Bebras Australia Computational Thinking Challenge

2022 Solutions Guide
Round 2

Secondary School
Grades 7–12

bebras.edu.au

Bebras Australia Computational Thinking Challenge

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

Started in 2004 by Professor Valentina Dagiene from the University of Vilnius, 'Bebras' is Lithuanian for beaver. This refers to their collaborative nature and strong work ethic.

The International Bebras Committee meets annually to assess potential questions and share resources. Questions are submitted by member countries and undergo a vetting process.

The Bebras international community has now grown to 60 countries with over 2.9 million students participating worldwide!

Bebras Australia began in 2014 and is now administered through CSIRO Digital Careers.

In Australia, the Bebras Challenge takes place in March and August–September each year. As of 2020, two separate challenges are offered for each round.

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

Engaging young minds for Australia's digital future

423
Australian schools participated in Round 2 2022

25,498
Australian students participated in Round 2 2022

2.9 million
Students participate worldwide

CSIRO Digital Careers supports teachers and encourages students' understanding of digital technologies and the foundational skills they require in an ever-changing workforce. Growing demand for digital skills isn't just limited to the ICT sector. All jobs of the future will require them, from marketing and multimedia through to agriculture, finance and health. Digital Careers prepares students with the knowledge and skills they need to thrive in the workforce of tomorrow.

csiro.au/digital-careers

CSIRO digital careers

What is a Solutions Guide?

Computational Thinking skills underpin the careers of the future. Creating opportunities for students to engage in activities that utilise their critical and creative thinking along with problem solving skills is essential to further learning. The Bebras Challenge is an engaging way for students to learn and practice these skills.

Within this Solutions Guide you will find all of the questions and tasks from Round 2 of the Bebras Australia Computational Thinking Challenge 2022. On each page above the question you will find the age group, level of difficulty, country of origin and key Computational Thinking skills.

After each question you will find the answer, an explanation, the Computational Thinking skills most commonly used, and the Australian Digital Technologies curriculum key concepts featured.

Contents

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.

More Computational Thinking resources

Visit **digitalcareers.csiro.au/CTIA** to download the Computational Thinking in Action worksheets. These can be used as discussion prompts, extension activities or a framework to build a class project.

Each resource was designed to develop teamwork; critical and creative thinking; problem solving; and Computational Thinking skills.

Computational Thinking skills alignment

2022 Round 2 Questions	Grade level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Years 7+8							
Boxes	Easy	■	■	■	■	■	■
Upcycling	Easy	■			■	■	■
Flooding	Easy		■	■	■	■	■
Robot Arm	Easy			■	■	■	■
Mouse Bot	Easy		■		■	■	■
Kayles Pins	Medium		■		■	■	■
Secret Message	Medium	■	■			■	■
Colouring Code	Medium	■	■		■	■	■
New Residents to Beaverland	Medium	■			■	■	■
Painting the Fence	Medium	■		■	■	■	■
Logs	Hard	■	■	■	■	■	■
Compare	Hard	■		■		■	■
Kangaroo	Hard	■	■	■	■	■	■
Egg Queries	Hard	■	■		■	■	■
Museum Walking	Hard	■		■	■	■	■
Years 9+10							
Alphabet Order Cypher	Easy	■	■			■	■
Hidden Chocolate	Easy	■		■		■	■
Cake	Easy	■		■		■	■
Bebras Runs	Easy	■	■			■	■
Lift	Easy	■		■	■		■
K-POP Contest	Medium	■				■	■
Decorations	Medium	■	■	■		■	■
Whatdoesitdo	Medium	■	■	■	■	■	■
Ice Cream Machine	Medium	■		■	■	■	■
Secret Number	Medium	■	■		■	■	■
Ada's Marble Machine	Hard	■	■		■	■	■
Smart Farm Robot	Hard	■	■		■	■	■
Unification	Hard	■		■	■	■	■
Symbol Reading Robot	Hard		■		■	■	■
Quiz Night	Hard		■	■	■	■	■

Computational Thinking skills alignment

2022 Round 2 Questions	Grade level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Years 11+12							
Taking Leaves	Easy		■		■	■	■
Shop Counters	Easy	■			■	■	■
Sorting Beavers	Easy	■	■		■	■	■
Fruit Stack	Easy	■		■		■	■
Longest Sequence	Easy		■	■	■	■	■
Grocery Shopping	Medium	■			■	■	■
Jumping Jack	Medium		■	■	■	■	■
Counting by Nodding	Medium	■		■		■	■
Playing with Hats	Medium	■		■		■	■
Compact Representation	Medium	■	■		■	■	■
Save the Trees	Hard	■	■	■	■	■	■
Turing Machines	Hard	■	■	■	■	■	■
Mastermind	Hard	■		■		■	■
Truth Table	Hard	■	■	■	■	■	■
Log Sort	Hard		■	■	■	■	■

Australian 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

2022 Round 2 Questions	Grade level	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Years 7+8											
Boxes	Easy	■				■	■				
Upcycling	Easy		■			■	■				
Flooding	Easy	■		■		■	■				
Robot Arm	Easy	■				■	■				
Mouse Bot	Easy			■		■	■	■	■		
Kayles Pins	Medium					■	■				
Secret Message	Medium			■	■	■	■		■		
Colouring Code	Medium			■	■	■	■		■		
New Residents to Beaverland	Medium					■	■				
Painting the Fence	Medium	■		■		■	■		■		
Logs	Hard	■				■	■				
Compare	Hard	■				■	■				
Kangaroo	Hard	■		■	■	■	■		■		
Egg Queries	Hard			■		■	■				
Museum Walking	Hard	■		■	■	■	■				
Years 9+10											
Alphabet Order Cypher	Easy			■		■	■		■		
Hidden Chocolate	Easy					■	■				
Cake	Easy	■		■		■	■		■		
Bebras Runs	Easy			■		■	■				
Lift	Easy	■		■							
K-POP Contest	Medium			■	■	■	■				
Decorations	Medium	■			■	■	■				
Whatdoesitdo	Medium	■		■	■	■	■	■		■	
Ice Cream Machine	Medium	■		■	■	■	■				
Secret Number	Medium				■	■	■				
Ada's Marble Machine	Hard				■	■	■	■			
Smart Farm Robot	Hard			■		■	■	■	■		
Unification	Hard	■				■	■	■	■	■	
Symbol Reading Robot	Hard					■	■	■			
Quiz Night	Hard	■		■	■		■	■			

Digital Technologies key concepts alignment

2022 Round 2 Questions	Grade level	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Years 11+12											
Taking Leaves	Easy										
Shop Counters	Easy										
Sorting Beavers	Easy										
Fruit Stack	Easy										
Longest Sequence	Easy										
Grocery Shopping	Medium										
Jumping Jack	Medium										
Counting by Nodding	Medium										
Playing with Hats	Medium										
Compact Representation	Medium										
Save the Trees	Hard										
Turing Machines	Hard										
Mastermind	Hard										
Truth Table	Hard										
Log Sort	Hard										

Bebras Challenge 2022 Round 2

Years 7+8

This question comes from Romania

Years 3+4
Years 5+6
Years 7+8 Easy
Years 9+10
Years 11+12

Boxes

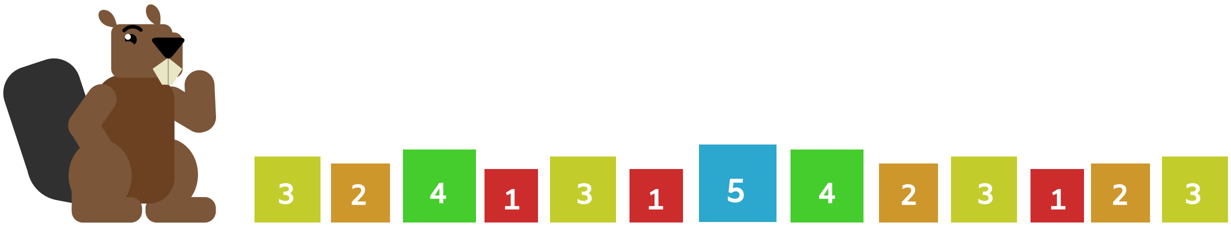
Penelope the Beaver has made wooden blocks of five different weights: 1 kg, 2 kg, 3 kg, 4 kg, and 5 kg. The weight is written on each block.

Penelope is tidying up and wants to put all the block into three tall cabinets: Cabinet A, Cabinet B, and Cabinet C.

Each cabinet can only hold a maximum weight of 15 kg.

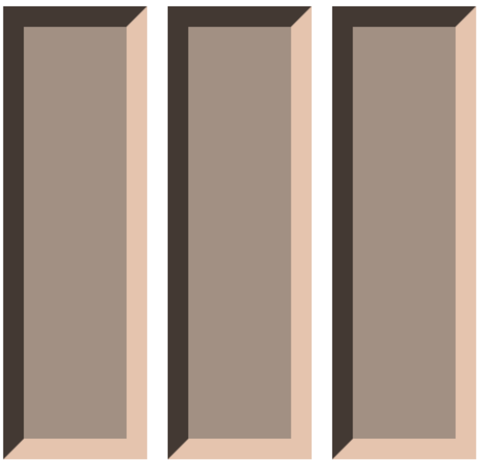
Penelope can't put a heavier block cannot on top of a lighter block, as it might crack the wood.

Penelope puts the blocks in the cabinets in the order that they are lined up, starting from the block on the left shown below. For each block, Penelope always checks Cabinet A first, then Cabinet B, and then Cabinet C, and will put the block in the first cabinet that it fits.



Question

Help Penelope to put all the blocks into the cabinets. Drag the blocks into the cabinets, following the rules.



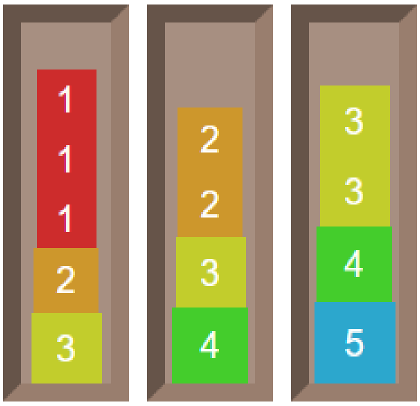
Continued on next page

This question comes from
Romania

Boxes – continued

EXPLANATION

Answer



Explanation

By sticking to the rules, the blocks are placed as follows:

- The first block of weight 3 goes into Cabinet A.
- The second block of weight 2 will be placed over the block of weight 3 in Cabinet A.
- The block of weight 4 can’t be placed in Cabinet A as it would have to sit on top of a lighter block, so it will be placed in Cabinet B.
- The block of weight 1 will be placed in Cabinet A, over the block of weight 2. While it is technically allowed by the second rule to stack this block in Cabinet B or C, Penelope’s third rule says that a block will be placed in the first cabinet where it can fit, which in this case is Cabinet A.

Continuing to follow these rules leads to the arrangement shown above.

BACKGROUND INFORMATION

A set of rules as seen in this question is what computer scientists call *specifications*. Based on these specifications, programmers create an algorithm that corresponds to these specifications: a description of the different steps someone would need to perform to solve the task, based on these specifications.

With these rules, there was no choice on where to put the blocks. The corresponding algorithm could be described as follows, using ‘for’ and ‘if’ statements which are commonly utilised in computer programming:

For each block starting from the left,

...For each cabinet starting with A, until you have placed the block,

......If the cabinet is empty, or the top block in the cabinet is heavier or as heavy as the block you are trying to place then,

.........If the total weight of the blocks in the cabinet, plus the weight of your block, doesn’t exceed 15 kg, then,

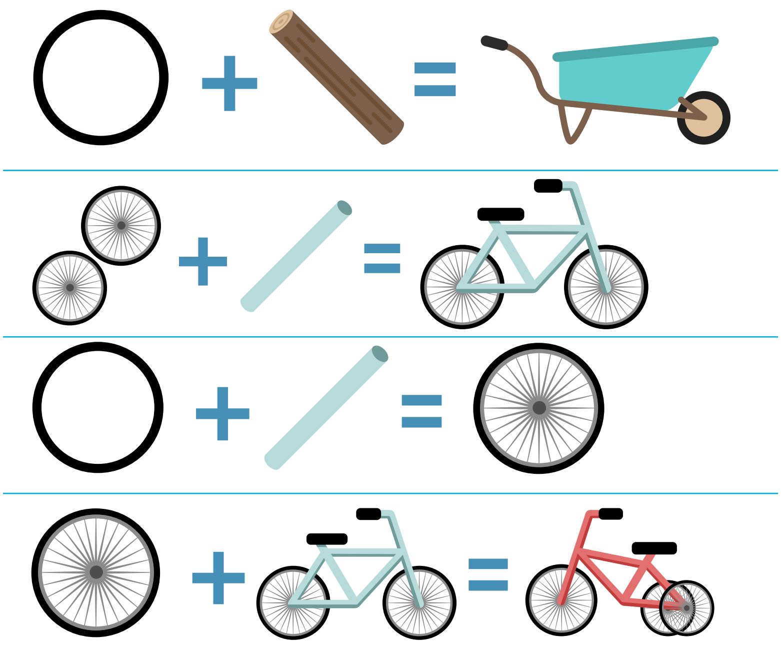
............Place the block in this cabinet.

This is just one way to write the algorithm outlined in the question.

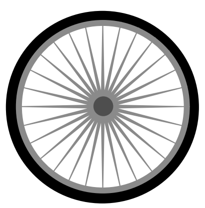
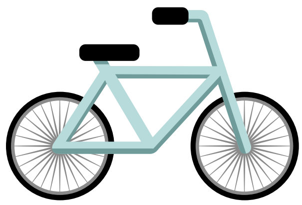
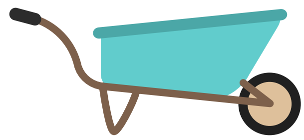
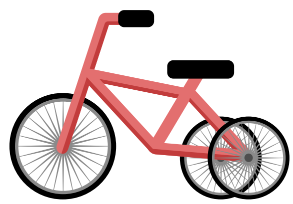
Upcycling

Beavers hate waste. They like to use old worn out things as materials to make new useful items. This is called upcycling.

Shown below are the materials required to make a wheel, a bicycle, a barrow, and a tricycle:



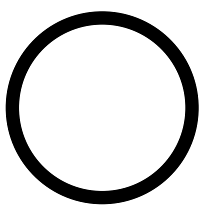
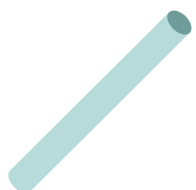
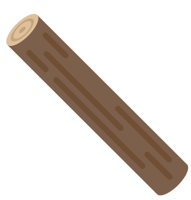
Doreen loves upcycling and likes to sell the items she is making. They can be sold at the market for these prices:

Wheel: $1	Bicycle: $10	Barrow: $5	Tricycle: $15
			

Question

Doreen has these materials: 4 tyres, 4 iron bars and 1 wood piece.
What is the most money she can make by upcycling when she sells the items she makes?

Give your answer in the form of an integer.

		
x4	x4	x1

Continued on next page

Upcycling – continued

EXPLANATION

Answer

$20.

Explanation

Doreen wants to maximise the profit she can make by using the materials in the most efficient way. For example, the value of a tricycle ($15) is more than a bicycle and a wheel ($1+$10=$11), so Doreen should aim to make a tricycle as it sells for $4 more while using the same materials.

A tricycle uses: 2 wheels and 1 iron bar (to make a bicycle), and another wheel.

3 wheels use: 3 tyres and 3 iron bars.

This leaves 1 tyre and 1 wood piece. The highest value item Doreen can now make is a barrow.

Total value of 1 tricycle + 1 barrow = $15 + $5 = $20.

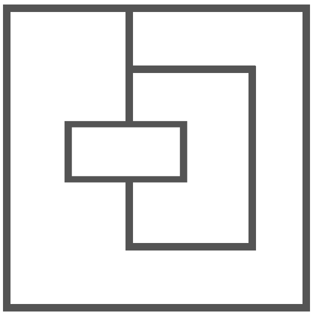
This amount of $20 is the most money that Doreen can make. This occurs when she always tries to build an item of the highest possible value, given what materials she has available. This works for the values in this task but will not always work. For certain combinations of materials, Doreen might need a different strategy to get the highest value in return.

BACKGROUND INFORMATION

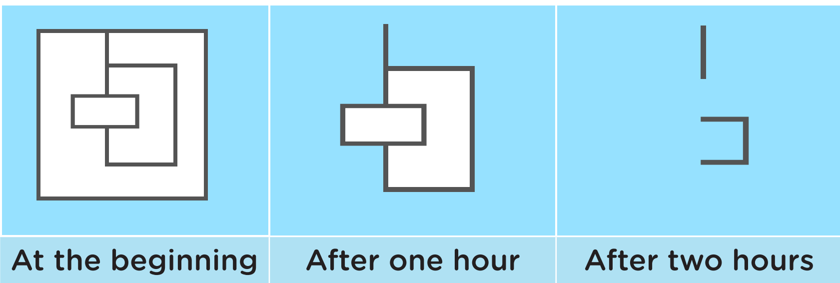
Efficient use of resources is a common problem in society, that computer scientists are often asked to write programs for to optimize. There are many algorithms used. Doreen used a Greedy Algorithm, so called because the algorithm always tries to make the highest value items first. Although this works in this particular case there are other situations where picking the highest value item first limits how many other items you can make and does not get to the highest total value possible.

Flooding

In ancient times there was a small castle with thick black walls.
One day, it was destroyed by a big flood.



When a flood destroys a castle, it happens in the following way:



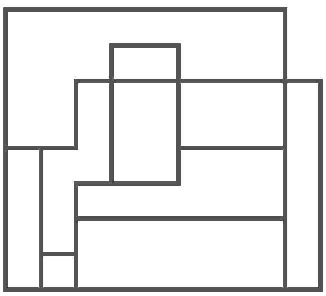
First, the water floods the exterior of the castle.

- After exactly one hour, every wall with water on one side and air on the other side breaks under the pressure of water.
- Water then floods the new area, not bounded by any remaining walls.
- Now, there may be new walls having water on one side and air on the other. After another hour, these walls also break down and water floods further. This procedure repeats until water has flooded the entire area.

After the flood, the ancient Beaverian society built a new castle, shown below.

Question

How many hours would it take to flood the entire area of this new castle?



1 hour | 2 hours | 3 hours | 4 hours | 5 hours | 6 hours

Flooding – continued

EXPLANATION

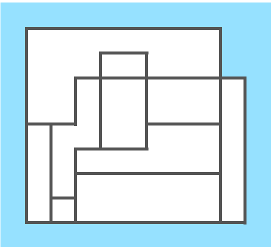
Answer

3 hours.

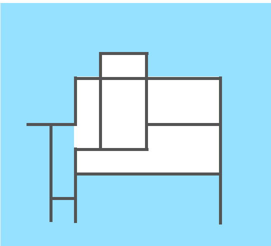
Explanation

According to the rules on how the castle floods, the process could be described by the following images:

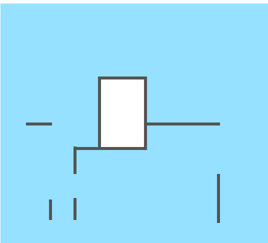
At the beginning:



After one hour:



After two hours:



After three hours:

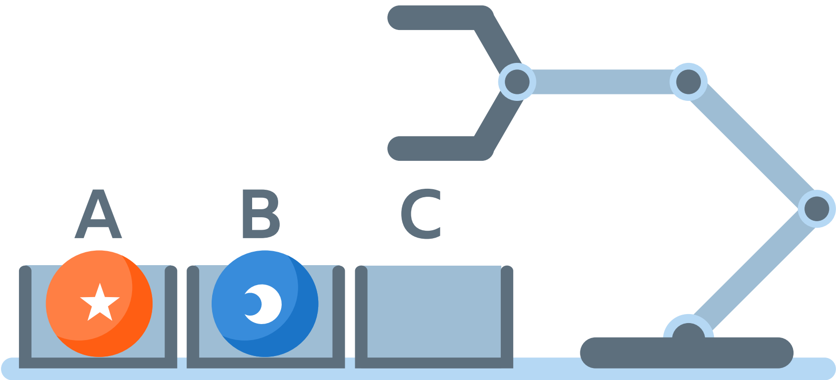
BACKGROUND INFORMATION

Flood fill is an algorithm that determines and alters the area connected to a given starting point. It is used in the bucket fill tool of paint programs to fill a bounded area with a specified colour, and in games such as Go and Minesweeper for determining which pieces are cleared. This Bebras task shows the effect of multiple flood fills. The number of necessary consequent flood fills determines also the shortest distance from a starting point to a destination.

This question comes from Germany

Robot Arm

A class in Beaver High School have built a prototype of a robotic arm. The class have decided to test the arm in the following way.



They place the arm on a table with two balls: one in tray A, and another in tray B. Tray C is empty.

The robot arm follows these steps in order:

1. Pick up the ball in A and put it in C.
2. Pick up the ball in B and put it in A.
3. Pick up the ball in C and put it in B.

Question

When the robot arm is finished, which of the following statements are true?

- The balls have swapped places
- There are two balls in tray A
- There are two balls in tray B
- Tray A is empty
- Tray C is empty
- Nothing has changed. Each ball is back in its place

Continued on next page

Robot Arm – continued

EXPLANATION

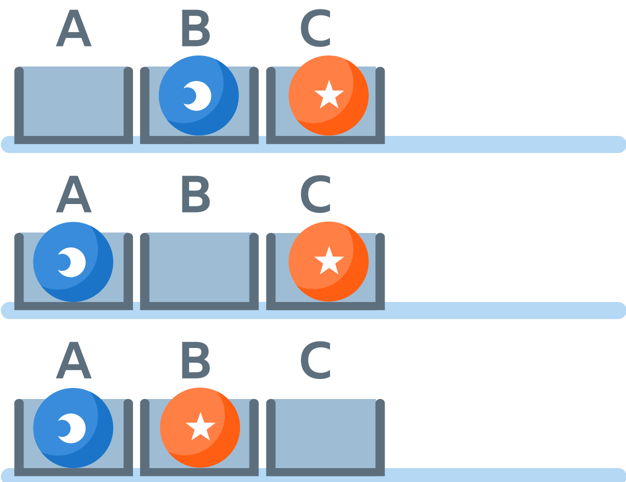
Answer

“The balls have swapped places”, and

“Tray C is empty”.

Explanation

Following the steps described in the question, the robot arm moves the balls as shown in the pictures below:



From this, each of the statements can be determined to be true or false:

- The balls have swapped places - True
- There are two balls in tray A - False, there is one ball in tray A
- There are two balls in tray B - False, there is only one ball in tray B
- Tray A is empty - False, there is one ball in tray A
- Tray C is empty - True
- Nothing has changed. Each ball is back in its place - False, the balls have swapped places.

BACKGROUND INFORMATION

To solve this task one needs to follow an *algorithm* - the steps in the given order, keeping track of the state of the balls after each step.

The robot arm swaps the balls. Because it can only pick up one ball at a time, it has to put the first one aside to make room for the second. It therefore needs three places for the exchange process.

