the graph starting from the initial position proposed in this task.



Let's show how the labels (in red) are (uniquely) assigned to each node, i.e., to each position. Final node 1 is labeled 0; it is the only adjacent node to node 2, and then we assign label 1 to node 2. In general, node X is labeled with the smallest natural number that is not the label of any node adjacent to X. In our case, therefore, we assign label 2 to node 11 (its adjacent nodes are labeled 0 and 1), 0 to node 3 (the only node adjacent to it is labeled 1), 1 to node 21 (its adjacent nodes are labeled 0 and 2), and so on, until reaching the initial node 211, which will have label 2.

The labels obtained in this way are the values of the Sprague-Grundy function associated with the respective nodes of the graph; the positions labeled 0 are precisely those in which whoever made the previous move is sure of being able to win. (Note that no node adjacent to a node labeled 0 can have label 0.)

However, if we simply want to identify the correct moves, we just assign a binary label to each node of the graph. The values of this labels are usually denoted by P (Previous) and N (Next); P-positions are winning for the “previous” player, who has just moved, N-positions are winning for the “next” player, who will have to move. Once final positions have been labeled P (in our case only one), label N is assigned to the nodes that have at least one adjacent node labeled P, whereas label P is assigned to the nodes that have all the adjacent nodes labeled N. Finally, nodes labeled P will be exactly those on which the Sprague-Grundy function has value 0 (and there will never be two adjacent nodes labeled P). Whoever has the winning strategy must move to a P-position every time.

It was the mathematician Charles L. Bouton who started the theoretical studies on this category of games, in 1901, with the analysis of a game that has become famous and paradigmatic, the NIM; fundamental and more general results were then obtained, independently, by Roland P. Sprague in 1935 and by Patrick M. Grundy in 1939. The Sprague-Grundy function contains a lot more information than just the P- and N-positions, which is useful for further game analysis.

This task involves abstraction skills (in representing game positions and how to pass from one of them to another), algorithmic thinking (to set the rules to follow in terms of “what if”, or nested if-then-else statements), evaluation (to check the correctness of the solution found) and, possibly, generalization (in wondering how to solve the game starting from an arbitrary position).

In dealing with tasks like this, we have to explore a range of possibilities at each of a series of choice points, starting from the given position. Of course, all the information about what choices have already been tested and which ones remain for further exploration must be kept (and this is precisely what happens in the execution stack by a recursive algorithm, when the process searches a branching structure in some systematic way). Often these challenges are too hard to completely solve in a short time, but the case of the task presented here is not overly challenging!

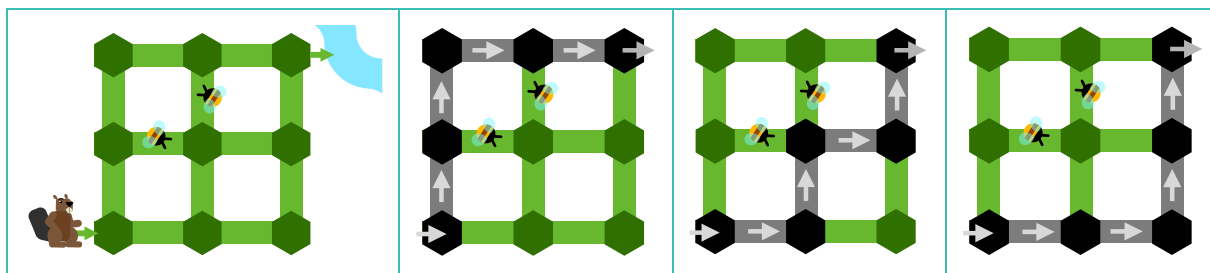
More in general, strategy games, as unplugged activities, represent an interesting interactive context for the development of computational thinking, encompassing several forms of reasoning and learning.

Walking in the Forest

Sam loves to walk in a forest. She always starts her walk on the **bottom left corner** on the map and goes to the **top right corner** where a beautiful river is located.

Moreover, she always **avoids bees**  and always **goes up or right** when walking. 