For some programming languages, to swap two variable values a, b, the value of the first variable is stored temporarily in a third *c*:

$c = a$
$a = b$
$b = a$

For numerical integer values, there is a procedure without a new third variable:

$a = a + b$
$b = a - b$
$a = a - b$

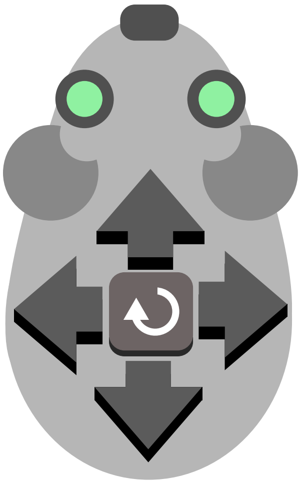
Some programming languages, for example Python, support multiple assignments using tuples, like this:
$a, b = b, a$

This question comes from
Egypt

Years 3+4
Years 5+6
Years 7+8 Easy
Years 9+10
Years 11+12

Mouse Bot

Beaver Hajar has a programmable mouse bot. The bot has 4 arrows and a repeat button:



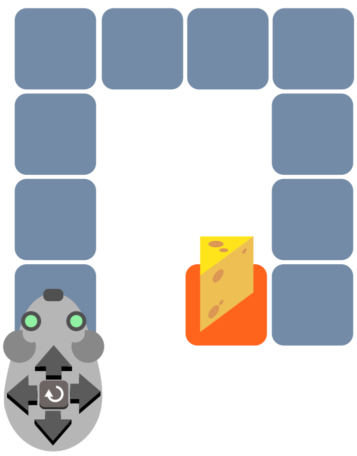
If Hajar would like to move the bot forward for 2 steps, she presses the Forward button  twice.

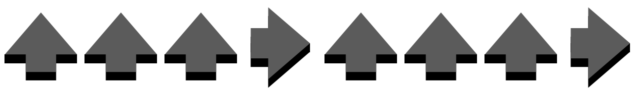
She can also rotate the mouse 90 degrees to the right  or left , or 180 degrees down  by pressing the corresponding buttons. The bot will rotate on the same tile.

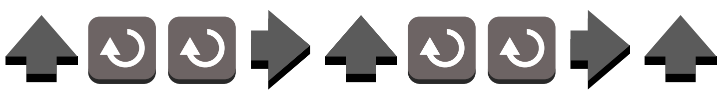
If Hajar would like to repeat a sequence of moves, she will press the dark gray button in the middle - when she presses this button, the bot will repeat the movement of all the inputs since the start of the instructions or the last middle button press.
If she double presses the centre button, the bot will repeat the inputs twice. If she presses it 3 times, the bot will repeat the inputs 3 times, and so on.

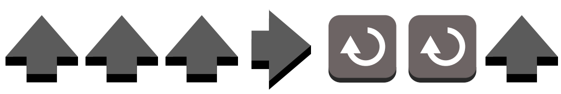
Question

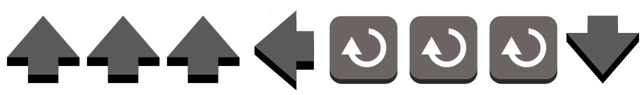
Hajar has to program the mouse bot to reach the cheese. Choose the sequence that will do this.











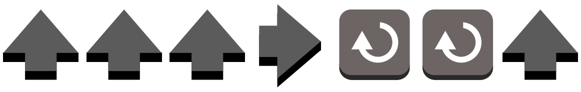
Continued on next page

Mouse Bot – continued

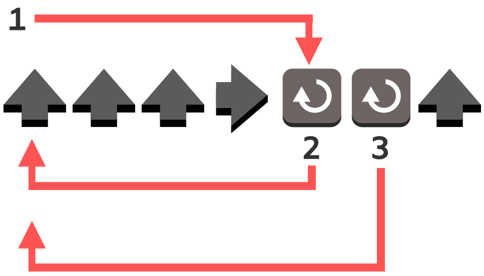
EXPLANATION

Answer

The correct answer is



Explanation



1. The mouse will do the sequence of moving forward 3 times and a rotating 90 degrees to the right. The mouse is now on the upper left corner and is facing towards the east.
2. Since the next step in the sequence is a repeat arrow, it will repeat the previous sequence. So the mouse is now on the upper right corner and is facing towards the south.
3. After repeating the sequence for the third time, the mouse is now on the bottom right corner and is facing towards the west.
4. The last arrow will move the mouse to the tile of the cheese.

Answer  could be right if it has one more move forward arrow at the end. It is a repeated pattern. It will move the mouse to the bottom right corner and facing towards the cheese, but the mouse will not move to the cheese tile.

Answer  will move the mouse one step forward, then will repeat this step 2 times. Now the mouse is on the upper left corner and is facing towards the north. Then it will rotate 90 degrees to the right, after which it will move one step forward. When the repeat button is executed, it will repeat the rotate arrow and the forward arrow 2 times. So the mouse will be out of tiles and, thus, off course.

Answer  will move the mouse forward 3 steps. It will be on the upper left corner, but it's facing towards the west. So the next step will put the mouse off course.

BACKGROUND INFORMATION

An *algorithm* is simply a set of steps used to complete a specific task. They're the building blocks for programming, and they allow things like computers, robots, smartphones, and websites to function and make decisions.

In addition to being used by technology, a lot of things we do on a daily basis are similar to algorithms. Let's say you want to make some spaghetti. In order to do this successfully, there's a certain set of steps you need to follow in a particular order.

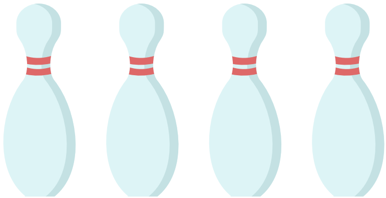
First, you'll need to boil a pot of water. Once it's boiling, you then add the spaghetti and cook it for a set amount of time, stirring occasionally. Once it's finished, you drain the water, then it's ready to be served with a sauce of your choice.

This question comes from
Italy

Years 3+4
Years 5+6
Years 7+8 Medium
Years 9+10
Years 11+12

Kayles Pins

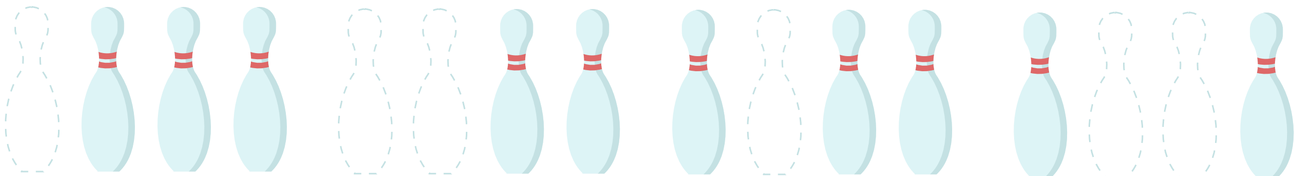
Marlee and Duane have invented a game using four bowling pins. They start by putting the pins in a row, beside each other.



On their turn, they can either drop a single pin or two adjacent pins in one shot (they can always choose which pin(s) to knock down). Whoever drops the last pin wins. Both Marlee and Duane can ensure their victory if they always choose well.

Question

Considering that Marlee plays first, which pin or pins should she drop first to ensure her victory?

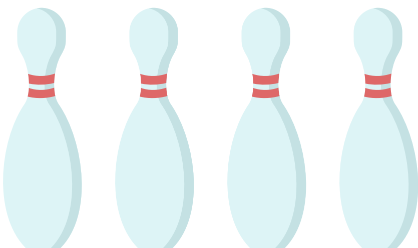


EXPLANATION

Answer

Explanation

Let’s number the pins from left to right.



1 2 3 4

Continued on next page

Kayles Pins – continued

If Marlee knocks down the first pin, Duane wins by dropping the third pin. (Mirroring, if Marlee threw down the fourth pin...)

If Marlee knocks down the first two pins, Duane wins immediately by dropping the last two pins left standing. (Mirroring, if Marlee knocked down the last two pins...)

If Marlee knocks down the second pin, Duane wins by dropping the third pin, as in case A. (Mirroring, if Marlee knocks down the third pin...)

By dropping the pair “second pin and third pin”, Marlee leaves only two detached pins; Duane can not help but knock one of them down, leaving the other to Marlee!

Therefore, case D is the correct answer, as shown in the image above.

BACKGROUND INFORMATION

This question is based on the real game called *Kayles - an impartial (combinatorial)* game (i.e., a game characterized by no distinction of “material” and no possibility of a draw). In general, given a row of $n (> 0)$ bowling pins, the two players take turns to knock out either one pin or two adjacent pins, until all the pins have fallen down. Under the normal play convention, whoever throws down the last pin wins; in this case, the first player has a winning strategy for any row of n (> 0) pins. Indeed, the first player ensures victory by leaving the opponent two equal (and detached) sets of pins: if n is odd, the first player throws down the central pin; if n is even, the first player knocks down the two center pins. Whatever action the second player performs on one of the two sets in the following rounds, the first player will “copy” it onto the other.

Kayles is a special case of the so-called “octal games”, that involve removing tokens from heaps of tokens, possibly arranged in rows, stacks or other forms, generally modeled by means of a graph.

This question comes from
Serbia

Years 3+4
Years 5+6
Years 7+8 Medium
Years 9+10
Years 11+12

Secret Message

Beavers Bonnie and Clyde exchange messages consisting of 12 digits with 0's and 1's. Because Beaver Ben understands their messages, they decided to encode them.

In the first encoding step they replace a pair of consecutive digits by a character A, B, C or D:

00	01	10	11
A	B	C	D

In the second step, they replace a pair of consecutive characters using the system below:

AA	AB	AC	AD	BA	BB	BC	BD	CA	CS	CC	CD	DA	DS	DC	DD
0	2	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Question

If Bonnie and Clyde's final message is C13, what was the initial message in digits 0 and 1?

- 11 00 00 01 00 11
- 10 10 10 10 11 11
- 10 10 10 11 10 11
- 10 10 11 00 10 01

EXPLANATION

Answer

11 00 00 01 00 11.

Explanation

Let's work backwards: C becomes DA, 1 becomes AB and 3 becomes AD, giving us DAABAD. Now, as D is replaced by 11, A by 00 and B by 01, we get the final answer 11 00 00 01 00 11.

BACKGROUND INFORMATION

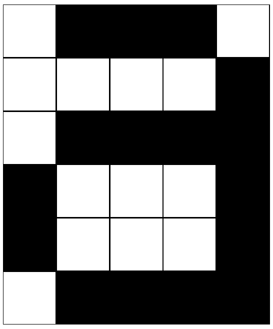
Cryptography is a science that deals with methods of preserving information secrecy. When a message is transferred over an open channel from one place to another, anybody (in the case of our task Ben) can read it. To protect its content, it is encrypted, which makes it not understandable. The basic technique of hiding messages is encoding, which is what is happening in our task using a two-step process. True encryption is more robust. To encrypt a message, the encoding should be parameterised using a key – a shared secret between Bonnie and Clyde. A possible way parametrisation could be how the second row of each table is permuted - the choice of key would change the order of the selections in the table.

This question comes from
Saudi Arabia

Colouring Code

The Beaver School of Art has created an exhibition using black and white squares called pixels. The artists paint their pictures using a clever set of instructions.

An example of a picture of the letter “a” is shown below. The first number in each row always relates to the amount of white pixels at the beginning of the row. If the first pixel is black the row begins with a zero.



1, 3, 1
4, 1
1, 4
0, 1, 3, 1
0, 1, 3, 1
1, 4

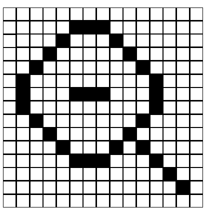
The next numbers alternate betwen white and black pixels. For example:

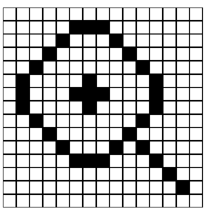
- 2,3,2,1 describes 2 white pixels, followed by 3 black pixels, followed by 2 white pixels, followed by a black pixel.
- 0,3,4,5 describes 3 black pixels, followed by 4 white pixels, followed by 5 black pixels.

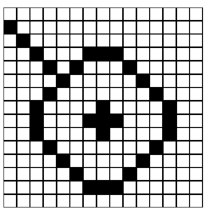
Question

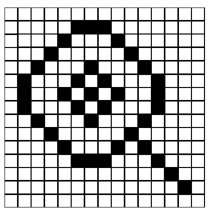
What picture is painted with the following instructions?

15
5, 3, 7
4, 1, 3, 1, 6
3, 1, 5, 1, 5
2, 1, 7, 1, 4
1, 1, 4, 1, 4, 1, 3
1, 1, 3, 3, 3, 1, 3
1, 1, 4, 1, 4, 1, 3
2, 1, 7, 1, 4
3, 1, 5, 1, 5
4, 1, 3, 1, 1, 1, 4
5, 3, 3, 1, 3
12, 1, 2
13, 1, 1
15







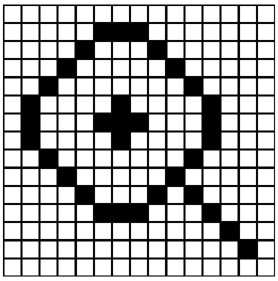


Continued on next page

Colouring Code - continued

EXPLANATION

Answer



Explanation

To find the correct answer, we do not need to check the whole code.

- Since the first row in all images is the same, we can skip the first code line.
- The second code line starts with 5. Thus, the second row in the picture should start with a white pixel. If it started with a black pixel, then the code would have to start with a 0 (according to the rule about the first pixel in the row being black). However, the code does not start with a 0, **so option C is incorrect**.
- The next two rows in the images A, B and D are identical, so we can also skip them.
- The fifth line of code has five numbers, but in image D there are seven segments of white and black. Alternatively, in the middle of this code line there should be a 1 for the black pixel located in the center of the fifth image – but the code calls for 7. **So, option D is incorrect**.
- The sixth code line has seven numbers, but in image A there are only five segments of black and white. Also, the code line should include 9 to refer to nine white pixels, but it does not. **So, option A is also incorrect**.
- **Thus, we have option B as the correct answer**. Just to be sure, we can check the sixth line of code, with the sixth row in image B as follows:
  - 1 – one white pixel
  - 1 – one black pixel
  - 4 – four white pixels
  - 1 – black pixel
  - 4 – four white pixels
  - 1 – one black pixel
  - 3 – white pixels

This is verified by the code, so we can be confident that B is correct.

BACKGROUND INFORMATION

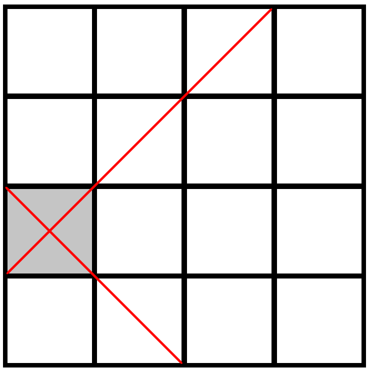
Raster images are represented on screen as small dots in the form of a square. Those dots are called *pixels*. Instead of storing the image as it is, computers use different *compression algorithms* to reduce the amount of stored information. For example, *lossy algorithms* are based on the human psycho-visual system feature to discard high-frequency information, i.e. sharp transitions in intensity, and color hue. Although, high-frequency information will not be stored, the quality of the image will remain high. Also, the size of these files will be reduced. It is possible to change the quality level, but a low level may lead to the appearance of artifacts. JPG is one of the most popular raster image formats.

Another type of digital graphics is a *vector image*. This is drawn by lines (paths) with a starting and ending points. Each path is represented as a mathematical expression. SVG is one of the most popular vector image formats.

New Residents to Beaverland

Four new families have applied for a construction license at the Bebrasland City Hall. The mayor has allocated the families a single plot each inside a large area of land. The area is divided into 16 equal, numbered plots as shown in the picture below:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16



The families can build their lodge inside their own plot, but they must follow four rules.

- Rule 1: Only one lodge can be built in a plot.
- Rule 2: There can be, at most, one lodge in each row of the area.
- Rule 3: There can be, at most, one lodge in each column of the area.
- Rule 4: There can be, at most, one lodge in each diagonal of the area – this is shown in the picture above, where the highlighted square is a lodge (9), and the red lines show the other square areas that are on the same diagonal (3,6,14).

Question

Select a plot for each of the four families to build their lodge that follows the rules.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Continued on next page

New Residents to Beaverland – cont'd

EXPLANATION

Answer

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Explanation

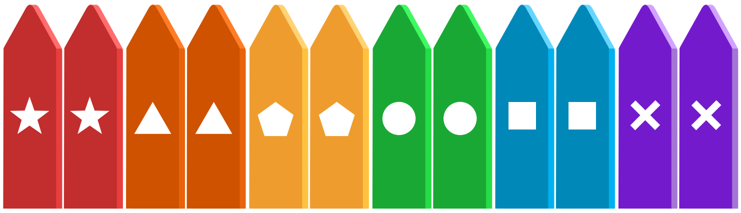
There are two solutions: 3 5 12 14 and 2 8 9 15, both with four ares for the bebras families to build nests, as shown above. These solutions are mirror images of each other. Any other combination of four areas violates at least one of the rules.

BACKGROUND INFORMATION

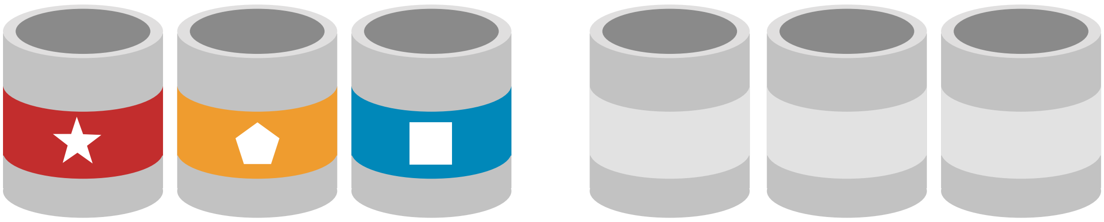
This is a simplified version of the 8 Queens problem with only 4 Queens. A solution can be found by a *backtracking method*. A student places a nest in an area and then checks if there is a place for another nest. If not, they change the place of the nest last placed and try again to place a new nest. If there is no suitable position for a new nest, the position of the previously fixed nest must be changed and so on. To be sure that the best solution is found, a student has to proceed by columns and in columns – by rows.

The task can also be solved by trial-and-error.

Painting the Fence



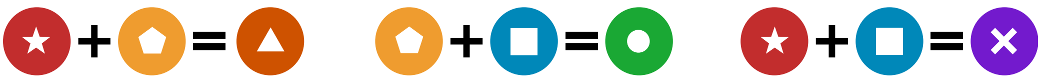
Beaver Jarrah wants to paint a 12-picket fence. They want to paint the fence red, orange, yellow, green, blue and violet, where each colour is assigned to two fence pickets.



Jarrah has three buckets full of red, yellow, and blue paint, and three empty buckets for mixing. The empty buckets are marked so that Jarrah knows when the bucket is one quarter full, half full, and three quarters full. Each picket needs a quarter of a bucket of paint.

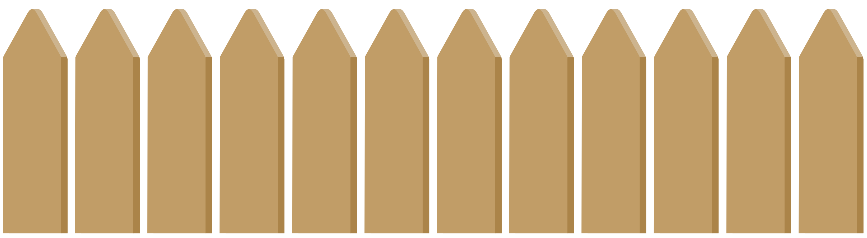
To get orange, green, and violet, Jarrah can mix the paint by following these rules:

- Red + Yellow = Orange
- Yellow + Blue = Green
- Red + Blue = Violet



Question

What is the maximum number of fence pickets can beaver Jarrah paint?



Continued on next page

Painting the Fence – continued

EXPLANATION

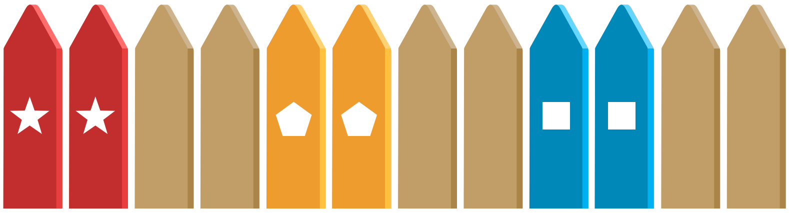
Answer

The correct answer is 12.

Explanation

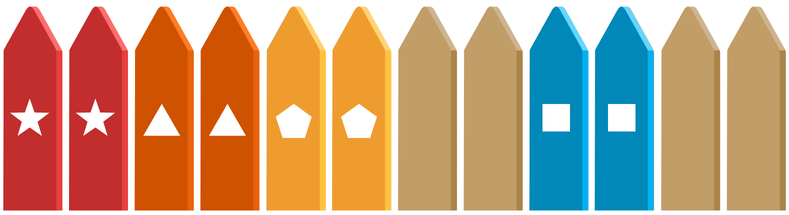
All 12 fence pickets can be painted. A quick check verifies that there are three full buckets of paint, or 12 quarters. Each picket requires a quarter of paint, so all 12 pickets should be able to be painted. However, we need to verify that the correct combination of colours can be achieved.

To do this, first, Jarrah has to paint two pickets red using half a bucket of red paint. Then they do the same with four other pickets using half a bucket of yellow and blue paint respectively.



Now, there are three half-full buckets of paint left.

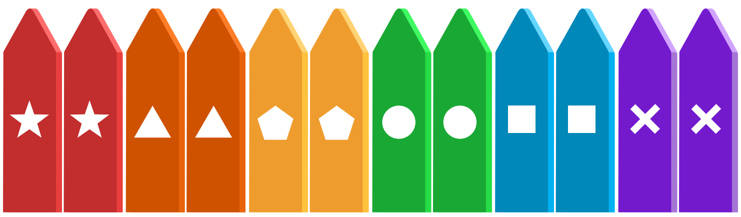
To paint two pickets orange, Jarrah needs to mix one quarter of the red-paint bucket and one quarter of the yellow-paint bucket in the first empty bucket. As a result, Jarrah will get half a bucket of orange paint.



After this step, there is one quarter bucket of red paint, one quarter bucket of yellow paint, and half a bucket of blue paint.

Next, Jarrah can paint two pickets green by mixing one quarter of the yellow-paint bucket and one quarter of the blue-paint bucket in the second empty bucket.

After this step, Jarrah mixes the remaining quarter of the red-paint bucket with the final one quarter of the blue-paint bucket to get a half bucket of violet. Enough for the last two pickets.



Continued on next page

Painting the Fence – continued

BACKGROUND INFORMATION

The main purpose of the RGB (red, green and blue) colour model is for the sensing, representation, and display of images in electronic systems, such as televisions and computers. The RGB colour model is an *additive* colour model in which red, green, and blue light are added together in various ways to reproduce a broad array of colours. The name of the model comes from the initials of the three additive primary colours, red, green, and blue. The RGB colour model is essentially opposite to the *subtractive* colour model (RYB colour model), which this task illustrates.

In addition, in order to understand how computers store colour, one needs to understand "*colour coding*". A digital image, as it comes out of a digital camera or a scanner, can be stored on the hard disk of your computer, processed, enhanced, retouched, and sent to a printer. A digital photograph consists of *pixels*, a pixel being a coloured dot, the smallest element of the image. Each pixel has one and only one colour. These colours are encoded as numbers.

RGB colour coding is probably the most prominent method of colour coding. Essentially, if you specify three numbers between 0 and 255, assign one to R, one to G and one to B, you can define a colour.

For these three numbers, 0 means "none" and 255 means "all". Remember that we are coding light, not ink or paint or something like that. A higher figure means more light. The higher the RGB values, the lighter the colour. The lower the RGB's, the darker the colour. We set RGB to 0,0,0 to get black. For example, RGB = 20,20,20 is a very dark grey, and RGB = 200,200,200 is a light grey.

Logs

Jack and Kalinda are building a log house.

Jack is bringing logs from the Forest to the Storage area. He can move from the Forest to the Storage area in 5 minutes and drag 2 logs at the same time.

Kalinda is taking the logs from the Storage area to the Construction site. She can move from the Storage area to the Construction site in just 2 minutes, but only carry 1 log.

Forest	Storage area	Construction site

Both beavers move at the same speed to and from the Storage area with or without logs. They are working as follows:

- When Jack arrives at the Storage area with new logs, he will drop the logs and call out to Kalinda before returning to the Forest; Kalinda will then stop working at the Construction site and take the logs from the Storage area.
- When Kalinda takes the last log from the Storage area and returns to the house, she will resume doing her work at the Construction site; but if there are logs left at the Storage area, Kalinda will drop the log at the house and immediately return for more logs to the Storage area.

Question

Jack starts in the Forest, and Kalinda starts at the Construction site. How many logs will be at most at the Construction site 30 minutes after the friends start working?

Continued on next page

Logs – continued

EXPLANATION

Answer

The correct answer is 5.

Explanation

In 30 minutes, Jack will have enough time for three runs from the forest to the storage area and back. More concretely, he will arrive with two new logs 5, 15, and 25 minutes after the start.

When Jack arrives with the first two logs 5 minutes into the workday, the storage area is empty and he will call Kalinda. It will take Kalinda 2 minutes to get to the storage area, so she will get the first of the two logs to the construction site at 9 minutes and the second one at 13 minutes. She will then resume other work.

The process repeats itself when Jack arrives with the next two logs at 15 minutes. Kalinda gets those to the construction site at 19 and 23 minutes, respectively.

Out of the third pair of logs that Jack brings to the storage area at 25 minutes, only the first will be at the construction site by the half-hour mark, as Kalinda will drop this log off at the 29 minute mark and not have enough time to return to get the remaining log. This brings the total number of logs at the construction site to 5.

BACKGROUND INFORMATION

The way the two friends are working is similar to the *producer-consumer* model of parallel processing in computers. Jack is the producer of logs for Kalinda, and Kalinda is the consumer of the logs that Jack has produced.

The storage area acts as a *buffer* so that Jack does not have to wait until Kalinda comes to collect the logs; instead, Jack can return to forest for the next pair of logs immediately and be more productive. Jack calling out to Kalinda when he adds new logs to the empty storage area is like the *signals* used in computer systems to allow one program to alert another. This lets Kalinda do other work instead of just waiting at the storage area. However, when Jack does call to Kalinda, it takes some time for Kalinda to go from the construction site to the storage area, causing *latency* in the movement of logs.

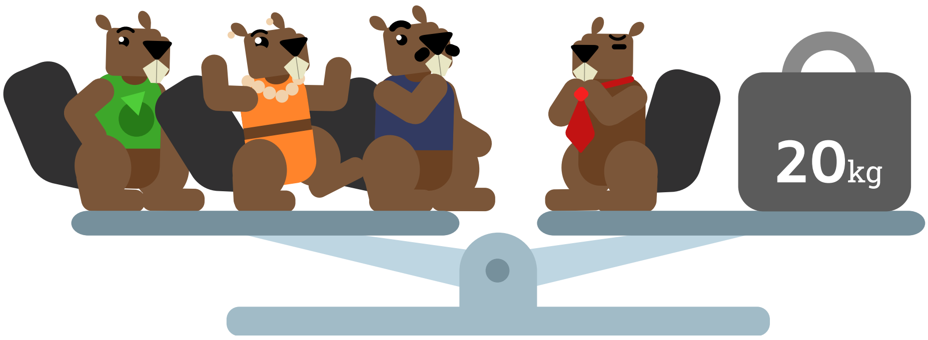
A difference of our task from the classical producer-consumer model is that in our case all the logs are considered equal and it is not required for Kalinda to bring the logs to the construction site in the same order as Jack collected them in the forest.

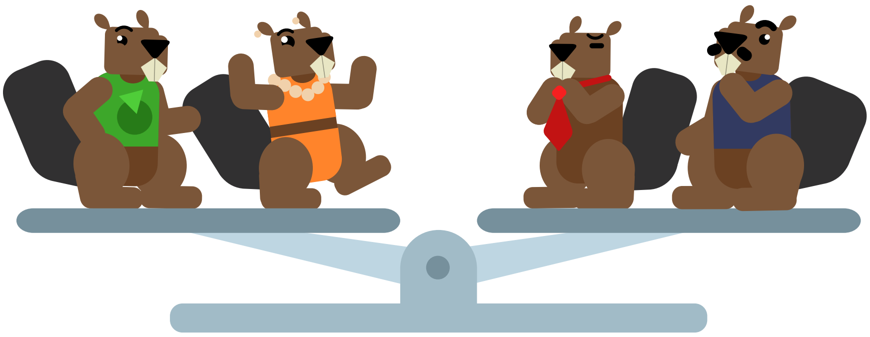
This question comes from
Lithuania

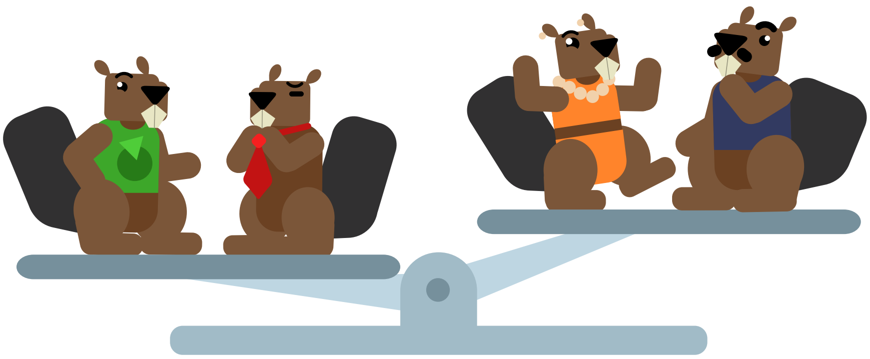
Years 3+4
Years 5+6
Years 7+8 Hard
Years 9+10
Years 11+12

Compare

Four beavers Ada , Berta , Cezar , and Dan  are playing with a scale and took many photos. Three of the photos are shown below:





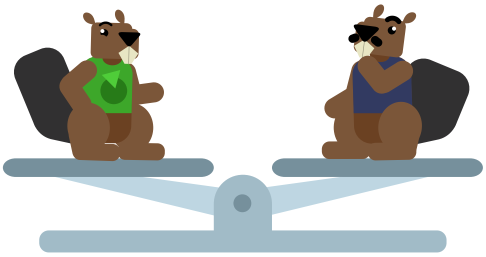


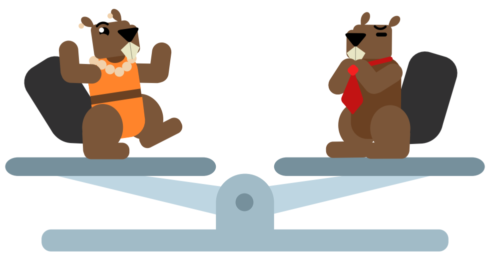
Continued on next page

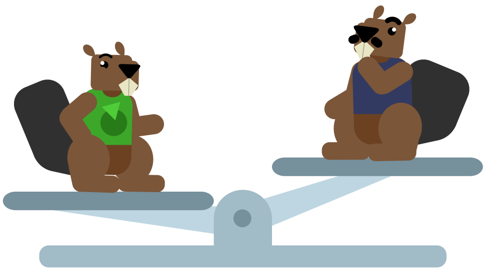
Compare – continued

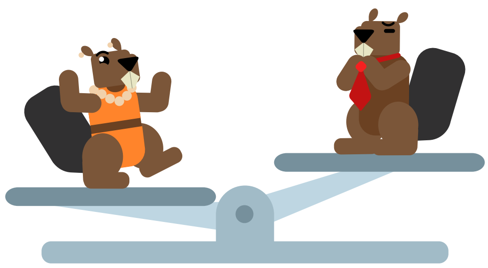
Question

Which of the following photos could also have been taken?



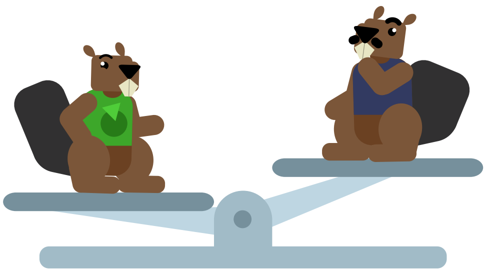






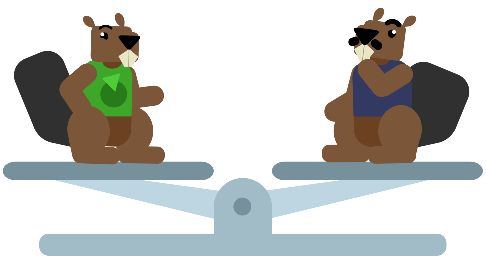
EXPLANATION

Answer



Explanation

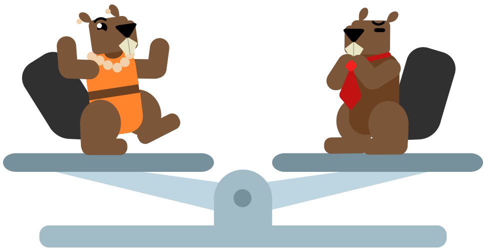
One way to solve this task is by using deductive reasoning:



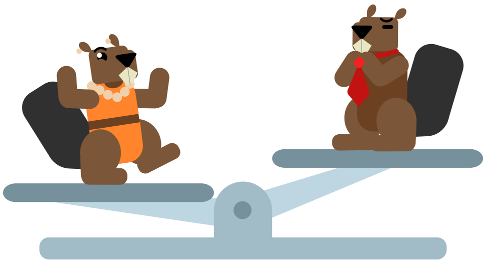
If this photo was true, then Ada also weighs the same as Cezar from the middle photo. In which case, in the lower photo, the scale should have been balanced (Ada+Dan = Berta+Cezar). This is a contradiction so the assumption of first photo cannot be true.

Continued on next page

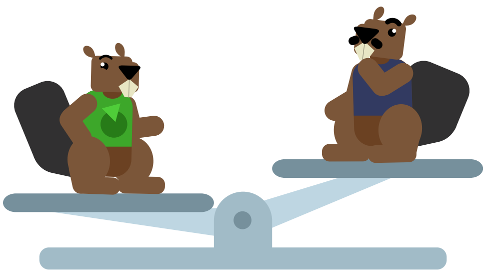
Compare – continued



If this photo was true, then Berta also weighs the same as Dan from the middle photo. The above argument still applies - the scale in the lower photo should have been balanced (Ada+Dan = Berta+Cezar). Therefore, the assumption of second photo cannot be true.



If this was true photo, then Berta weighs more than Dan (Berta > Dan). Cezar must weigh more then Ada (Cezar > Ada) to keep the scale balanced in the middle photo. This means in the lower photo, it cannot be true that Berta+Cezar together weigh less than Ada+Dan. As a matter of fact, it should be the other way around. Therefore, the assumption of the fourth photo cannot be true.



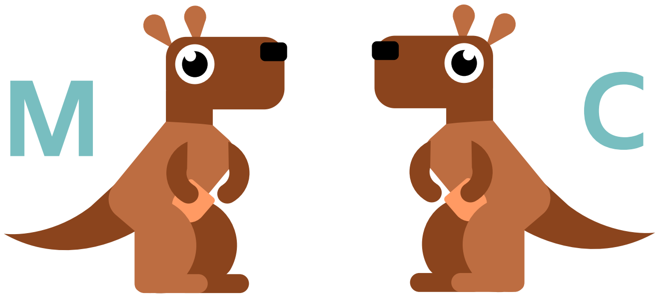
If this photo was true, then Cezar weighs less than Ada (Ada > Cezar). Berta must weigh less than Dan (Berta < Dan) to keep the scale balanced in the middle photo. There is no contradiction that Ada+Dan weigh more than Berta+Cezar as shown in the lower photo. Subsequently there is also no contradiction that 20kg is enough to balance the scales in the top photo. Therefore, this photo is allowed and is the correct answer.

BACKGROUND INFORMATION

Logical reasoning is fundamental to computer science. Essentially, a computer program is an implementation of logical steps that lead to a solution of the problem at hand. But logical reasoning is not only important in programming, computer scientists often use logical and mathematical analysis to determine the complexity of a program and to prove the correctness of the program.

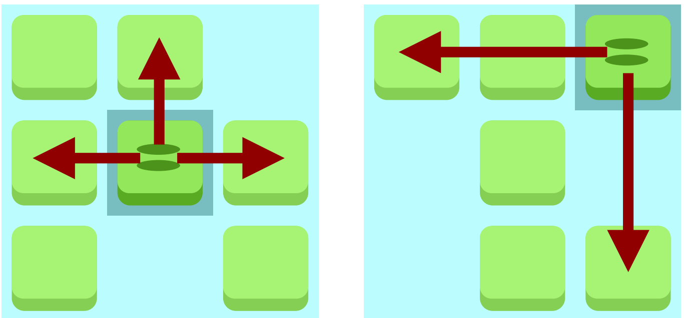
By finding contradictions like in this question, programmers are able to find bugs in their code, or identify and change the way they plan to create a program.

Kangaroo



Kangaroo Mae (M) needs to reach Kangaroo Claire (C). To achieve this, Mae needs to jump on little green islands in a swamp. The swamp is represented as a grid as shown below.

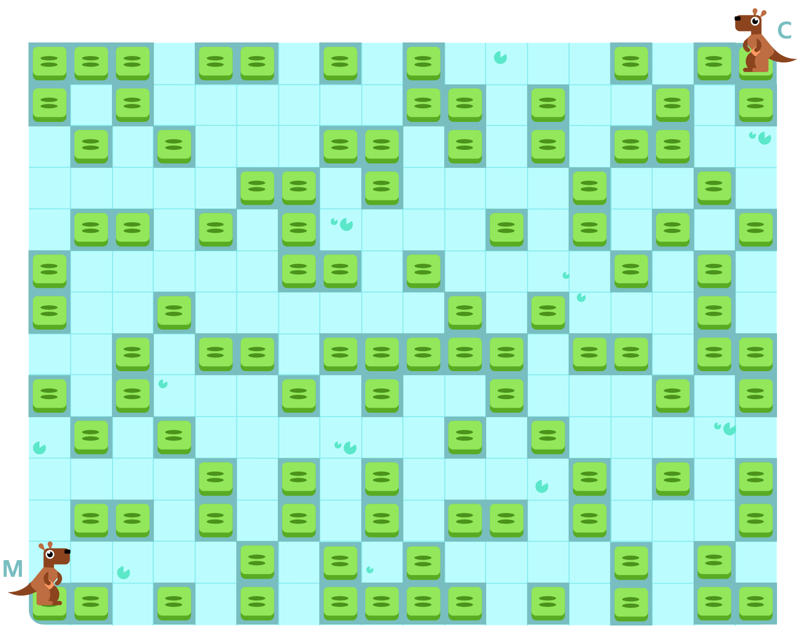
Mae can only perform two kinds of jumps: short and long. Short jumps are to move from a cell to any of four neighbouring cells. Long jumps are to jump straight over any of four neighbouring cells. The images below show short jumps on the left and long jumps on the right:



Mae cannot make any other jumps (diagonal moves, jumps over two or more cells, etc). Long jumps are more tiring and dangerous, so she cannot make two consecutive long jumps.

Question

Find the way for Mae to reach Claire by clicking on the cells in the exact order of Mae's jumps. The last clicked cell must be Claire's cell.



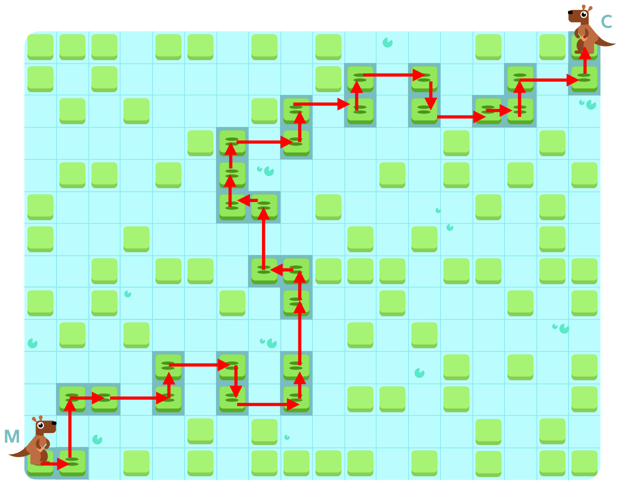
Continued on next page

Kangaroo – continued

EXPLANATION

Answer

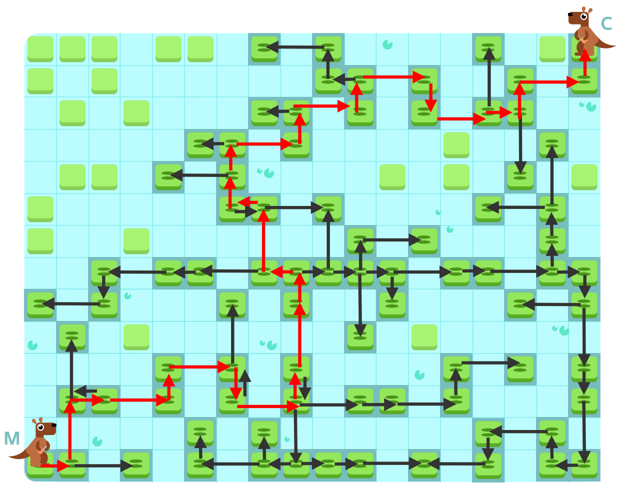
The picture below shows the solution to the problem.



Explanation

All the possible cells that Mae can visit are highlighted. The correct path is in red, while all other accessible paths that lead to a dead-end are in grey.

One way to solve this problem is via trial-and-error modelling. Whenever Mae reaches a tile that can branch off in multiple paths, one route is selected. If this route ends in a dead-end, the path is backtracked to the fork, and the other path is chosen. If all routes in the fork are dead-ends, then the path is backtracked to the previous fork, and so on.



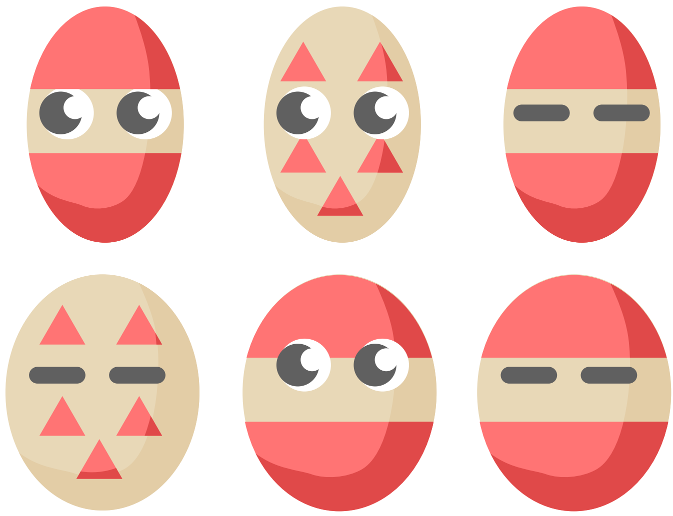
BACKGROUND INFORMATION

This question can be approached using a *Depth-First Search (DFS)* algorithm. Like *Breadth First Search*, which looks for all of the nodes in a graph that satisfy a certain condition, Depth-First Search is a graph algorithm. Its principle is to fully explore a path before moving on to the next. From a vertex, it is about exploring a path until reaching a dead end or a vertex already visited. DFS is often expressed recursively.

A DFS strategy is outlined in the answer explanation of the question - searching down a route, and if a dead-end is reached, backtracking to the previous fork and trying the next possible path, and so on.

Egg Queries

Mary painted six Easter Eggs as shown in the image below.



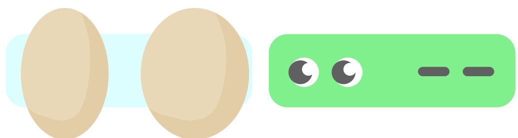
Sian chooses one of them, without telling Mary her choice.

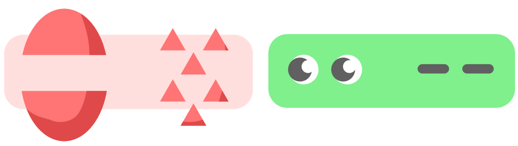
In order to guess which egg Sian selected, Mary is allowed to ask two questions from the following:

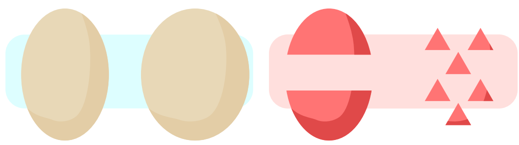
- Is your egg thin or wide? (*shown in blue below*)
- Does your egg have a red top and bottom, or red triangles on its face? (*shown in red below*)
- Does your egg have open or closed eyes? (*shown in green below*)

Question

Which combination of questions offers the best chance to be able to identify the chosen egg, regardless of what egg Sian chose?