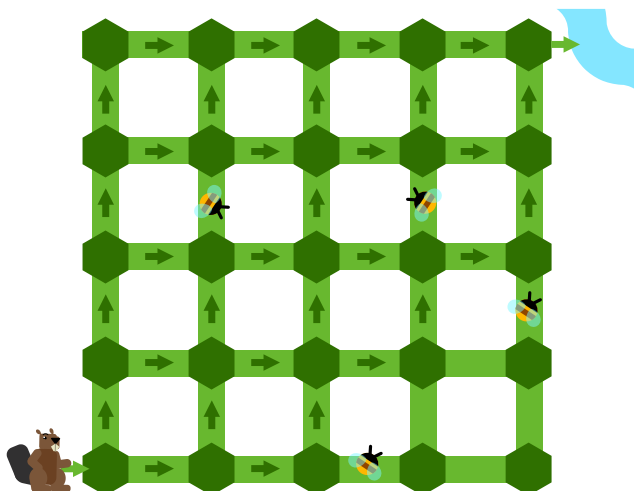
Sam is easily bored and always prefers to avoid walking the exact same path, so she quickly figured out that the forest has 3 different paths she could use:



Sam regularly visits her friend Noa who lives in a larger forest, and she wants to discover how many different paths exist to get the river there. Can you help her?

Question

How many different paths can Sam use in the following forest, assuming she always goes up or left and avoids bees? Enter the number and click "Save" when you are done.



Answer:

Save

Explanation

The correct answer is 32.

There are **32** different paths that avoid bees and always go up and left.

One could try to enumerate all possible paths, but this would take a lot of time and would be very prone to errors.

There is however a strategy that would allow us to much more easily compute the desired answer in a way that scales better with the size of the forest.

Let's start with the original smaller 3x3 forest. One could more easily represent it with a 3x3 grid, with each cell being an intersection and with the bees represented as red segments that indicate we cannot traverse them. In this grid we want to go from cell (1,1) to cell (3,3):

(1,3)	(2,3)	(3,3)
(1,2)	(2,2)	(3,2)
(1,1)	(2,1)	(3,1)

With this one could write on each cell the number of paths starting on that cell that can reach cell (3,3). At first you will enter a 1 (for 1 possible path) in the top right corner, and also a 1 in all cells at the top (1st row) and at the right (3rd column) of the grid. Not considering bees, then the number of possible paths from (x, y), represented as $nr_paths(x,y)$, is defined as $nr_paths(x,y) = nr_paths(x+1,y) + nr_paths(x,y+1)$, that is, the paths from a cell are all the paths starting on the upper cell plus the ones of the left cell (this works for all cells but the top row or the right column, that is why we initialized these first). If there is a bee separating us from a neighbor cell, we simply do not sum the corresponding number. We can now fill in the cells with the number of paths in a way that when we need a value, it is already there (e.g. fill from the top to the bottom and in each line from right to left). At the end our result is on cell (1,1), that shows the 3 existing paths:

1	1	1
1	1	1
3	2	1

To solve the asked 5x5 grid we can now simply fill the respective cells, with the final result being on cell (1,1):

1	1	1	1	1
5	4	3	2	1
9	4	4	1	1
18	9	5	1	0
32	14	5	1	0

Background information

To solve this problem we needed to think on a suitable algorithm, that is a sequence of steps that would allow to perform the desired computation. There are typically several possible algorithms for a given problem, that would take different amounts of time to be applied.

Trying to solve by enumerating all possible paths is a strategy known as exhaustive search (or brute search), which would not be feasible for larger forests, as the number of paths grows exponentially with the size of the forest.

The proposed algorithm here follows a technique known as dynamic programming. At its essence, we first divided the problem into smaller instances of the same problem (which is known as divide-and-conquer), that is, we characterized the solution of a the problem (the number on one cell) as being a composition (in this case a sum) of two smaller problems (the neighbor cells). Moreover, the number of cells is much smaller than the actual number of paths, since in a way we can reuse results and never have to recompute the number of a paths from a certain cell.

Dynamic programming is a very general strategy and can be applicable to many different classes of problems. Its gains in efficiency are precisely due to the fact that the results of sub-problems can be reused.