It does not matter

Continued on next page

Egg Queries - continued

EXPLANATION

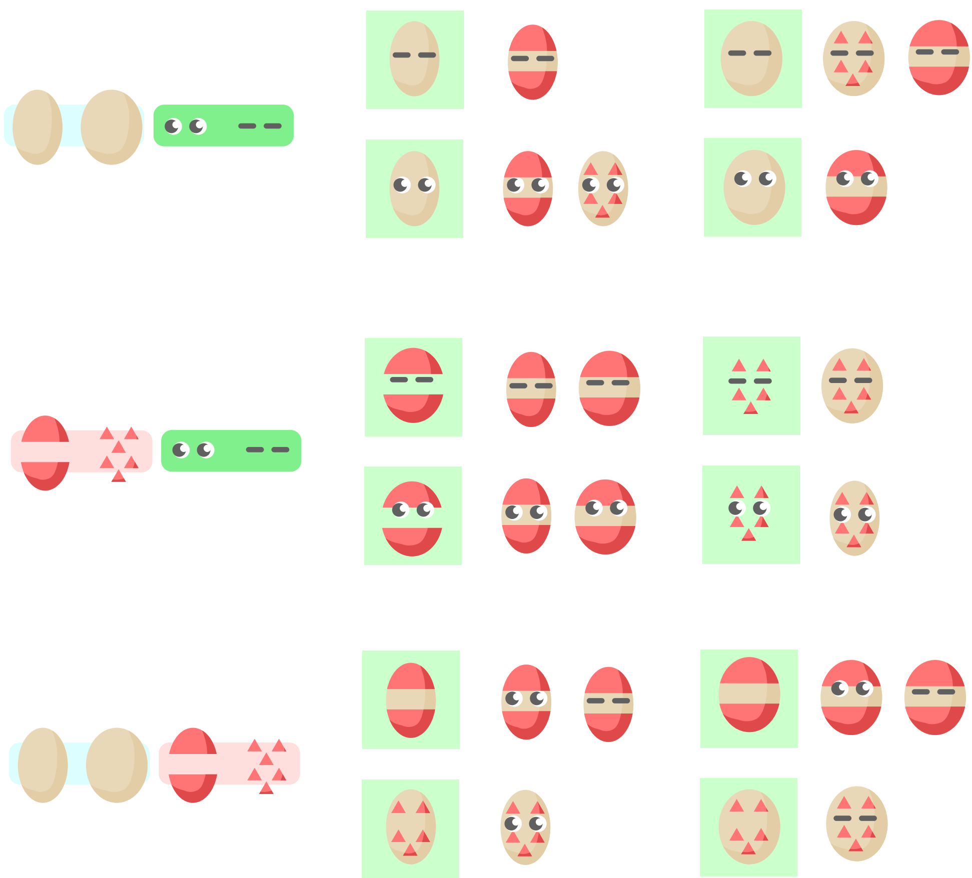
Answer

It does not matter.

Explanation

The images on the right of each option below show that two possible combinations can identify the one possible egg, but there are two other combinations that leave two possible eggs.

This means that by asking two questions, it is possible to distinguish only four different types of eggs. But there are six eggs. So there is a possibility not to identify the egg that Sian chose. It doesn't matter which combination of questions is chosen - they will always split the eggs into two groups of two eggs, and two groups of one egg. Therefore, the correct answer is that it does not matter which questions Mary decides to ask!



BACKGROUND INFORMATION

For many situations in both computer science and in other real world places there are groups of objects with binary attributes - characteristics that are one of two options. With n attributes, it is possible to distinguish 2^n different instances. For example, for an egg that can be big or small, and either red or blue, there are $2^2 = 4$ possible eggs that could exist.

Each attribute can be described using a bit, a binary digit (0 or 1). In analysing algorithms for searching through these objects, understanding these binary distinctions is needed. It is also possible to express these distinctions as booleans (logical variables 'true' or 'false' instead of 0 and 1).

This question comes from Russia

Years 3+4
Years 5+6
Years 7+8 Hard
Years 9+10
Years 11+12

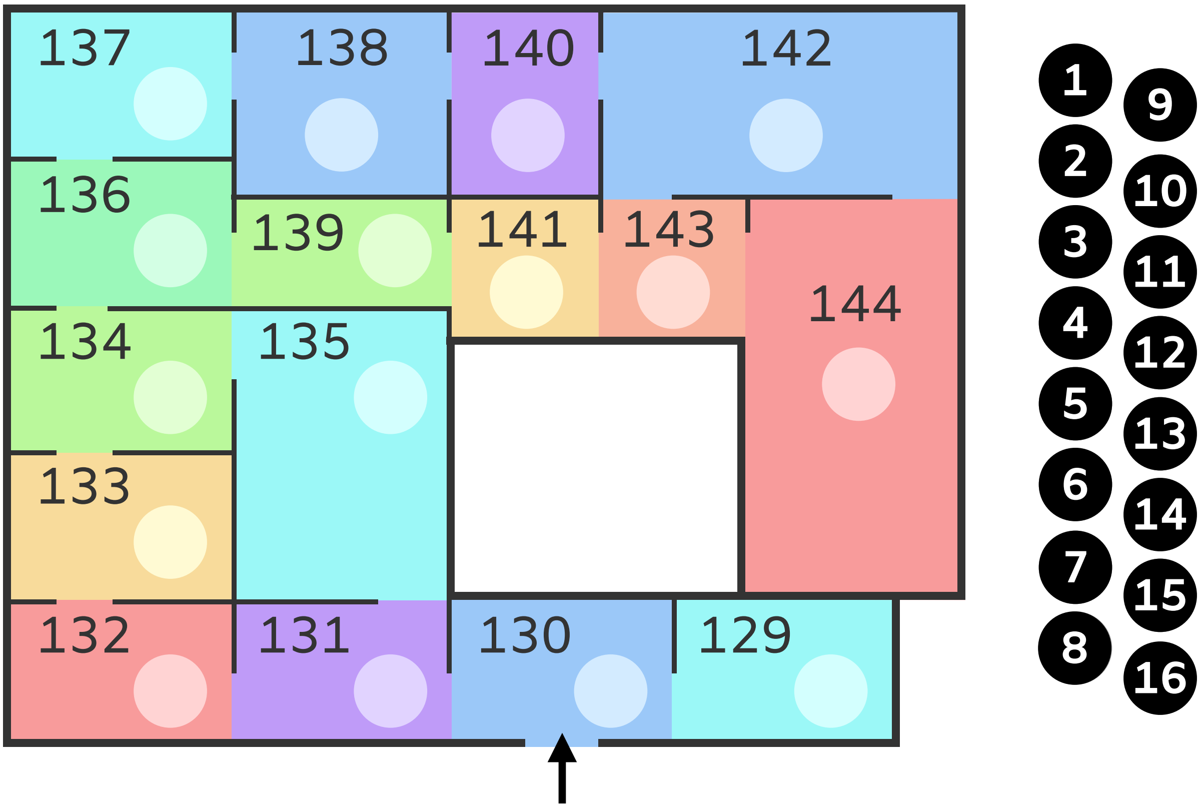
Museum Walking

Here is a plan of one section of the Louvre in Paris, France. Victoria arrives, entering from the black arrow. She wants to visit each room and view the artifacts in order based on the following rules:

- Every time she arrives in a room, she will check if there is an unvisited adjacent room:
  - If so, she will immediately move to the unvisited adjacent room with the smallest possible number.
  - If not, she will view the artifacts in the current room.
- When she finishes viewing the artifacts in a room, she will return to the last room she came from and repeat the above steps.
  - Victoria will never return to a room where she has viewed the artifacts, even if she just came from there - she will instead return via the next most recent way she entered the room she is in.

Question

Trace Victoria's steps through each room and mark in which order she views the artifacts.

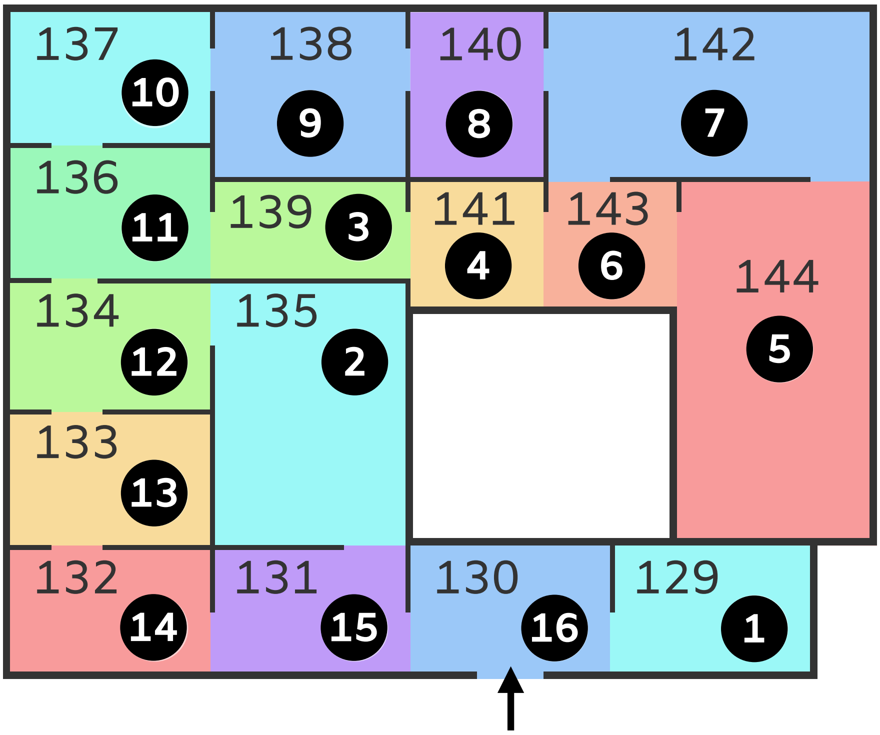


Continued on next page

Museum Walking – continued

EXPLANATION

Answer



Explanation

The above image shows the order in which Victoria views the artifacts. In the beginning, Victoria is in room 130, and there are 2 unvisited adjacent rooms, 129 and 131. She then visits room 129 where there is no unvisited adjacent room, so she views the artifacts there and returns to room 130.

Similarly, from room 130, she visits 132 → 133 → 134 → and arrives in room 135 where there is no unvisited adjacent room, so she views the artifacts there and returns to room 134.

Then from room 134, she visits 136 → 137 → 138 → 140 → 142→ 143 → 141 → and arrives in room 139 where there is no unvisited adjacent room, so she views the artifacts there and returns to room 141.

Then she views the artifacts in room 141 and returns to room 143. From room 143, she visits room 144, the last unvisited room. After viewing the artifacts in room 144, she returns to room 143. She views the artifacts in this room, and then returns to 142, as this was the most recent way she came in where she is returning to a room with unvisited artifacts. She then views artifacts in each room in reverse, all the way back to room 130.

BACKGROUND INFORMATION

The way Victoria searches for unvisited rooms is based on an algorithm known as *Depth-First-Search (DFS) algorithm*. Alongside the Breadth-First-Search algorithm, it is a common way to go through the data structure organized as a graph if we don't want to miss any of its elements.

Its principle is to fully explore a path before moving on to the next. From a vertex, it is about exploring a path until reaching a dead end or a vertex already visited.

Bebras Challenge 2022 Round 2

Years 9+10

This question comes from Turkey

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Easy
Years 11+12

Alphabet Order Cypher

The Beavarian Astronomy Society has developed an algorithm for encryption. An encrypted word has two parts: the first part consists of the numerical value of the word to be encrypted, and the second part consists of the alphabetical order of each letter in the word. They use the table below for all encryption operations:

A	B	M	N	O	R	S	T	U
1	2	4	10	50	180	300	650	960

For example, the word “MARS” is encrypted as follows:

- The numerical value of the word is equal to the sum the corresponding values of the letters in the table (4+1+180+300= 485).
- In the word MARS, if we order all letters alphabetically we have A-M-R-S. The alphabetical order index is A=1, M=2, R=3, S=4, which gives 2134.

Therefore, the encryption for the word MARS is 485;2134.

Question

If the word SATURN was encrypted using the same algorithm, which of the following would be the correct encryption?

1440;415632 | 1440;718964 | 2101;415632 | 2101;718964

EXPLANATION

Answer

2101;415632.

Explanation

The solution can be found by directly following the encryption algorithm. However, there are shortcuts to arrive at the correct option faster:

First we check the first part of the code - a sum of the encrypted letters. There are two possible answers: 1440 and 2101. The word SATURN contains the letters T and U from the right side of the table with very big numbers, 650 and 960. It is visible that the sum of just these two numbers is bigger than 1440. So the correct sum must be 2101.

The second part of answers contains only two possibilities: 415632 and 718964. SATURN has 6 letters and the ciphers in the code refer to the positions of the letters. It is impossible for the cipher to include a number 7 or greater (in the word SATURN there is no letter in 7-th place). So 718964 is wrong and 415632 must be correct.

Answers A is incorrect because the first part of the code is incorrect.
Answer D is incorrect because second part of the code is incorrect.
Answer B is incorrect because both parts of the code are incorrect.

Continued on next page

Alphabet Order Cypher – cont'd

To prove that answer C is indeed correct, the full algorithm can be performed:

First part: S=300, A=1, T=650, U=960, R=180, N=10. Total value = 300+1+650+960+180+10= 2101.

Second part: alphabetical order of letters in the word is S=4 A=1 T=5 U=6 R=3 N=2 and the code is 415632.

Hence the encryption is 2101;415632.

BACKGROUND INFORMATION

This problem uses the idea of a *transposition cipher* in combination with the use of a *checksum* for encoding. In a transposition cipher, the letters are reversed, and the key indicates the correct order of the letters. Message checksums are used for encoding rather than encryption. Checksums help us to identify errors caused by interference in the transmission of messages.

This question comes from
New Zealand

Hidden Chocolate

Four students, Arjun, Ben, Charlie and Dian decided to play a prank by hiding their teacher's chocolate. When the teacher realised that the chocolate was missing, they asked the four students, "Which one of you hid the chocolate?".

The students gave the following answers:

- Arjun: I didn't hide the chocolate.
- Ben: I know that Dian didn't hide the chocolate.
- Charlie: Amy hid the chocolate.
- Dian: Either Ben or Charlie hid the chocolate

The chocolate was hidden by a single student. Only one student's statement is true - the other three students are lying.

Question

Which student hid the teacher's chocolate?

Arjun	Ben	Charlie	Dian

EXPLANATION

Answer

Dian. The answer Arjun was also accepted*.

Explanation

The answer can be found by considering the outcomes of whether each student hid the chocolate. For example, the proposition "Arjun hid the chocolate" can be represented by P_A and the proposition that "Arjun didn't hide the chocolate" can be represented by $\neg P_A$. By formal logic, P_A and $\neg P_A$ cannot be simultaneously true.

The four statements that each of the students make can now be represented in this new format as shown below (with the symbol in brackets indicating what is true if the student is lying):

Arjun $\neg P_A$ (P_A)
Ben: $\neg P_D$ (P_D)
Charlie: P_A ($\neg P_A$)
Dian: $P_C \vee P_D$ ($\neg(P_C \vee P_D)$)

for Dian, 'v' means 'or' (either Charlie OR Dian hid the chocolate), so if Dian is lying the symbols mean Charlie OR Dian didn't hide the chocolate.

Continued on next page

Hidden Chocolate – cont'd

Now we can work through the possibilities of who is telling the truth. Only one student's statement can be true:

- If Arjun is telling the truth, then Charlie is lying and that re-enforces it wasn't Arjun. Ben is lying so that means Dian hid the cholate. This is supported by Dian's lie which leads to neither Ben or Charlie having taken the chocolate. There is no contradiction here.
- If Ben is telling the truth, both Arjun and Charlie are lying so we have (P_A) and ($\neg P_A$) – a contradiction.
- If Dian is telling the truth, Arjun and Charlie are lying so again, a contradiction.
- If Charlie is telling the truth, then both Arjun (who is lying) and Charlie's statements would lead to Arjun hiding it. But Ben is also lying, which means P_D. This means both Dian and Arjun would have hidden it and again we have a contradiction as only one student hid the chocolate.

Only the situation where Arjun is telling the truth doesn't result in a contradiction. If Arjun is the only student telling the truth, then the one truth and three lies means that Dian took the chocolate. Therefore, Dian is the correct answer.

**The answer Arjun was also accepted. Charlie's statement was intended to read "Arjun hid the chocolate" and not "Amy hid the chocolate". However, due to this typo in the question text, students who interpreted Amy as an additional character could reasonably arrive at a solution where only Ben is telling the truth and hence Arjun hid the chocolate. These students were also marked as correct and were not penalised.*

BACKGROUND INFORMATION

The concept of a condition being True or False is used in programming. For example, a program may check if some variable x is either equal to 3 (*x==3*) if x is not equal to 3 (*x!=3*) and perform a different action depending on which statement is true. Similarly, the concept of two things being joined with an 'or' is a basic concept in programming.

The device used to explain the correct answer is called *propositional logic*. It has its roots in both Philosophy and Mathematics but is also often studied by Computer Scientists. Computer Science is a discipline in which the objects that we want to reason about are extraordinarily complex, and are often abstract and formal. Therefore, there is a need for logic to be especially clear - breaking a problem down using propositional logic can help simplify the task. It has many practical applications such as in the design of computing machines, artificial intelligence, the definition of data structures for programming languages, and in many more areas.

This question comes from Pakistan

Cake

Bruna has two open cartons of 1L of milk, each of them being *slightly* more than half full. She wants to make a cake and needs to measure *exactly* 700ml of milk into one of the two cartons. She also has an empty bottle of 500ml and an empty cup of 100ml.

In the process of obtaining exactly 700ml, Bruna can completely fill or empty any of the four containers.

1	2	3	4

Question

Bruna represents four possible sequences of pouring milk using the symbols below. Two items connected by an arrow means that Bruna pours the contents of the first container (left) into the second (right) until this last container is full or until the first is empty, whichever happens first.

Which of the sequences is both the shortest and succesfully measures exactly 700ml into one of the cartons?

- (1)→(4); (1)→(3); (1)→(2); (3)→(1); (4)→(1); (2)→(4); (4)→(1)
- (1)→(2); (2)→(3); (3)→(4); (4)→(2); (3)→(4); (4)→(2)
- (2)→(3); (2)→(1); (3)→(2); (1)→(4); (4)→(2); (1)→(4); (4)→(2)
- (1)→(4); (4)→(2); (1)→(4); (4)→(2)

Continued on next page

Cake – continued

EXPLANATION

Answer

(1)→(2); (2)→(3); (3)→(4); (4)→(2); (3)→(4); (4)→(2)

Explanation

By following the sequence in option (B), which is (1)→(2); (2)→(3); (3)→(4); (4)→(2); (3)→(4); (4)→(2), Bruna succesfully ends up with 700ml in one of the cartons:

First, she fills the second pack to 1000ml, then pours 500ml into a third bottle. Then, using the fourth bottle, she pours 100ml twice into the second container. This means that there is exactly 700ml in carton 2. In total, this sequence used 6 pours.

To verify this is the correct answer, the other options need to be checked:

- In sequence (A) we cannot be sure that the 500ml bottle will be filled with milk, because after pouring 100ml from the first pack into the 100ml recipient we do not know for sure that we still have 500ml in the pack - therefore this sequence doesn't achieve Bruna's goal and is incorrect.
- In sequence (C) one of the cartons ends up with 700ml, but it takes 7 pours - one more pour than option (B).
- In sequence (D) we do not know exactly how much milk was initially in the second carton, we only know it had a little more than 500ml. So after pouring 100ml twice into the second carton there will be *slightly more* than 700ml. Therefore, this sequence is incorrect.

In reality, once (B) is verified to be a sequence that results in 700ml in one of the cartons, only sequence (D) needs to be checked to verify that (B) is the right answer, because sequences (A) and (C) are longer than (B).

BACKGROUND INFORMATION

This task is related to the concept of an algorithm, a sequence of instructions needed to perform some task - in the particular question, the task is to obtain 700ml of milk in one of the available recipients. All computers work by executing sequences of instructions in order to run their system as well as to run applications.

Algorithms can often become complex and long for large applications. Computer Scientists often represent steps in the algorithm using symbols which makes the flow of the application easy to understand and interpret - this can help identify bugs and reveal parts of the algorithm that can be optimised. This is the essence behind many coding languages - block coding is a prime example of creating an easy-to-understand representation of complex algorithms.

This question comes from
Poland

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Easy
Years 11+12

Bebras Runs

A fellowship of nine beavers must solve a riddle to open an ancient door on their quest. To open the doors, the beavers must sort a sequence of numbers from smallest to largest. Beaver Boromir believes he has a great idea that can sort any list of numbers they encounter. He goes through the list of numbers from left to right and performs the following steps:

- He compares the current number with the next number in the list.
- If the next number is smaller than the current number he swaps them.
- He moves to the next position in the list and repeats the steps above.
- When he reaches the end of the list, this is called one pass.

Boromir performs one pass on the following list of numbers:

5	3	5	6	7	4	3	6	8	4

The steps that Boromir performs in the first pass are highlighted below.

3556743684

3**55**6743684

35**56**743684

355**67**43684

3556**47**3684

35564**37**684

355643**67**84

3556436**78**4

35564367**48**

After he has finished the first pass, the list of numbers looks like this:

3	5	5	6	4	3	6	7	4	8

Question

What does the list of numbers look like after two more passes?

EXPLANATION

Answer

3543564678.

Continued on next page

This question comes from
Poland

Bebras Runs – continued

Explanation

Performing two more passes leads to the following list order:

3556436748

3**55**6436748

35**56**436748

355**46**36748

3554**36**6748

35543**66**748

355436**67**48

3554366**47**8

35543664**78**

And,

3554366478

3**55**4366478

35**45**366478

354**35**66478

3543**56**6478

35435**66**478

354356**46**78

3543564**67**8

35435646**78**

Therefore, the sequence will be the following after the fellowship of beavers make two more passes:

3	5	4	3	5	6	4	6	7	8

BACKGROUND INFORMATION

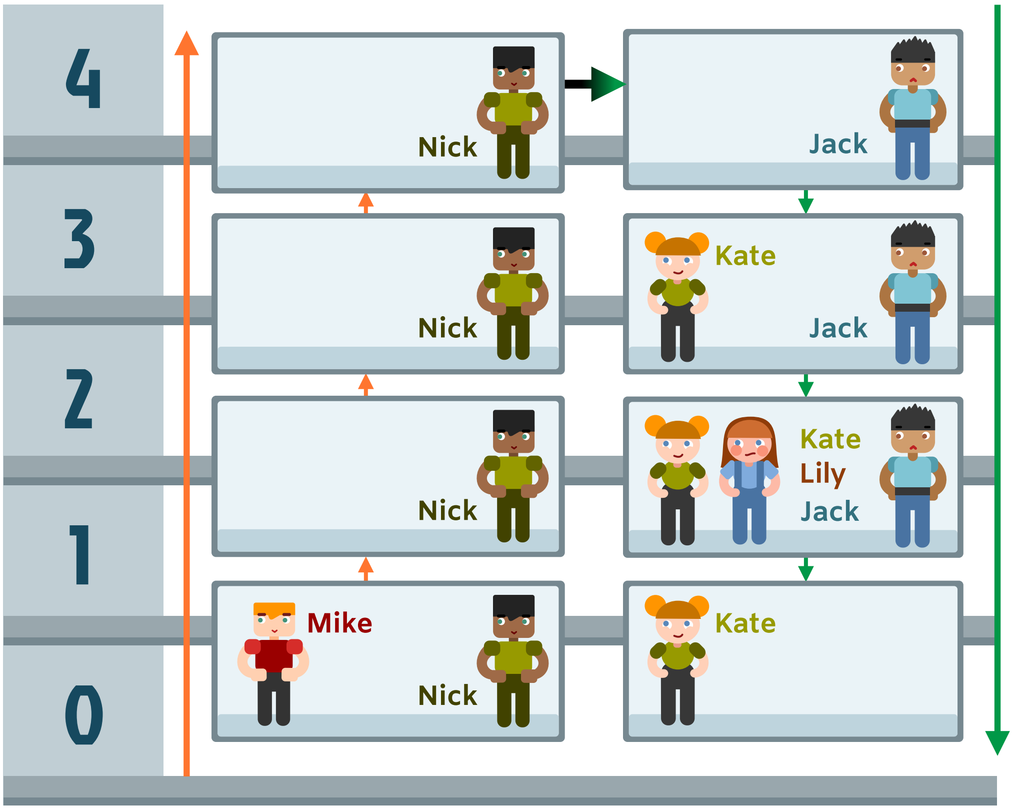
Ordering lists of data is a common problem in computer science. This question demonstrates an algorithm called a *bubble sort*. This algorithm makes passes through the data and swaps adjacent entries if it satisfies a certain condition - in this question, if the number on the right is smaller than the number on the left.

This is just one of many sorting algorithms that exist. The bubble sort is a great demonstration of these types of algorithms as it is one of the most simple. However, simple algorithms like this often take lots of time to sort lists as they become longer. Other more complex algorithms such as *quicksort* can order data faster than the bubble sort.

Lift

There is a lift at school. Nick , Lily , Kate , Mike  and Jack  all take a trip on the lift.

On this trip, the lift started from the ground floor (0), went up to the top floor (4), and then went back down to the ground floor. Along the way, it made several stops. The image below shows who was in the lift between each of the floors:



Question

Which of the following statements is true?

- Jack went from the top floor (4) to the ground floor (0)
- Mike was the only person to take the lift for one floor
- Two people left the lift at the first floor (1)
- Jack and Kate each took the lift to go three floors

Continued on next page

Lift – continued

EXPLANATION

Answer

Jack and Kate each took the lift to go three floors. Kate went from floor 3 to floor 0, and Jack went from floor 4 to floor 1.

Explanation

The statement *Jack and Kate each took the lift to go three floors* is true as Kate went from floor 3 to floor 0, and Jack went from floor 4 to floor 1.

Let us look at the other statements as well:

1. *A) Jack went from floor 4 to floor 0*. It is false because Jack went from floor 4 to floor 1 when the lift went back down. Between floor 1 and floor 0, only Kate was in the lift.
2. *B) Only Mike took the lift to go one floor only*. It is false because Lily did so, too. She went from floor 2 to floor 1, while Mike went from floor 0 to floor 1 with the lift going up.
3. *C) Two people left the lift at floor 1*. When the lift was going up, Mike left the lift at floor 1. When the lift was going down, Jack and Lily left at floor 1. Therefore, three people left the lift at floor 1.

BACKGROUND INFORMATION

In informatics we usually need to deal with data about some situation. In this task there were two types of information – one picture of a building with people waiting for the lift, and diagram in which we saw how people used it. Even though there were no implicit information about who was in the lift for three floors, who left it where, or who was the first or last one in it, we could simply deduce it by reading the diagram.

Diagrams representing the changes of a system are often parts of problem specification which have to be programmed. It's easier to use them compared to writing down all information we have. In diagrams like this one there are states (in this task a position of the lift and its direction) and transitions between them (who was in the lift between two states). With the help of transitions, we can say who was in the lift and when.

Reading the diagram is a skillset that can be used often in everyday life, e.g., as part of operating instructions for various home appliances.

This question comes from
South Korea

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Medium
Years 11+12

K-Pop Contest

A singing contest was held in Beaver Village with six singers in the final round.

Four experts were invited to judge them. Each expert gave points according to their own criteria.
The score table looks as follows:

Judges / Singers	Jin	Hope	Rosé	Jungkook
Anna	80	8	60	0
Betty	90	10	80	50
Carrie	85	7	90	100
Dennis	100	9	100	30
Elin	95	6	70	10
Frank	75	5	50	20

The organisation decided to rank the singers in each of the judge's lists according to the given scores as 1st, 2nd, 3rd, etc. For example, if a singer finished 4th, 2nd, 3rd and 6th on the judges' lists respectively, they would score a total of 4+2+3+6=15.

The winner of the contest is then determined by the smallest sum of all the rankings.

Question

Who is the winner?

Anna | Betty | Carrie | Dennis | Elin | Frank

Continued on next page

K-Pop Contest – continued

EXPLANATION

Answer

The correct answer is Dennis.

Explanation

First of all the organiser had to rank all six singers in each judge's list. It means that a singer who had the highest score got #1 ranking; a singer with the lowest score got #6 ranking. Assigned rankings are provided in brackets (in the table below). Next, the organizer had to add up each singer's rankings and to find the smallest sum. This condition is fulfilled by Dennis who became the winner.

Judges / Singers	Jin	Hope	Rosé	Jungkook	Sum of rankings
Anna	80(5)	8(3)	60(5)	0(6)	19
Betty	90(3)	10(1)	80(3)	50(2)	9
Carrie	85(4)	7(4)	90(2)	100(1)	11
Dennis	100(1)	9(2)	100(1)	30(3)	7
Elin	95(2)	6(5)	70(4)	10(5)	16
Frank	75(6)	5(6)	50(6)	20(4)	22

BACKGROUND INFORMATION

Machine learning, a hot topic these days, is about training a machine using a large amount of data with various characteristics. For example, suppose a machine learning model is trained using data with characteristics such as area of a house, number of rooms, distance from next school, age of house, distance from next shop, etc. to predict the price of a house. The units of each characteristic such as area, age, and distance are different, and the range of price will be quite different.

In this case, *normalisation*, a process of converting values so that all features have a similar influence, is required. Therefore, in machine learning, the normalisation process is important to ensure that all data injected into a model are reflected at the same scale (importance).

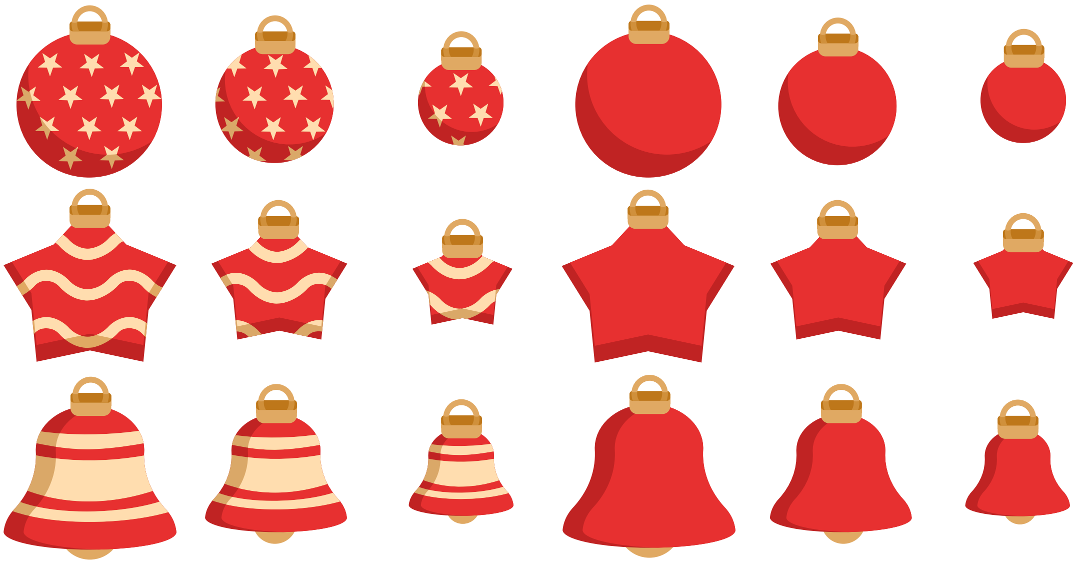
In addition, for very large data, *robustness* is required meaning that the calculation is not affected by outliers or input errors.

This question comes from
Slovakia

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Medium
Years 11+12

Decorations

Lara and Michaela have decorated their Christmas tree using 18 unique ornaments. The ornaments all have a shape (star, ball, bell), a size (small, medium, large) and sometimes a pattern.



Michaela has a favourite ornament and Lara has to figure out which one it is, by asking yes/no questions about the shape, size, and pattern of the favourite ornament.

Lara needed *exactly* four yes/no-questions before she knew with certainty that Michaela likes the medium sized star without a pattern the most.

Can you figure out which four questions Michaela asked?

Question

Drag the questions Lara may have asked next to the answers Michaela has given.
Note: there is more than one correct solution, you only have to provide one.

Is it a star?

Is it a bell?

Does it have a pattern?

Is it large?

Is it small?

Is it medium?

Is it round?

Lara's questions	Michaela's answers
	Yes
	No
	Yes
	No

Continued on next page

Decorations – continued

Answer

Here are four possible solutions:

Lara's questions	Lara's questions	Lara's questions	Lara's questions	Answers
Is it medium?	Is it medium?	Is it a star?	Is it a star?	Yes
Is it round?	Is it a bell?	Is it large?	Is it small?	No
Is it a star?	Is it a star?	Is it medium?	Is it medium?	Yes
(always as final question:) Does it have a pattern?				No

Explanation

Since there is only one way to ask about a pattern ('*Does it have a pattern?*') and the resulting favourite ornament has no pattern, one of the questions Lara asked was '*Does it have a pattern?*' and the answer to it was no. We have three questions left.

To guess the shape or size, we need one question (if we guess correctly right away) or two questions (if our first guess was wrong).

Let's first consider the possibility that Lara guessed the shape correctly the first time. Thus she had to ask '*Is it a star?*' and she got the answer yes. Then she asked about the size twice, answering no for the first time and yes for the second time. She could get a negative answer to either question '*Is it large?*' or '*Is it small?*'. The positive answer was to the question '*Is it medium?*'.

Now let's discuss the possibility that Lara guessed the size correctly the first time. Thus she had to ask '*Is it medium?*'. She then asked about the shape twice, getting the answer no for the first time and yes for the second time. She could get a negative answer to either question '*Is it round?*' or '*Is it a bell*?' and the positive answer to the question '*Is it a star?*'.

She had to ask the question '*Does it have a pattern?*', to which the answer was no, at the end, because one of the other questions needs no as an answer, followed by a yes, as shown above. If '*Does it have a pattern?*' was asked second, then Lara would know the ornament after 3 questions, and the 4th question would be redundant. Since the question states that Lara needed four questions before being able to determine with certainty what the favourite ornament was, this can't have been the case.

BACKGROUND INFORMATION

This is an example of a classification (categorisation) task, a very importnat problem in infomatics. In the classification task, we try to classify each object into a certain category based on its characteristic properties. In our case, the ornaments according to size, shape or pattern. In addition, in this task, each ornament has a different triple of characteristic properties, so we can uniquely identify each ornament.

Common classification problems in real life are classification of medical images, facial recognition, and email spam detection.

The task is similar to games like 'Guess Who?' or 'Twenty Questions'. The main principle of these games is that by each question we divide the set of possibilities into two groups, one with no satisfactory objects (which we can eliminate for the next turn) and the second one with potentially satisfactory objects (which we consider in the next turn). If these two groups are always the same size, then the process is analogous to the binary search algorithm.

This question comes from
Poland

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Medium
Years 11+12

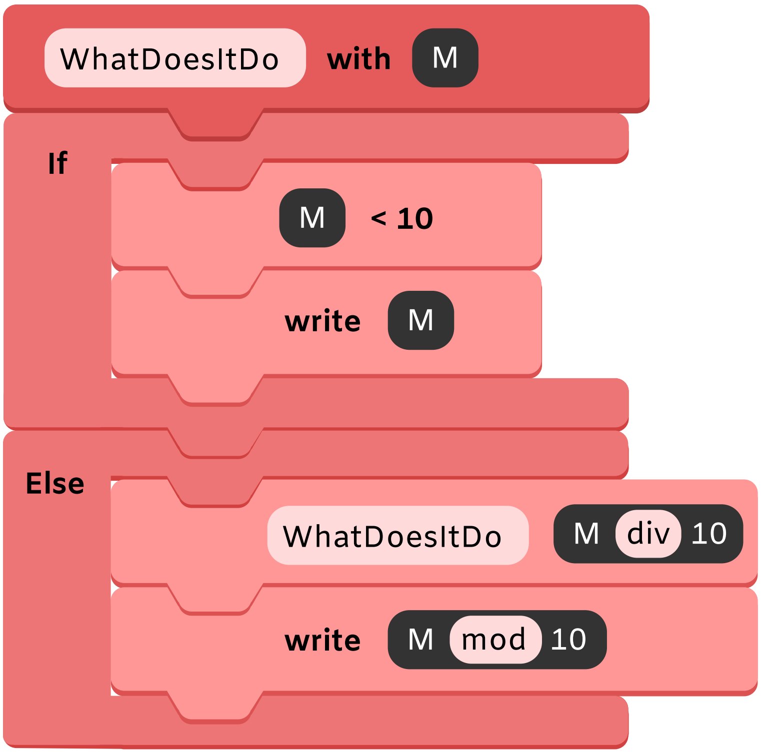
WhatDoesItDo

Beavers Gabe and Melissa are playing around with block code. They have figured out through trial and error that `div` takes the result of a division and rounds it down to a whole number, and that `mod` takes the remainder. For example:

90 `div` 7 is 12, and

90 `mod` 7 is 1.

Melissa has given Gabe the following function called *WhatDoesItDo*, which takes an integer *M* as input. The function first checks the size of M before deciding what part of the program to follow. The function continues until it reaches the end of the instructions.



Question

When *M* is 30241, what number will the function write?

Continued on next page

WhatDoesItDo – continued

EXPLANATION

Answer

The correct answer is 30241.

Explanation

Since the function includes itself as part of it, the integer M is processed through the function over and over again. The input is updated each pass through the function. This information and the respective output that is written by the function is shown in the following table. The *Function Level* refers to where that particular function is nested within previous functions. In other words, when 30241 is initally passed through the function, it is at level 1, however this leads to 3024 being passed through the function which is nested within the original pass. This new pass of the function is now level 2, and so on.

Function Level	Input	p(m<10)	m div 10	m mod 10	Output
1	30241	False	3024	–	–
2	3024	False	302	–	–
3	302	False	30	–	–
4	30	False	3	–	–
5	3	True	–	–	3
4	30	–	–	0	0
3	302	–	–	2	2
2	3024	–	–	4	4
1	30241	–	–	1	1

BACKGROUND INFORMATION

When we repeatedly need to ask the computer to do the same set of tasks over and over again, we will define this set of tasks as a *function* for easy reference in our program. This way, when we need to do this set of tasks, we can call the function instead of writing out the set of instructions over and over again.

Sometimes, when we define a function, we will refer to the function itself as part of the task which the function should do. We call this technique *recursion*. Recursion is often difficult for us humans to keep track of, but it is easy carried out by computers as at its core it is just following a well-defined algorithm to completion.

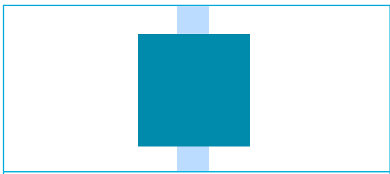
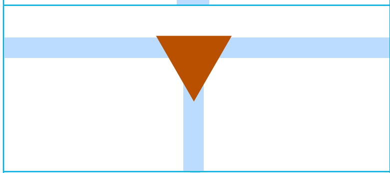
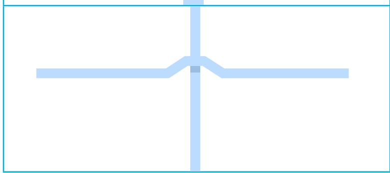
This question comes from
New Zealand

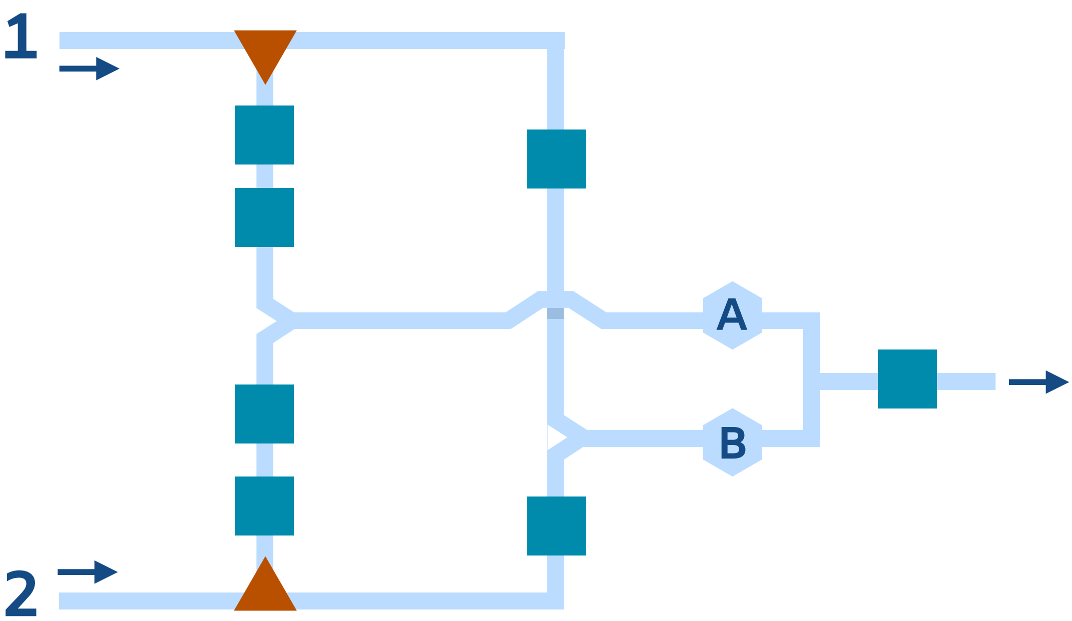
Years 3+4
Years 5+6
Years 7+8
Years 9+10 Medium
Years 11+12

Ice Cream Machine

In Beavaria, the ice cream shop “Baskin Loggins” owns a special ice cream machine. It has two ingredients that can be put into the machine to give the ice cream flavour: pistachio and vanilla. These ingredients flow through the tubes from left to right, to where the ice cream comes out.

The machine is built up with the following devices:

	This device changes the flavour – vanilla to pistachio, or pistachio to vanilla.
	If pistachio flavouring passes through this device, it will change its direction to the tube it is pointing at. Any other flavour than pistachio won’t change direction.
	At this place, the tubes do not join but pass over each other.



Question

For today, “Baskin Loggins” can only pour one kind of ingredient into both inputs 1 and 2. Either they pour vanilla into both inputs, or pistachio into both inputs.

Where do they have to place another device in the machine to get *pistachio ice cream* as the output no matter which flavouring was used as the input?

A | B | A and B | Neither A and B

Continued on next page

Ice Cream Machine – cont'd

EXPLANATION

Answer

A and B.

Explanation

The correct answer is they have to place a device at both point A and point B. A single device changes the flavour so two consecutive devices will change the flavour back to what was originally put in. Thus to have pistachio ice cream after pouring in the pistachio flavouring, the flow must go through an even number of devices:

- If they pour pistachio into input 1 and 2 to start with, both go through the left vertical route, and a device has to be placed at A to ensure it has gone through 4 devices (an even number) so the outcome is pistachio.
- If they pour vanilla into input 1 and 2 to start with, the ingredients will flow straight through the direction changing device and along the horizontal path. If they want to get pistachio from the machine, when vanilla has been poured in, there must be an odd number of devices, and this is achieved by placing a device at B.

BACKGROUND INFORMATION

The scheme that needs to be completed in this problem can be viewed as a *synchronising automaton*, which, regardless of the type of ice cream supplied to the input, always outputs pistachio ice cream.

It can also be viewed as a *logic diagram*. At the same time, one needs to pay attention to the fact that the task uses a *ternary logic* - which differs from binary as there are three possible states - since there may be vanilla ice cream in the pipe, pistachio ice cream, or the pipe may be empty. Accordingly, the device for changing the taste is somewhat different from the logical NOT - it changes the taste of ice cream varieties, but does nothing when there is nothing in the pipe: NOT (F) = T, NOT (T) = F, NOT (U) = U.

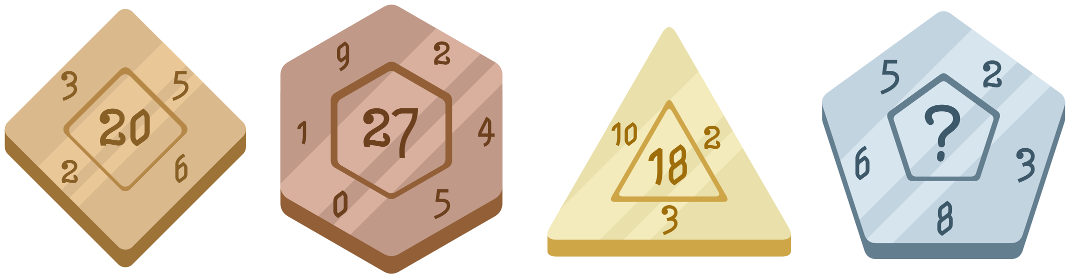
The device shown with a green dot has one input and two outputs. Therefore, it can be represented as the operation of two ternary logical devices, both of them will differ from the ternary NOT described above. One of them (denote it P) will work like this: P (F) = U, P (T) = T, P (U) = U. The second device R is as follows: R (F) = F, R (T) = U, R (U) = U. Computers can be built on ternary logic - in fact, while modern computers function on binary, ternary computers were really built. In some aspects related to information theory, such computers are better than binary computers.

This question comes from Uruguay

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Medium
Years 11+12

Secret Number

In Bebravia payments are made with special coins. Each coin has its own value written in the centre.



A citizen of Bebravia has the four coins shown above, but the value at the centre of one of the coins is rubbed out. However, each coin's value can be worked out using the same set of rules. From this, the citizen was able to determine the missing value.

Question

What is the missing value?

22 | 23 | 26 | 29

EXPLANATION

Answer

29.

Explanation

The value of each coin is calculated as following: sum of the numbers surrounding the central coin value plus the number of sides the coin has. In this task there are four different coins:

Square: 3 + 5 + 6 + 2 + 4 = 20
Hexagon: 1 + 9 + 2 + 4 + 5 + 0 + 6 = 27
Triangle: 10 + 2 + 3 + 3 = 18
Pentagon: 5 + 2 + 3 + 8 + 6 + 5 = 29

A student attempting this might at first try to find a relationship between the numbers around the outside and the value in the middle and, after failing to find a simple pattern (and remembering that Bebras tasks do not require any mathematical knowledge beyond simple numeracy) go back to the task and check they have extracted all of the data given. Looking for more data should enable the student to notice that the coins have different numbers of sides and then quickly work out the pattern.