To solve this problem, several Computational Thinking competencies are required:

1. Decomposition: the problem can be broken down to smaller parts, which are in itself smaller instances of the same problem.
2. Pattern recognition and abstraction: how to compose the solution of a given cell from two other cells and then abstract that concept for all cells.
3. Algorithmic thinking: establishing a set of steps, such as the order in which to fill in the cells, to compute the desired answer.

Painting doors

Arlo lives in a high rise apartment building. The doors of the apartments are either red (x) or blue (+). See what the building looks like below.

5	1	x	2	+	3	x	4	x
4	1	+	2	x	3	+	4	x
3	1	x	2	x	3	x	4	+
2	1	+	2	+	3	+	4	x
1	1	x	2	x	3	x	4	x
0	1	x	2	+	3	+	4	x

The floors are labelled 0 to 5 and the doors on each floor are labelled 1 to 4.

A painter needs to paint the doors and uses the following procedure:

Paint(floor,door)

IF the apartment(floor,door) exists

THEN IF the door of the apartment(floor,door) is red

THEN change the colour of the door to yellow and

Paint(floor, door - 1) and then

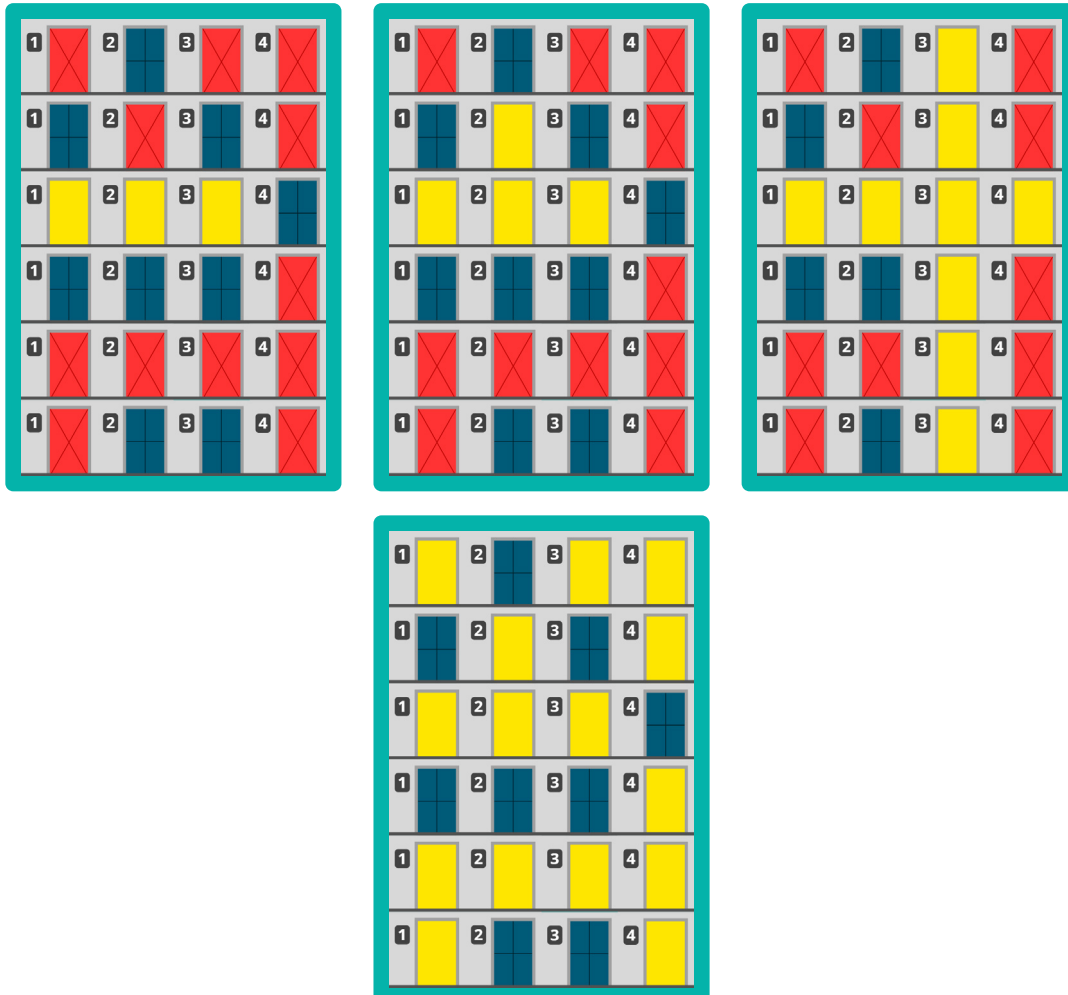
Paint(floor, door + 1) and then

Paint(floor - 1, door) and then

Paint(floor + 1, door)

Question

What does the building look like after the painter executes the procedure **Paint(3,3)**?



Explanation

Solving this question requires stepping through the instructions and following them step by step.

Paint(floor,door)

IF the apartment(floor,door) exists

THEN IF the door of the apartment(floor,door) is red

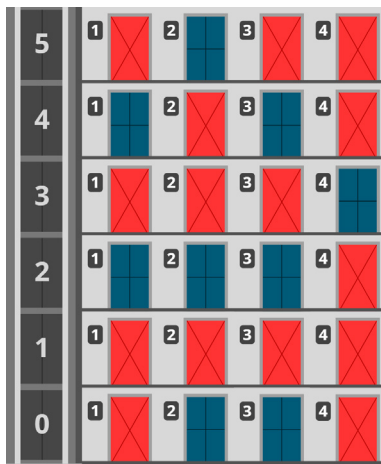
THEN change the colour of the door to yellow and

Paint(floor, door - 1) and then

Paint(floor, door + 1) and then

Paint(floor - 1, door) and then

Paint(floor + 1, door)



Paint(3,3) starts at floor 3, door 3.

This door does exist, and is red.

Therefore, we carry on with the instructions to change the colour of the door to yellow, and then apply **Paint()** to four more doors.

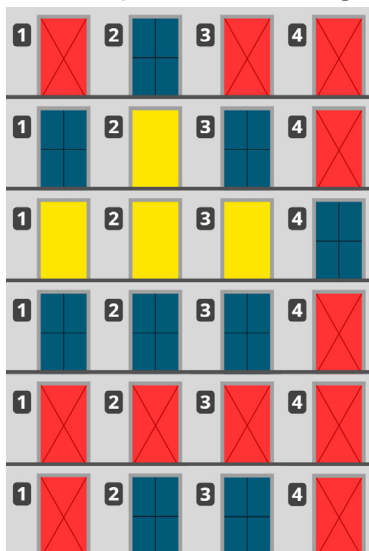
(floor, door - 1), (floor, door + 1), (floor - 1, door), and (floor + 1, door) correspond to the doors immediately adjacent to the current door.

Thus, we see that the overall purpose of this code is to progressively paint adjacent red doors yellow.

However, we can also see that **Paint()** is only applied to adjacent doors when the current door is red. Once a blue door is reached, the painting will not progress further from that blue door.

This means that only the red doors which can reach floor 3, door 3 through a path of adjacent red doors will be painted yellow.

This corresponds to the following doors being painted yellow:



Background information

This question involves the interpretation of *pseudocode* that implements an example of *recursion*.

Pseudocode is a way of describing the steps of an algorithm in natural language. It is often used to communicate an algorithm in an easy to understand manner without losing the structure of the logic.

Recursion is a method commonly used in solving computational problems, where a defined sequence of steps involves "calling itself" ie. repeating those sequence of steps for a different set of inputs and utilising the output in computing the overall output. This is useful when a larger problem requires the solutions to smaller instances of the same problem.

For the example of this question, painting an adjacent area of red doors starts at a single door which is painted yellow if it is red. Then moving on to its adjacent doors, the process of painting one door repeats until the entire adjacent area is painted. We say that the doors have been painted recursively.