BACKGROUND INFORMATION

The value of each coin is formed by two parts. It is important to figure out that the coin value (the number in the centre) is obtained from the sum of numbers located on each side of the coin plus the number of sides the coin has. All value calculations follow the same pattern.

Patterns and pattern recognition are used in engineering, computing, mathematics and are related to physical or abstract objects. Pattern analysis can be simple or sophisticated. Patterns can be obtained from the processes of segmentation, extraction of characteristics and description where each object is represented by a collection of descriptors. The purpose of the analysis is to extract data that allow us to recognise properties and mark regularities among sets of objects. In computer science, a software design pattern is a common template solution that helps speed up the development of computer programs.

Ada’s Marble Machine

Ada the engineer is working with a set of marbles that have four properties, but only certain combinations are permitted as shown below:

Size	Colour	Material	Design	Possible?
Small	Red	Stone	Glitter	No
Small	Red	Stone	Mosaic	No
Small	Red	Metal	Glitter	Yes
Small	Red	Metal	Mosaic	No
Small	Yellow	Stone	Glitter	Yes
Small	Yellow	Stone	Mosaic	Yes
Small	Yellow	Metal	Glitter	Yes
Small	Yellow	Metal	Mosaic	Yes
Large	Red	Stone	Glitter	No
Large	Red	Stone	Mosaic	No
Large	Red	Metal	Glitter	No
Large	Red	Metal	Mosaic	No
Large	Yellow	Stone	Glitter	No
Large	Yellow	Stone	Mosaic	Yes
Large	Yellow	Metal	Glitter	No
Large	Yellow	Metal	Mosaic	No

Ada also provides a flowchart that can correctly check whether the given marble belongs to the set described by the table. Only one of the four flowcharts shown below was made by Ada.

Continued on next page

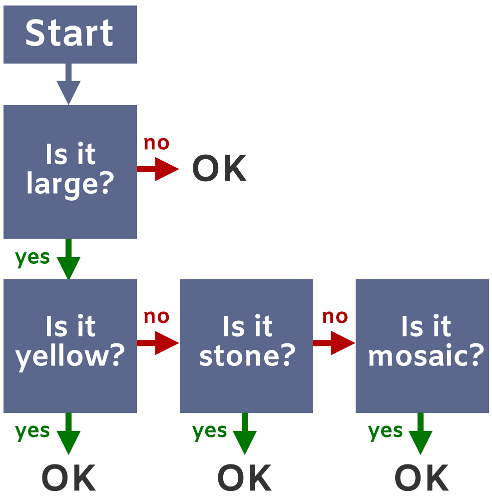
This question comes from Australia

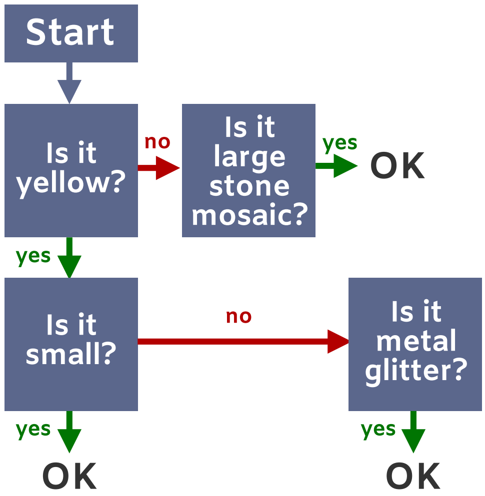
Years 3+4
Years 5+6
Years 7+8
Years 9+10 Hard
Years 11+12

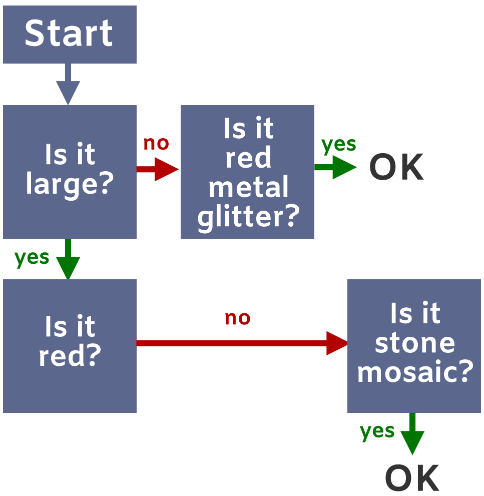
Ada’s Marble Machine - cont’d

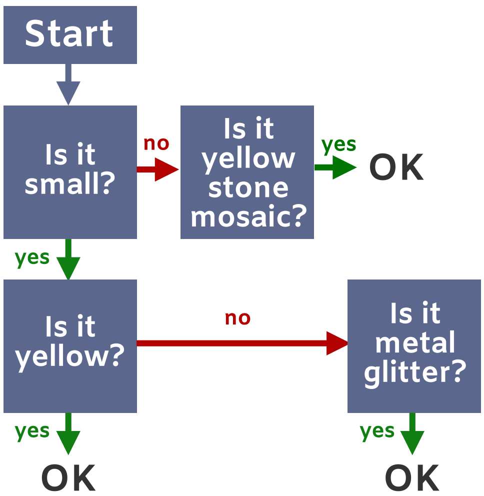
Question

Which of the following four flowcharts correctly checks for any given marble if it belongs to the set described by the table?







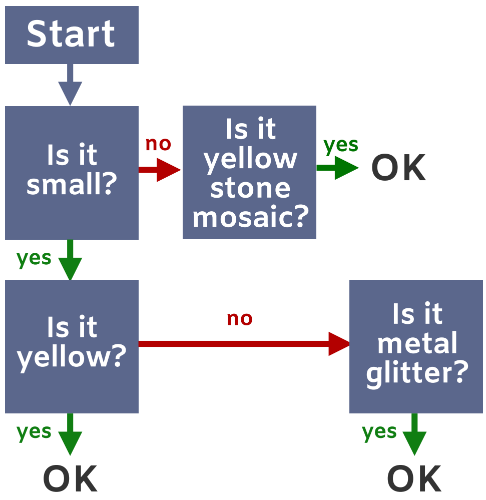


Continued on next page

Ada’s Marble Machine - cont’d

EXPLANATION

Answer



Explanation

- The first flowchart accepts the marble after “is it large? / no”. That means all small marbles are accepted which is not correct.
- The second flowchart incorrectly identifies “large red stone mosaic” and “large yellow metal glitter” as a valid combination. The flowchart successfully finds the valid “small yellow” combinations.
- The third flowchart correctly finds the valid combinations “small red metal glitter” and “large yellow stone mosaic” but it fails to identify the valid “small yellow” combinations.
- The final flowchart correctly finds the valid “small red metal glitter”, “large yellow stone mosaic” combinations and additionally the valid “small yellow” combinations.

BACKGROUND INFORMATION

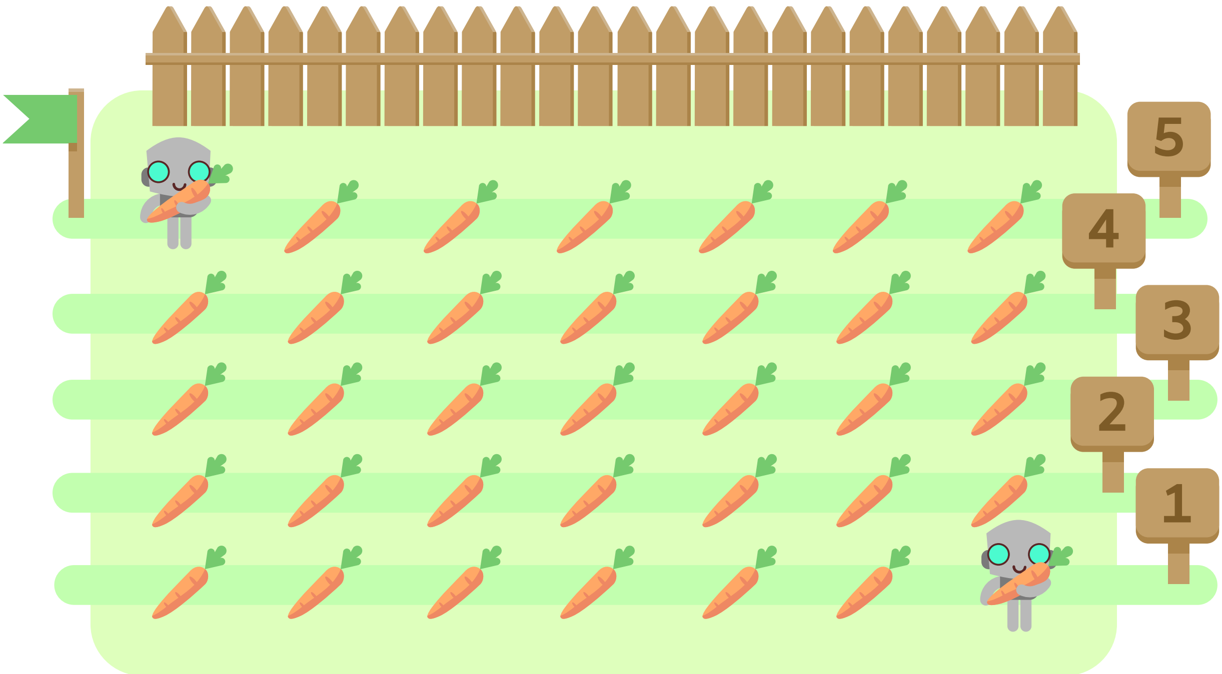
Only a subset of possible combinations of properties were considered valid in the table from this question. For those combinations, the term “well-formed” is used as opposed to “malformed” for the incorrect ones. *Flowcharts* are a possible notation to formalise ‘well-formedness’ for a particular set by describing the process of validating a given combination. In our example the set was very small, and it was possible to show the list of valid elements, but for very large sets this is no longer possible and formalised methods to check for validity are necessary. The flowchart could be incorporated in a program to automatically run checks on large numbers of elements quickly.

The character in this question is named after Ada Lovelace, often referred to as the world’s first computer programmer.

Smart Farm Robot

On Beaver Island, there is a carrot farm. The farm has 5 rows with 7 equally spaced carrots in each row, as shown below. Last year, the beavers started using a robot to harvest the carrots. The robot follows instructions using the following commands:

- P - Look for a carrot in the same row and pick it up.
- F - Go forward in the same row.
- D - Go down to the next row with a smaller number.
- U - Go up to the next row with a bigger number.
- R - Turn right 90°.
- L - Turn left 90°.



If the robot needs to repeat the instruction more than once, it can be represented as number before the instruction. For example:

- 3F means the robot will move forward 3 spaces in the same row.

A set of instructions can also be repeated in this way using parenthises. For example:

- 3(F P) means the robot will repeat the instructions 'move forward one space and then pick up a carrot in that space' three times.

The robot can only move in the direction it is facing - that means the robot can only follow an F command if it is facing left or right along a row, and can only follow a D or U command if it is facing down or up respectively.

Question

A single robot starts in the top-leftmost space (next to the green flag) and is facing towards the sign with the number 5 on it. Which set of instructions will get the robot to harvest all carrots?

- 2(P6 (F P) R D R) (P 6(F P) L D L) 7P
- R 3(P 4(D P)L F L)(P 4(U P) R F R)
- 2(P 6(F P) R D R P 6(F P) L D L) P 6(F P)
- 5(7P D) 7P

Continued on next page

Smart Farm Robot – continued

EXPLANATION

Answer

2(P 6(F P) R D R P 6(F P) L D L) P 6(F P).

Explanation

Only the instruction *2(P 6(F P) R D R P 6(F P) L D L) P 6(F P)* will allow the robot to harvest all the carrots. To see that this set of commands is correct, they can be carried out fully as shown below.

1. To harvest all carrots at the farm, start with Row 5 and pick up the 7 carrots from left to right. We write the instruction P 6(F P) - pick up the carrot in the starting plot, and then repeat the instruction ‘move forward one space and harvest the carrot’ six times. This will instruct the robot to pick all the carrots in Row 5.
2. To go to Row 4, the robot should turn right 90°, then go down. And the robot has to turn right another 90º before it can pick up carrots along Row 4. Therefore, we write the instruction R D R.
3. In Row 2, the instruction P 6(F P) makes the robot pick up all 7 carrots along the row, as in instruction 1.
4. To go to Row 3, the robot should turn left 90°, then go down. In order to move along Row 3, the robots has to turn left 90° again. We write the instruction L D L.
5. The previous 4 sets of instructions makes the robot pick up all the carrots along the rows. We can repeat these instructions for the next two rows, namely Row 3 and Row 2. Therefore, we group the previous instructions together with () and put the number 2 at the beginning, forming 2(P 6(F P) R D R P 6(F P) L D L).
6. And last, the instruction P 6(F P) makes the robot pick up all the carrots in Row 1.

Therefore, the instruction should be: 2(P 6(F P) R D R P 6(F P) L D L) P 6(F P)

The other options are either incorrect (for example, asking the robot to move up, down, or forward while facing the wrong direction), or does not harvest all the carrots.

BACKGROUND INFORMATION

Looping is a type of instruction used in all program languages to execute a sequence of statements multiple times. The set of instructions will repeat a certain number of times or until it reaches the specified condition. Loops are useful as it allows complex sets of instructions to simplified into understandable representations. This allows programmers to find bugs in the code more easily, as well to both communicate how the code works and to find points within the code that can be optimised.

This question comes from the Philippines

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Hard
Years 11+12

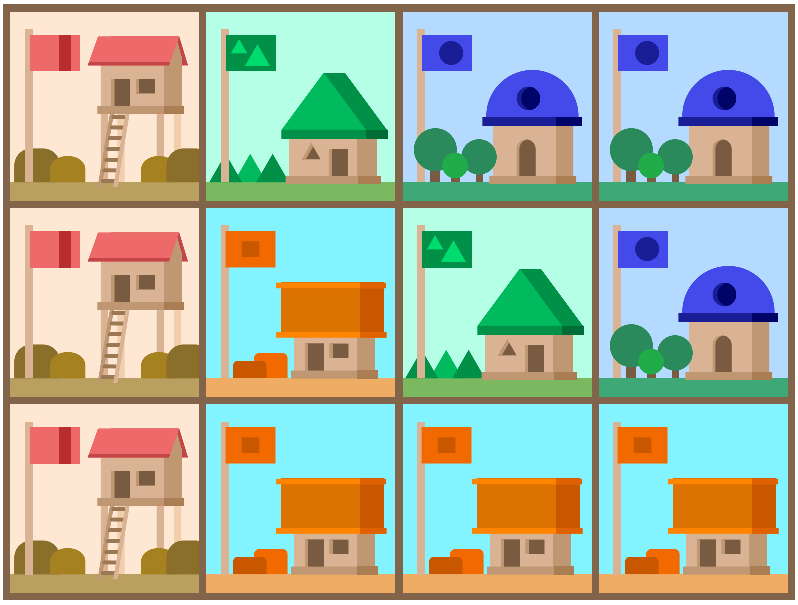
Unification

In ancient Beavaria, there lived four tribes consisting of several villages. Each tribe had their own flag as shown in the picture below.

One day, the tribes decided to unify. However, in order to not cause chaos, it was decided that only two tribes can be unifying at the same time.

The time needed to unify two tribes, in months, is equal to the total number of villages in these two tribes.

After this, the two tribes become one single tribe, and the unification process is repeated until there is only one unified tribe remaining.



Question

What is the minimal amount of months needed for the tribes to unify?

23 | 24 | 25 | 26 | 27

EXPLANATION

Answer

24.

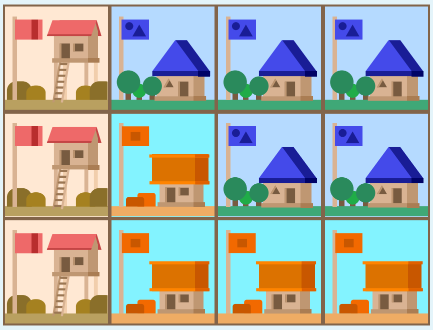
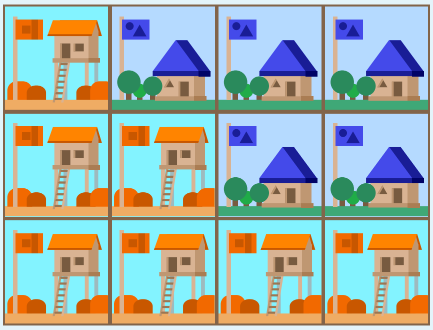
Explanation

The optimal strategy to minimise the total number of months needed to unify all the tribes is to minimise the number of times each village is included in the unification processes. This can be done by merging the largest tribes last, as the largest tribes with the greatest number of villages will then only be added the least amount of times. In order to do this, each unification step should happen between the two tribes with the fewest villages.

Continued on next page

Unification – continued

This is illustrated in the table below:

1. Green triangles have the least number of villages, so they will be chosen for unification first. Since there are two tribes that have 3 villages, we can choose to unite either one, for example, green triangles and blue circles. After unification they can be called blue triangles and circles.	2. The tribes that now have the least number of villages are orange squares (4) and red stripes (3). After unification we can call them orange stripes and squares.	3. Lastly, 5 blue triangles and circles and 7 orange stripes and squares villages unite into one large orange shapes tribe.
		
This takes 5 months and results in 3 red stripes, 4 orange squares, and 5 blue triangles and circles villages.	This takes 7 months and results in 5 blue triangles and circles, and 7 orange stripes and squares.	This takes 12 months.

Therefore, the minimum number of months to unify all four tribes from the land of Beavaria is 5+7+12=24.

BACKGROUND INFORMATION

This challenge is an example of an *optimisation problem*, a task whose goal is to come up with a strategy that maximises or minimises a certain quantity, subject to some constraints. Optimisation problems are ubiquitous in our everyday lives: finding the shortest route to a destination, creating a schedule that accommodates the most number of non-overlapping activities, and so on. There are several ways to approach solving an optimisation problem, and these include *greedy algorithms*.

Greedy algorithms rest on the assumption that making the best choice at each stage (*local optimum*) will result in the best final outcome (*global optimum*). In this problem, this assumption is satisfied: tribes have to minimise the number of months for each unification in order to minimise the number of months for the entire unification process.

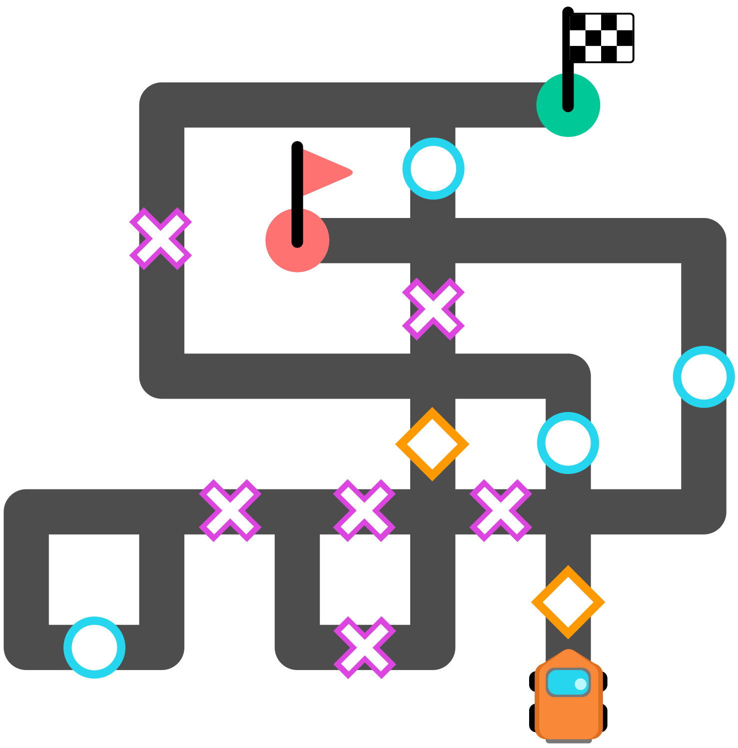
It must be emphasised, however, that the greedy paradigm is not a universal solution to all types of optimisation problems. Nevertheless, it usually provides a decent approximation within a reasonable time.

This question comes from Czechia

Years 3+4
Years 5+6
Years 7+8
Years 9+10 Hard
Years 11+12

Symbol Reading Robot

A robot starts from the position shown below and moves along the lines. There are three symbols ◇, ○ and ✕ on the lines that decide the direction it should take at the next intersection. The robot must not reach the flag.

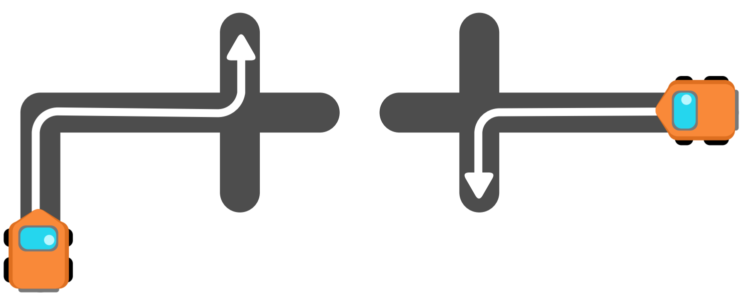


Each symbol has a different meaning and could mean:

	turn left at the next intersection, or
	*turn right at the next intersection, or*
	*go straight at the next intersection*

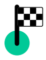
Unfortunately, we do not know which symbol means what.

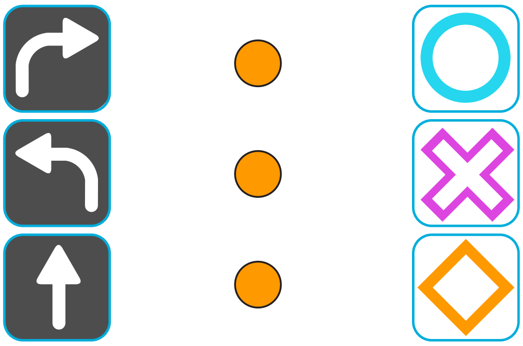
The meaning of the symbol remains the same regardless of the direction the robot is moving. For example, the arrow in the picture below shows how the robot would turn, coming from either direction, if a triangle symbol meant "turn left".



Symbol Reading Robot – cont’d

Question

Help the robot reach  by assigning correct meaning to the symbols.