2024 Round 2 All

ID	Question	Year Level	Decomposition	Pattern Recognition
2023-TR-02	Friendly Aliens	3/4 Easy	○	○
2023-IE-05	Coded Ages	3/4 Easy	○	
2023-UY-02	Carnival Masks	3/4 Easy	○	○
2023-UY-04	Hidden treasure	3/4 Easy	○	
2023-JP-05	Hamburger Shop	3/4 Easy, 5/6 Easy	○	
2023-JP-03a	Walking Logs	3/4 Medium, 5/6 Easy	○	
2023-NL-04	Mazes	3/4 Medium, 5/6 Easy	○	
2023-CN-03	Treasure Island	3/4 Medium, 5/6 Easy	○	
2023-IE-04	Travel by Coin	3/4 Medium	○	
2023-AZ-03	Encrypted Message	3/4 Medium, 5/6 Easy	○	
2023-UA-02	Favourite Drinks	3/4 Hard, 5/6 Medium, 7/8 Easy	○	
2023-AZ-04a	Maze game	3/4 Hard, 5/6 Medium, 7/8 Easy	○	
2023-PH-03	Sprinklers	3/4 Hard, 5/6 Medium, 7/8 Easy	○	○
2023-HU-03	Gift selection	3/4 Hard, 5/6 Medium, 7/8 Easy, 9/10 Easy	○	
2023-SK-06	Robot on the path	3/4 Hard, 5/6 Medium, 7/8 Medium, 9/10 Easy	○	
2023-PH-04b	Sealed Letters	5/6 Hard, 7/8 Medium, 9/10 Easy	○	
2023-VN-04	Watercolour	5/6 Hard, 7/8 Hard, 9/10 Medium, 11/12 Easy	○	
2023-VN-02	Snail Compress	5/6 Hard, 7/8 Medium, 9/10 Medium, 11/12 Easy	○	
2023-CA-03	Delivering Mail	5/6 Hard	○	
2023-PR-01	Jumping Together	5/6 Hard, 7/8 Hard, 9/10 Medium, 11/12 Easy	○	○
2023-HR-01	Frog and Mosquito	7/8 Easy	○	
2023-PT-02	BebrasGPT	7/8 Medium, 9/10 Easy, 11/12 Easy	○	
2023-BR-04	Stamp Machines	7/8 Medium, 9/10 Easy	○	○
2023-LT-03	Correcting error	7/8 Hard, 9/10 Medium, 11/12 Easy	○	
2023-SK-05	Bebras Ball	7/8 Hard, 9/10 Medium, 11/12 Medium	○	
2023-NL-01	Triangle	7/8 Hard, 9/10 Hard, 11/12 Medium	○	○
2023-HU-05	Open It	9/10 Hard, 11/12 Medium	○	○
2023-IT-01a	Channel plan	9/10 Hard, 11/12 Medium	○	
2023-AU-02	Palindrome Passwords	9/10 Hard, 11/12 Medium	○	○
2023-TW-05	Line Up	9/10 Hard, 11/12 Hard	○	
2023-IE-01	Mapping the roads	11/12 Hard	○	
2023-IT-02	Logs to the warehouse	11/12 Hard	○	○
2023-PT-03	Walking in the Forest	11/12 Hard	○	
2019-NL-12	Painting doors	11/12 Hard	○	○

Abstraction	Modelling & Simulation	Algorithms	Evaluation	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
○			○	○	○		○		○				
○			○	○		○			○				
○		○	○	○				○	○				
○	○	○		○					○				
○	○	○	○	○		○		○	○		○		
○			○	○		○	○				○		
○	○	○	○	○				○	○	○			
○	○	○	○	○		○		○	○	○		○	
○	○	○	○	○		○		○	○	○		○	
○		○	○	○				○	○	○			
○		○	○	○	○	○		○	○	○			
○	○	○	○	○					○	○		○	
○		○	○	○		○	○	○	○	○			
○	○	○	○	○				○	○	○	○		
○			○	○	○			○					○
○	○	○	○	○					○	○		○	
○		○	○	○		○			○	○	○		
○	○		○	○				○	○	○	○	○	
○	○		○	○					○	○	○	○	
○	○		○	○					○	○	○	○	
○	○	○	○	○					○	○	○		○
○		○	○	○			○		○	○			○
		○	○			○			○	○	○		○
○	○	○	○	○					○	○	○		
	○	○			○		○		○		○		
○			○	○		○				○			
○	○		○	○				○	○		○	○	
○		○	○	○		○			○	○	○		
○	○		○	○	○			○	○	○	○	○	○
○			○	○					○	○			
○	○	○	○	○	○				○	○	○		