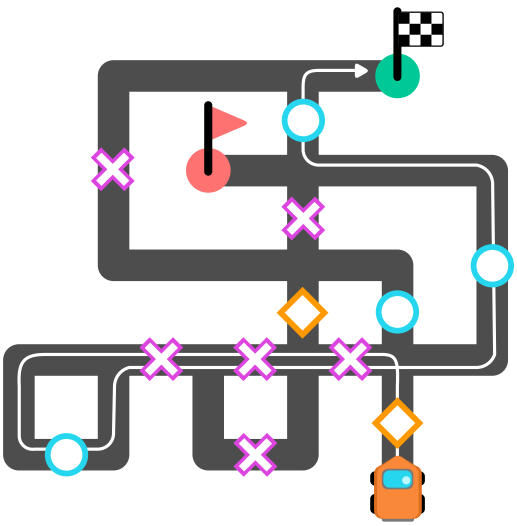


EXPLANATION

Answer

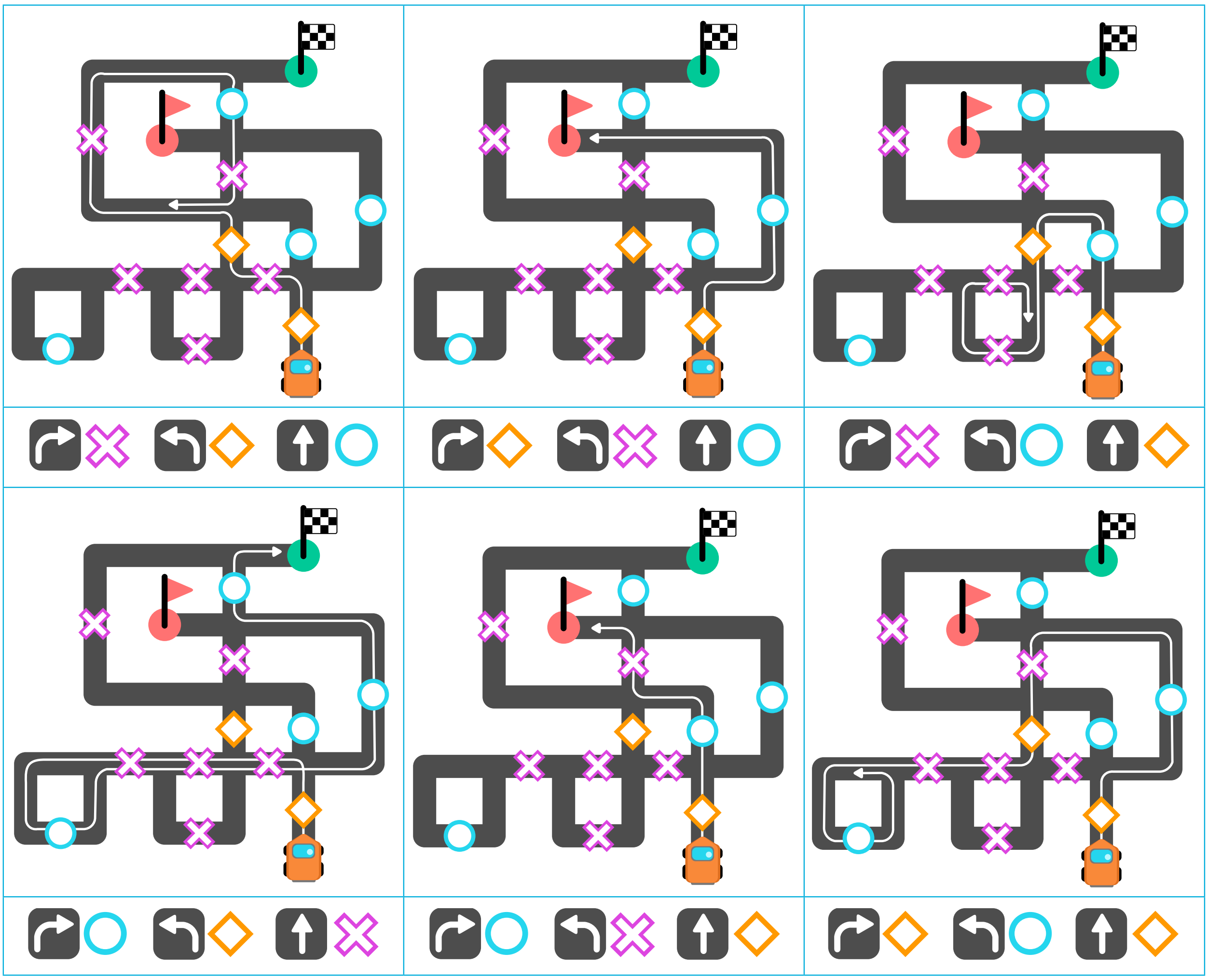
Explanation

The picture shows robot’s walk:



Symbol Reading Robot – cont’d

We can use different strategies to solve the problems. One of them is going through every possibility. In this task there are only 6 different ways to interpret the symbols (the pictures show all of them) and only one of them leads to :



Another strategy is to trace the path the robot could take and assign meaning to symbols as we follow the path. If the robot reaches the red flag or enters a loop, or the meanings cannot be applied to symbols consistently, we retrace to the previous step (in this case intersection) and try another path.

BACKGROUND INFORMATION

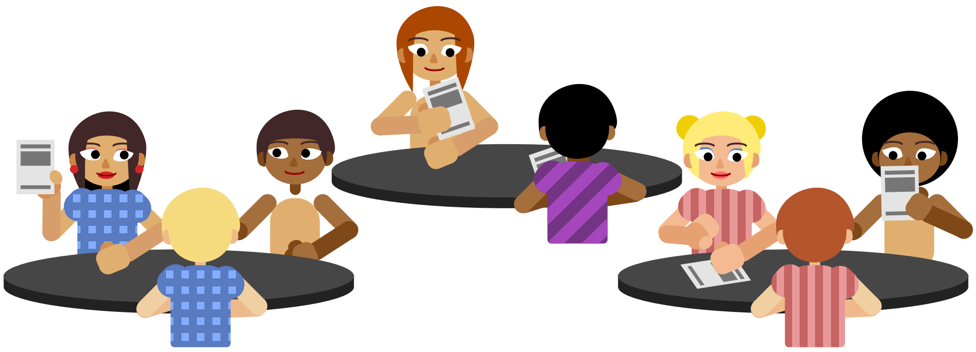
The first strategy proposed in the explanation is called *brute force*. This means to consider and check every possibility until one finds the desired result. Sometime there can be many possibilities and considering all of them could be very time consuming and therefore one need to find other strategies.

The second strategy is called *backtracking*. In this technique one incrementally builds candidates to the solutions, and abandons a candidate as soon as one determines that the candidate cannot possibly be completed to a valid solution. The advantage of backtracking with respect to brute force is that you do not have to reconsider the new candidate solutions from the beginning, since you go back only to the step where you made your last choice and continue on from there.

For the given problem brute force may work better, as we have few variables (symbols) and few paths for the robot to take. However in general for larger complex problems, when the number of variables increase and the paths to explore are more, backtracking provides better and elegant solution. Puzzles like Sudoku can be elegantly solved using backtracking.

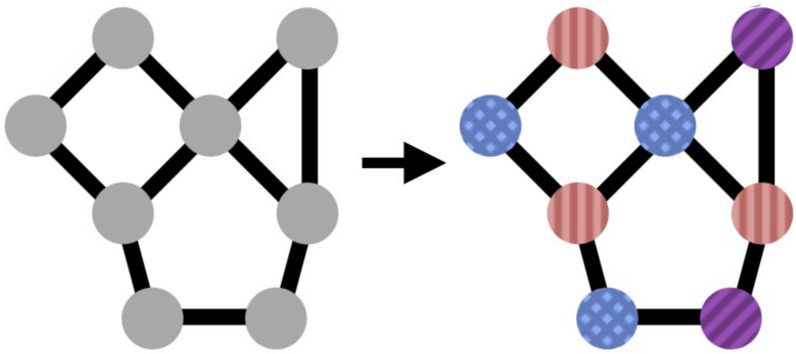
Quiz Night

Eight people usually form three quiz teams:



The teams are organised so that each player knows all of the other players on their team. They also know some of the players on the other teams. Who knows who can be represented by a graph.

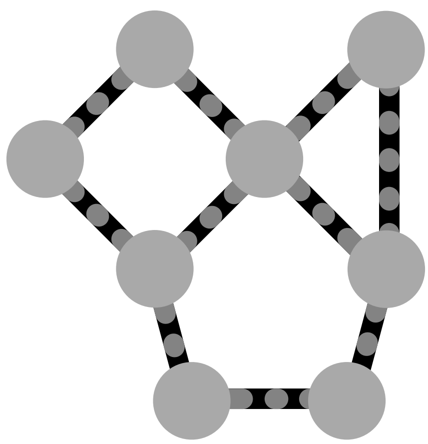
In the graphs below, circles represent people. If there is a line between two people this means that they **do not know each other** and so cannot be on the same team. The graphs can be helpful when assigning teams by colouring in the circles, for example:



Unfortunately, one of the tables is broken tonight, so only two quiz teams can be formed.
At the moment, two teams cannot be formed unless two people are introduced to each other.
This introduction can be shown by removing a line on the graph. But who should be introduced?

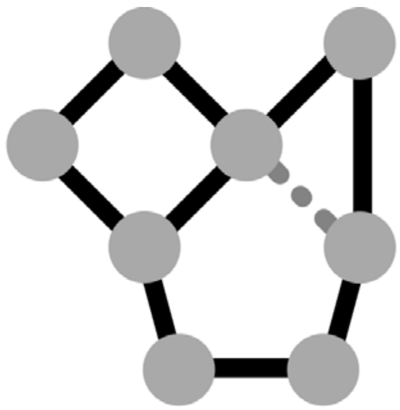
Question

Select one line from the graph below that, when removed, allows two teams of four to be formed.



Quiz Night – continued

Answer

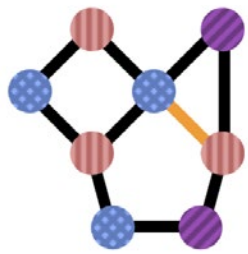


EXPLANATION

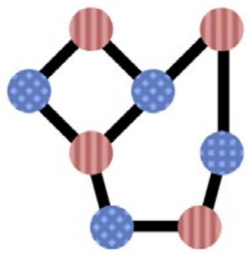
Explanation

Introducing two people means deleting an edge. We need to delete an edge so that two colours are enough to colour all the vertices (people) but no two vertices of the same colour are connected by an edge.

The only possible option is the edge marked in orange below.

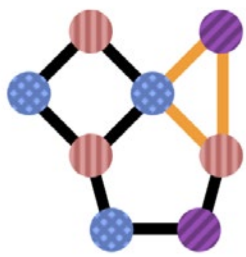


After deleting this edge, we can colour the graph with two colours as shown below right.



To test that deleting this edge is the only possible choice, we need to consider both the triangle in the upper right and the pentagon on the bottom.

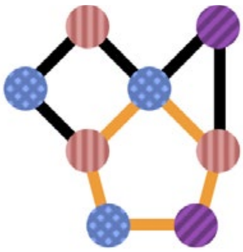
First consider the triangle in the upper right:



If any edge outside of this triangle is deleted, we still need three colours just for the three vertices of that triangle. So one of these three edges needs to be deleted.

Quiz Night – continued

Now consider the pentagon on the bottom:



If any edge outside of this pentagon is deleted, then it is impossible to colour all five of its vertices with only two colours. To test this, we can cycle clockwise through the five vertices, alternating colours for each one. But when we reach the last vertex, it will have the same colour as the first vertex because the number of vertices in the cycle is odd.

Therefore we need to delete an edge that destroys both the triangle in the upper right and the pentagon on the bottom at the same time. There is only one edge that is shared by both shapes, leading us to the only possible answer.

BACKGROUND INFORMATION

Many real-world problems can be reframed as coloured vertices on a graph. An example is a graph where the vertices are students and an edge between two students shows that they can't be placed in the same group. If we colour the vertices with k colours, this can be seen as assigning every student to one of k groups. Such a colouring is *proper* if any two vertices directly connected by an edge have different colours. Often, we just say colouring when we mean a *proper colouring*. An edge is sometimes called *critical* if deleting it makes a *proper colouring* with fewer colours possible. In the example, this means that if the corresponding two students are introduced and can then work together, then having fewer groups becomes possible.

Bebras Challenge 2022 Round 2

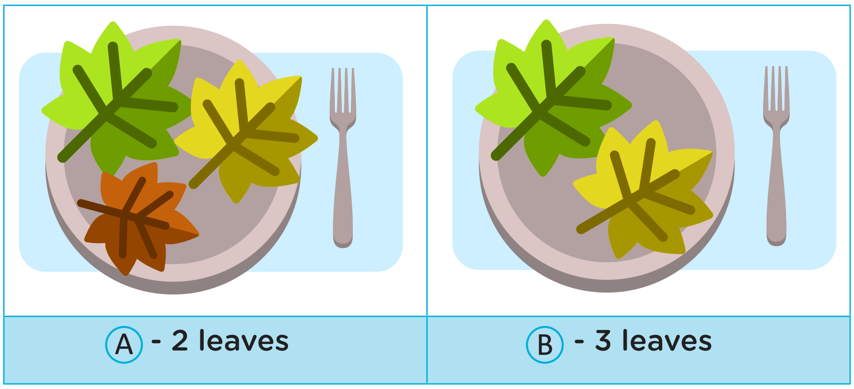
Years 11+12

This question comes from South Korea

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Easy

Taking Leaves

Beavers Miley and Jin take turns playing a game of "taking leaves". As shown in the picture below, leaves are piled on two plates A and B. When it is a beaver's turn, they must take 1 or more leaves from any one plate. The winner of this game is the beaver who takes the last leaf.



Question

Choose the starting move where Miley can win regardless of what her opponent does.

- Miley starts by taking 2 leaves from A
- Jin starts by taking 1 leaf from A
- Miley starts by taking 2 leaves from B
- Jin starts by taking 1 leaf from B

EXPLANATION

Answer

Jin starts by taking 1 leaf from A

Explanation

Miley and Jin will try to take the last leaf to win the game. In order for Miley to take the last leaf, one plate has to be empty already. This can happen only if Jin takes the last leaf from any one plate right before Miley. Therefore, Miley has to play in such a way that Jin would be forced to take the last leaf from one of the plates.

Of the four situations presented in this task,

1. If Miley first takes 2 leaves of plate A, Jin can win the game by taking all the 3 leaves from B.
2. If Miley first takes 2 leaves from B, Jin can take one leaf from A thus forcing Miley to take the last leaf from one of the plates and Jin wins.
3. If Jin starts by taking one leaf from A, Miley can respond by taking two leaves from B and leaving Jin forced to take the last leaf from one of the plates. Miley then empties the remaining plate and wins.
4. If Jin starts by taking one leaf from B, Miley has to take one leaf from one of the plates. Then Jin takes one leaf from the other plate. In this way Miley has to take the last leaf from one of the plates and loses the game. We can also see, that this starting move by Jin is the best move for the player starting the game.

So, the correct answer is 3.

Continued on next page

Taking Leaves – continued

BACKGROUND INFORMATION

This task is a variation of the famous *Nim game*, one of the first-ever computerized games. The computer that was designed for this game in 1940, called Nimatron, is considered the first gaming computer in the history of computing. This is one of the games, like tic-tac-toe, where it is impossible to lose if you know the winning strategy.

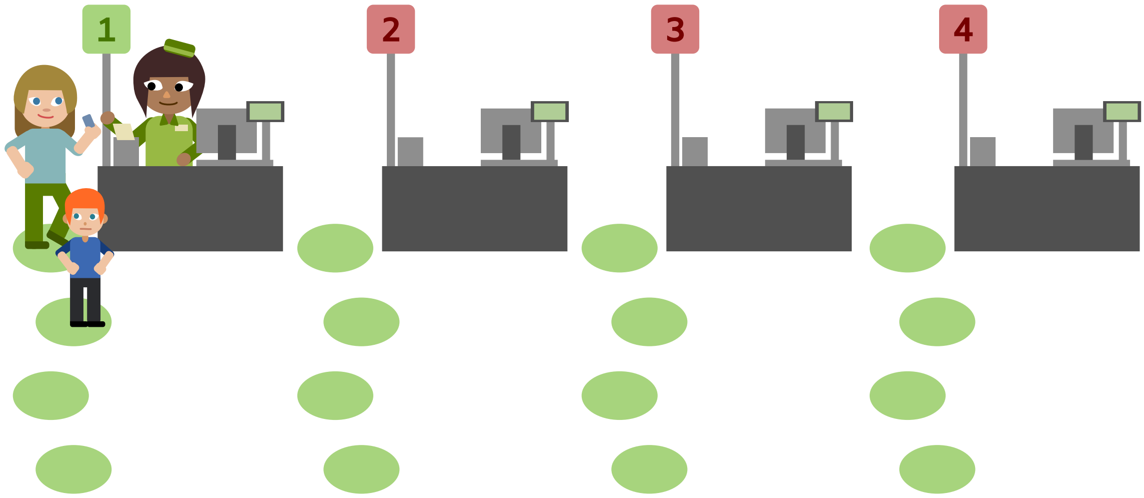
It can be challenging to arrive at a winning strategy in games like this by considering all possible cases, especially if the situation is more complicated and involves more plates and leaves. To reduce the complexity of this (and other similar problems) it can be useful to solve the more simple cases first and then reuse the answers later in more and more complicated cases, until you arrive at the answer. In computer science this solving strategy where we reuse answers of sub-problems to solve the main problem is called *dynamic programming*. Programmers often use dynamic programming to substantially increase the effectiveness and reduce the running time of their programs. In this task we also use a similar idea by recognizing what situations lead to winning or losing. If we later arrive at these situations, we don't have to repeat the same steps.

This question comes from Pakistan

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Easy

Shop Counters

A shop has four billing counters numbered 1, 2, 3, 4. Each counter can have a queue of at most 4 customers, including the customer being served. Each counter can serve one customer at a time. It takes 2 minutes to serve a customer.



When a customer wants to pay, they join the end of the queue at the first counter where the queue is not full. They try the counters in ascending order - first counter 1, then counter 2, etc.

If there is no space available at any of the counters currently open, a new counter opens and the customer joins the queue there. However, it takes 1 minute to set up a counter, so it takes 3 minutes to serve the first customer at a newly opened counter. Each following customer will be served in 2 minutes as usual.

At any given time, if there are customers who leave their queue after being served at the same time that there are new customers who want to join a queue, you can assume that the served customers leave first and create an empty space in their queues where new customers can join.

Question

Just after the store opens, the queues empty and only counter 1 is open. 12 customers arrive at the queues, two at a time each minute (two customers arrive initially, another two after 1 minute, etc.). How long does it take to serve them all?

12 minutes | 11 minutes | 13 minutes | 8 minutes

EXPLANATION

Answer

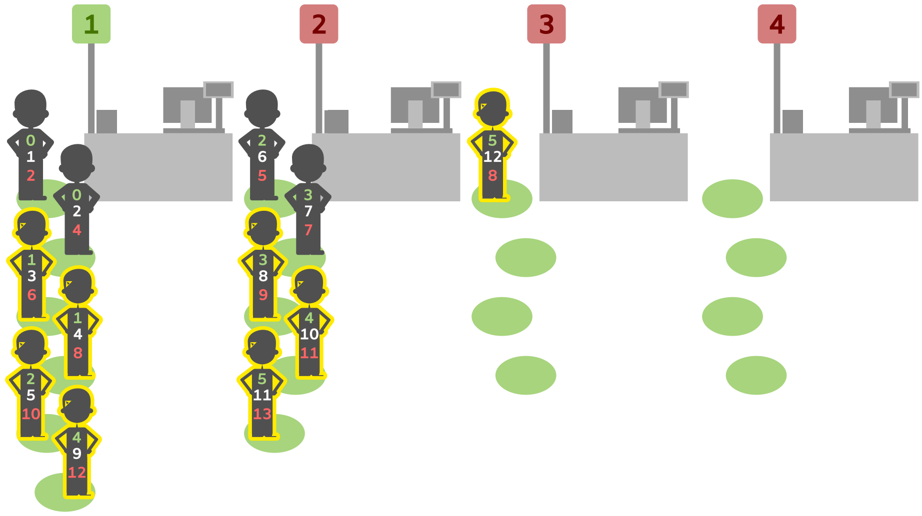
The correct answer is 13 minutes.

Continued on next page

Shop Counters – continued

Explanation

In the following picture the time (in minutes) are listed for when each customer (numbered in the middle, white) arrives at the counter (in green, top) and when they leave (in red, bottom):



The leaving times in each counter queue is predictable, as each customer leaves 2 minutes after the previous one. Th leaving times for each customer (shown in red, bottom) need to be tracked in order to solve the problem, which can be computed for each customer when they arrive.

After 2 minutes, when 2 new customers arrive, customer 1 can be removed from counter 1 (as they have finished being served) and one new customer joins the back of the queue at counter 1. This new customer will finish being served 2 minutes after the customer in front of them in the queue (8+2=10), while the other new customer goes to a newly opened counter (counter 2) and will be served 3 minutes after the current time (2+3=5).

After 5 minutes, the queues will consist of the customers highlighted in yellow. However, the timer continues until the last customer is served, which will be the last customer from the second queue, at 13 minutes. Thus, the correct answer is 13 minutes.

For some of the other options, 12 minutes is the time the last customer finishes in the first queue, while 8 minutes is the time the last customer finishes in the third queue.

BACKGROUND INFORMATION

Cloud services such as Google Cloud, Amazon Web Services, Microsoft Azure, etc., scale up the availability of computing resources dynamically, based on the usage requirements of customers. This way, customers only pay for the resources that they need. This flexible allocation of resources is called *dynamic scaling*.

This task illustrates a simple example of dynamic scaling. Here the resources are billing counters. Keeping a billing counter open requires paying one more staff member, for instance, so counters are opened on demand as the number of customers grows. This keeps down the cost when demand is low.

In practice, dynamic scaling also involves scaling down when demand decreases. In the context of this task, this would mean a strategy for closing counters when the number of customers starts to decrease.

Sorting Beavers

A class in Beavaria High School has seven beavers in it. Each beaver is given a flag with a number on it. They are seated in a row behind one another. In the beginning, the beavers are sitting randomly as shown in the picture.

The class's teacher wants to sort the beavers into ascending order, from number 1 at the front to number 7 at the back. They can only be sorted using swapping operations, where in each swap exactly two beavers exchange places. For example, when beavers 3 and 1 swap it means that beaver 3 goes to 1's place and 1 goes to 3's place.

Question

Given how the beavers are currently sitting, what is the minimum number of swaps needed to obtain the desired order?

3 swaps | 4 swaps | 5 swaps | 6 swaps

EXPLANATION

Answer

5 swaps.

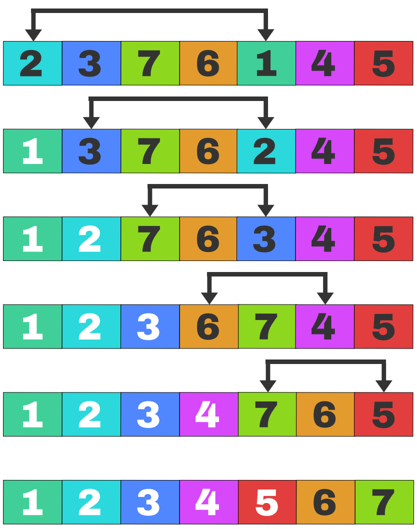
Explanation

The correct answer is 5 swaps, since we have to swap two different beavers 5 times as shown below.

This problem can be solved using a selection sort algorithm. This algorithm divides the input list into two parts: the sublist of items already sorted, which is built up from left to right at the front (leftmost part) of the list, and the sublist of items remaining to be sorted that occupy the rest of the list. Initially, the sorted sublist is empty and the unsorted sublist is the entire input list. The algorithm proceeds by finding the smallest (or largest, depending on sorting order) element in the unsorted sublist, exchanging (swapping) it with the leftmost unsorted element (putting it in sorted order), and moving the sublist boundaries one element to the right.

Sorting Beavers - continued

With the first order (the order seen in the body of the task), the lowest unsorted number (the number 1) is swapped with the number of the first place of the list (the number 2). In the next step, starting from the second place of the list, the lowest unsorted number (that is, the number 2) is swapped with the number of the second place of the list (the number 3). Following this algorithm, 5 steps are needed as shown below:



Given that only direct swaps can be made to sort the beavers, 5 swaps is the minimum number of swaps needed to have the beavers sitting in the number order of their flags.

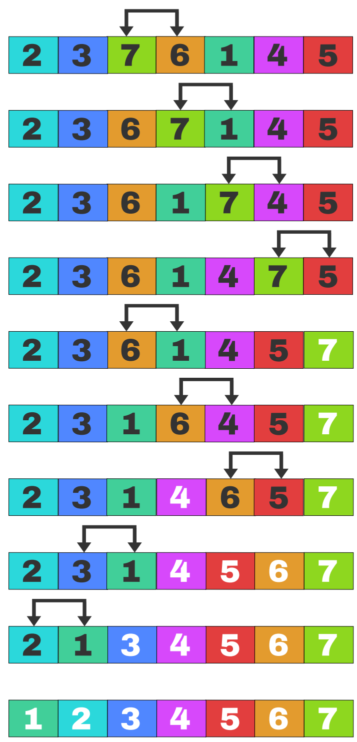
BACKGROUND INFORMATION

Sorting algorithms allow us, as their name says, to sort information in a special way based on a sorting criterion. In computer science, data sorting plays a very important role, either as an end in itself or as part of other more complex procedures. Many techniques have been developed in this field, each with specific characteristics, and with advantages and disadvantages over the others.

The *selection sort* algorithm improves on the *bubble sort* algorithm by making a single swap for each pass through the list. In order to do this, a selection sort looks for the lowest value as it makes a pass and, after completing the pass, puts it in the correct location. As with bubble sorting, after the first pass, the lowest item is in the correct location. After the second pass, the next lower is in place. This process continues and requires, in general, n-1 passes to sort the n items, as the final item must be in place after the (n-1)-th pass.

Due to the reduction in the number of trades, selection sort usually runs faster than bubble sorting, although for other sorting problems, it may not be the faster algorithm (there are a lot of algorithms for sorting “things”).

If we compare selection sort with the bubble sorting, we can see that the selection sort (detailed in the answer explanation) is faster.



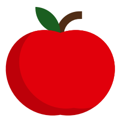
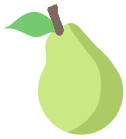
Fruit Stack

A family of four prepares breakfast for the next day. They pile up four boxes, each filled with a different fruit:

apple, pear, orange, or strawberry.

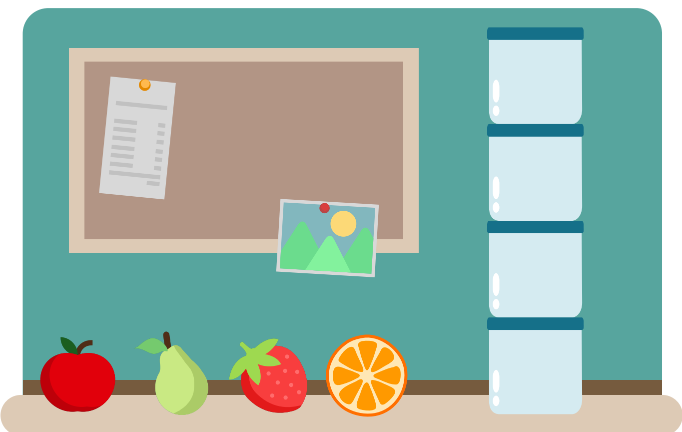
As they are sleepy in the morning, they all just grab the box off the top of the pile. They do not know in which exact order they will get to the pile of boxes, but the mother always gets there before the daughter, and the father is always last.

Each of the four like and dislike different fruits. Fruits they like are marked below with a tick and fruits they dislike are marked with a cross:

				
Father	✗	✗	✓	✗
Mother	✓	✗	✓	✓
Daughter	✓	✓	✓	✗
Son	✓	✓	✗	✓

Question

Drag the fruits into the boxes so that everyone is guaranteed to get a fruit they like.



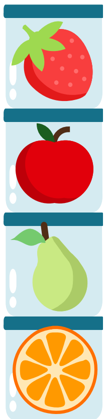
Continued on next page

Fruit Stack - continued

EXPLANATION

Answer

There is only one correct solution:



Explanation

We first look at what the father wants. He only likes oranges and will be the last one to reach the boxes. Therefore, we need to put oranges into the box at the bottom so he is guaranteed to get a fruit he likes.

Because we know that the mother will be taking her box before the daughter gets up, the mother is either the first or the second one to take a box.

For the same reason, the daughter is the second or third one to take a box. The son can be first, second, or third.

To summarize, the following three arrival orders are possible:

1st	Mother	Mother	Son
2nd	Son	Daughter	Mother
3rd	Daughter	Son	Daughter
4th	Father	Father	Father

We see that the second one to get up can be either the son, daughter, or mother. This means that the fruit in the second box from top must me something that they all like.

Continued on next page

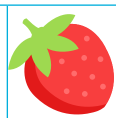
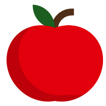
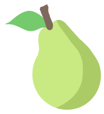
Fruit Stack - continued

Looking at the table of options, the only option that all three enjoy is the apple. (Second row in the table below.)

So we are left with two choices for the topmost box - pear and strawberry. This box can be taken by either the mother or the son. The mother does not like pear. Therefore we have to put strawberry into the first box, which the son also likes. (First row in the table below.)

We can now put pear into the third box, which both the son and daughter like. (Third row in the table below.)

In summary, we have the following options for the order of the family members arriving, which gives us the order of fruits shown below.

1st	Mother or Son	
2nd	Daughter or son or mother	
3rd	Daughter or son	
4th	Father	

BACKGROUND INFORMATION

One of the first things computer scientists learn is the importance of having everything correctly sequenced and the need to understand the background information of the problem. Without knowing exactly who will eat first, we need to organise the data to make the problem solvable. The actual order used in this task is stack order, in particular “Last in, First out” or LIFO. The pile of boxes in the fridge is what computer scientists would call a stack: a structure where only the item on top of the stack can be accessed. Only after removing the top item does another one becomes available. Stacks are used very frequently in programming.

The task asks to find a way of sorting of the fruits which will work under multiple possible conditions. But there are some constraints, and not all possible orders of family members can occur. Solving such constraint problems can be very difficult. Often the best idea to do so is writing and using a computer program to solve the problem.

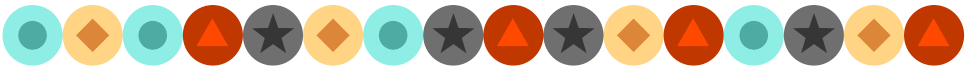
Logic is important in computer science and computer programming, which is why problems that help students understand logic lay a good foundation for when they start creating computer programs. Creating tables to display all possibilities (as shown in the explanation) is a good way to sort and sequence the given data. The use of Boolean logic may also be useful by using AND, OR and NOT to determine which data is useful in any given sequence. Students will start to understand conditionals in programming too, such as ELSEIF, by solving computational problems like the one shown in this question.

Once students have a good understanding of logic and how problems can be dealt with by following and sequencing commands logically, they will be better placed to write their own computer programs to solve problems with many variables. They will then be able to write programs to help deal with stacks.

Longest Sequence

Beaver Ali is making jewellery using beads of four unique colours. Ali makes a chain by randomly selecting and lining up 16 beads in a row. However, Ali prefers long unbroken sequences of identical beads, so they allow themself to change the colour of at most 3 beads in the chain.

Ali pulls out the following 16 beads and lines them up randomly as shown:



Question

What is the length of the longest possible unbroken sequence of identical beads that Ali can make for this chain?

4 | 5 | 6 | 7

EXPLANATION

Answer

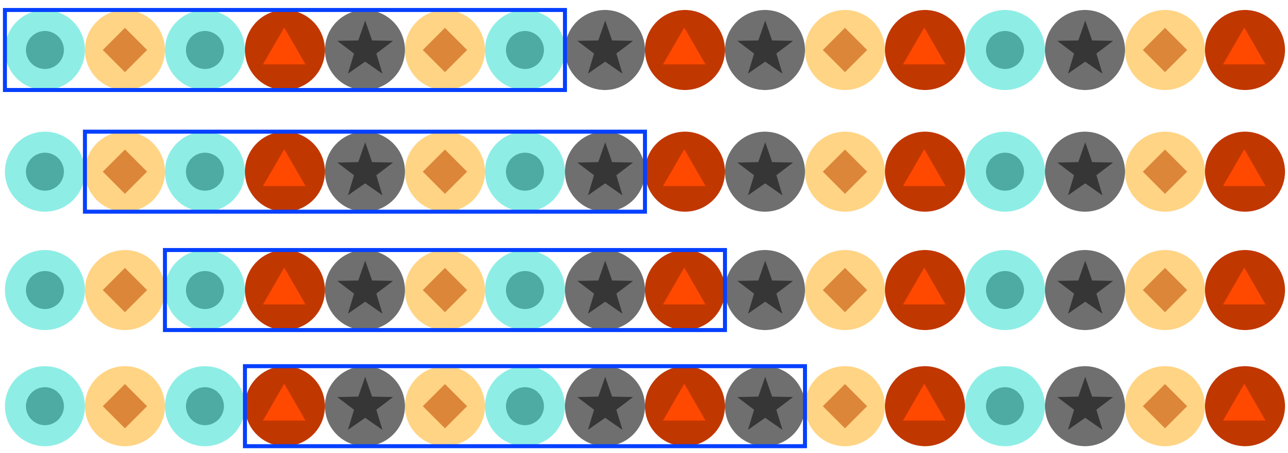
The correct answer is 6. To show this, we need to prove two things:

1. That an unbroken chain of length 6 is possible, and
2. that an unbroken chain of length greater than 6 is not possible.

The first part is easy to prove. Here is how an unbroken chain of 6 stars can be made:

To prove that an unbroken chain of length greater than 6 is not possible, consider any chain of length 7. Since we are only allowed to change three shapes, any chain of length 7 in the original sequence must already have four identical shapes in it.

There are ten chains of length 7 in the original sequence, a few of which are shown below. In no chain can four identical shapes be found.



Since it is not possible to have an unbroken chain of length 7, it is certainly not possible to have an unbroken chain of length greater than 7.

Thus, we have shown that the length of the longest unbroken chain of identical shapes possible is 6.

Continued on next page

Longest Sequence – continued

BACKGROUND INFORMATION

This task is related to finding a longest substring that matches some given criteria. There are many instances in informatics where finding a longest substring is useful, in particular, finding a longest common substring given two strings.

Finding the longest common substring can help detect plagiarism, and help compress data by data deduplication (removing redundant copies of data). Some techniques that can be used to find longest sequences include the two pointer method, and the sliding window.

This question comes from
Taiwan

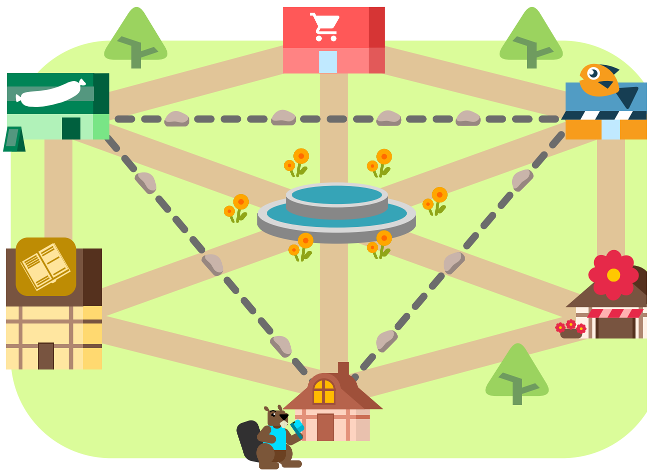
Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Medium

Grocery Shopping

Below is the map of the village where the Beaver family lives. To go from one building to another, Beavers walk along either a muddy path 2021-TW-05_mud or a rocky path 2021-TW-05_rock.

Walking between two buildings takes Little Beaver 5 minutes by muddy path 2021-TW-05_mud and 8 minutes by rocky path 2021-TW-05_rock. For example, it takes Little Beaver 5 minutes to go from the plaza to either the flower shop or to the bookstore, and 8 minutes to go home from either the seafood shop or the meat shop.

Starting from the Beaver's home at the bottom of the map and moving clockwise, the shops in the village are the bookshop, the meat shop, the supermarket, the fish shop, and the flowershop.



Shopping List

Fish / Seafood Shop

Chicken / Meat Shop

Eggs / Supermarket

Flowers / Flowershop

Mother Beaver asks Little Beaver to help with grocery shopping. The shopping list is shown above. Little Beaver has to start the trip from home, finish all the shopping, and come back home. Also, in order to keep the fish fresh, Little Beaver has to visit the seafood shop right before going home.

Question

What is the minimum walking time Little Beaver needs?

EXPLANATION

Answer

The correct answer is 33.

Continued on next page

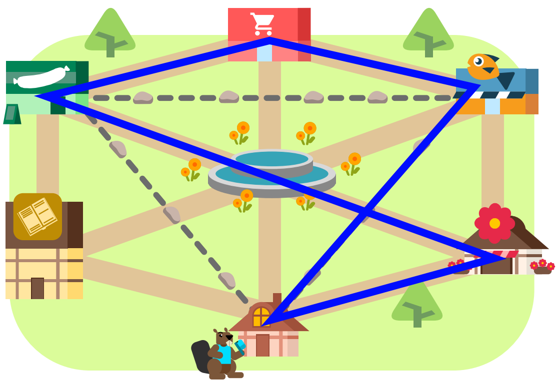
Grocery Shopping – cont'd

Explanation

The original task can be broken down into two ordered sub-tasks: (1) finish shopping at three shops, (2) buy fish and go home. The least walking time needed for each sub-task is shown below:

(1) shopping at 3 shops		(2) buy fish and go home		Total time (mins)
Route	Minimum time (mins)	Route	Minimum time (mins)	
home-meat-flower-market	28	market-seafood-home	13	41
home-meat-market-flower	23	flower-seafood-home	13	36
home-flower-market-meat	20	meat-seafood-home	16	36
home-flower-meat-market	20	market-seafood-home	13	33
home-market-meat-flower	25	flower-seafood-home	13	38
home-market-flower-meat	30	meat-seafood-home	16	46

By breaking down the task, every possible route to complete each sub-task can be explored.
By calculating the minimum time needed to complete each sub-task, it is possible to then get the minimum time needed to finish the whole task. Based on the table above, the minimum walking time Little Beaver needs to finish shopping and also buy fish last before going home is 33 minutes. The route Little Beaver should take is shown below.



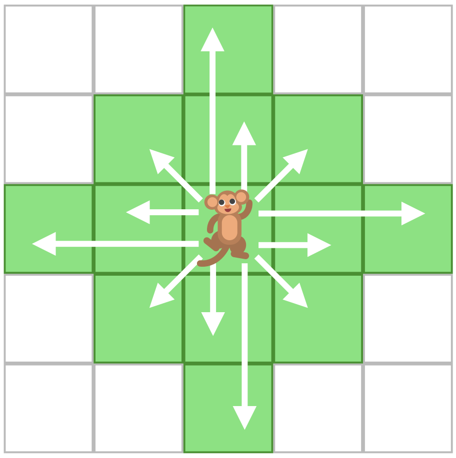
BACKGROUND INFORMATION

In this task, we can find the shortest route by drawing it out. However, in real life, when there are thousands of paths to choose from instead of 15, we will need help from technology, such as GPS and routing software, to find the shortest path.

In computer science, graphs are a common way to show relationships between data. Graphs can be used to represent links between objects. It is a method of representing the connections between things with vertices and edges. Graphs also make it easier to describe the relationships (often represented by edges) amongst key points (often represented by vertices) of complex concepts.

In *graph theory*, the *shortest path problem* aims to find a path (edge) with the shortest distance between two things (vertices) in a graph. Sometimes, the paths have different weights, so the distance between two things has to be multiplied by the weights. In this case, the shortest path problem is solved by finding the minimized weighted distance.

Jumping Jack



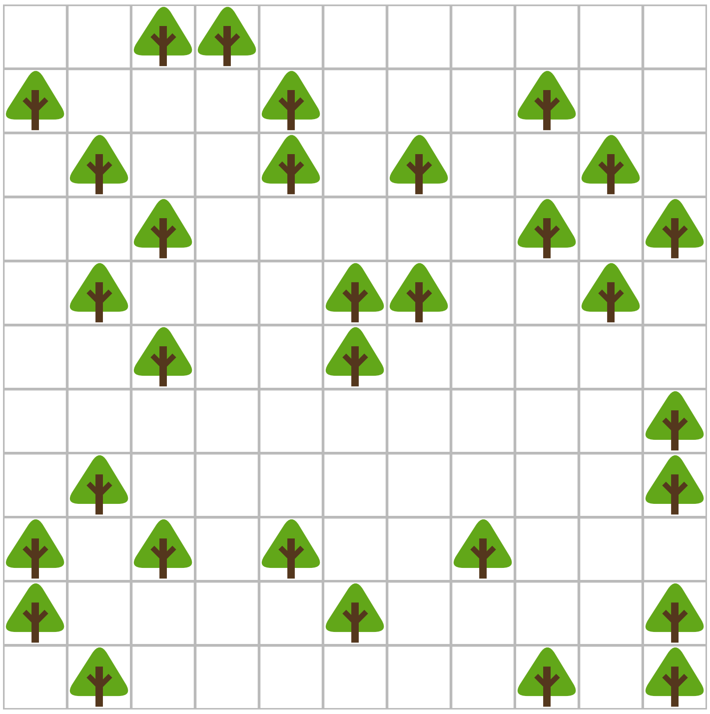
Jack the monkey lives in a park. He can jump from one tree to another if it is either up to two cells away horizontally or vertically, or one cell away diagonally, as shown in the diagram on the right.

Jack plays a game in which he jumps to as many different trees as possible without touching the ground. He can start from any tree in the park.

In the map below you can click on a tree to change it from one type to another.

Question

Find the biggest number of trees Jack can visit in one go without touching the ground and change them to orange square trees.

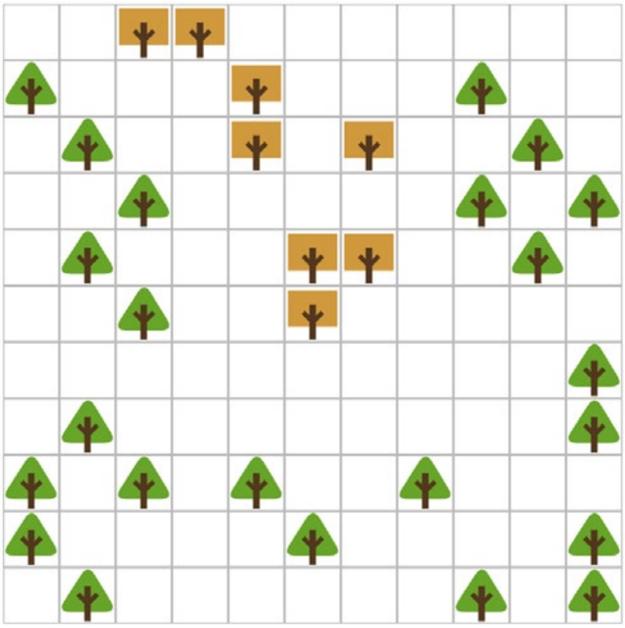


Continued on next page

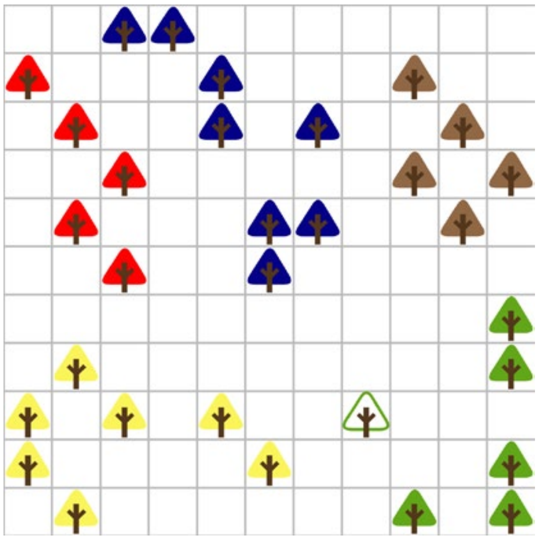
Jumping Jack – continued

EXPLANATION

Answer



In the diagram below, the groups of trees that Jack can visit without touching the ground have been coloured in different colours.

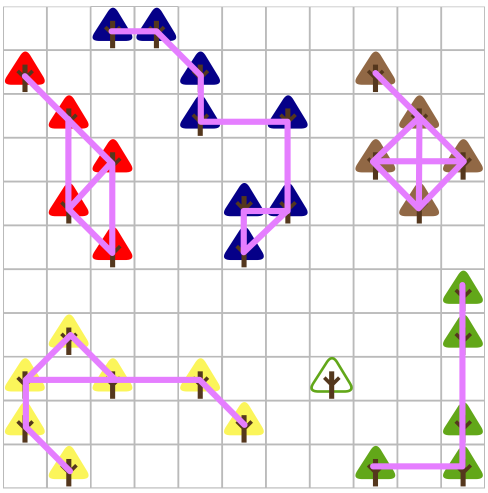


There are six groups of trees in the park. If Jack starts on a tree coloured in yellow, he can reach all the yellow trees, and no trees of other colours. How do we find such groups? Pick a random tree and colour it in a certain colour. Then use the same colour for all trees that are reachable from it. And all trees that are reachable from those trees, too. And so on, until you cannot reach any other trees. If there are any trees that haven't been coloured yet, take another colour and start again from a random uncoloured tree. This colouring simulates Jack exploring.

The largest group of trees is the dark blue one, which contains 8 trees. The dark blue cluster is the correct answer.

BACKGROUND INFORMATION

From the point of view of computer science, this question involves manipulating a *graph*: the trees are called *vertices*, and two trees are connected with an edge when Jack can jump between them. In the diagram below, the edges are as purple lines between trees.



If there is a path using these edges that allows Jack to go from one tree to another, then these two trees belong to the same group. These groups are called the *connected components* of the graph. Here a different colour is used to represent each connected component.

The procedure for colouring is similar to a number of different graph algorithms that deal with searching: *breadth-first search* and *depth-first search*.

Counting by Nodding

A ticket vending machine uses computer vision (CV) for communication. To purchase *n* tickets, the customer standing in front of the vending machine must nod *n* times and then raise their head once. The CV system constantly detects the vertical length of the bridge of the nose in the live camera image and assigns it to the variable *nose*.

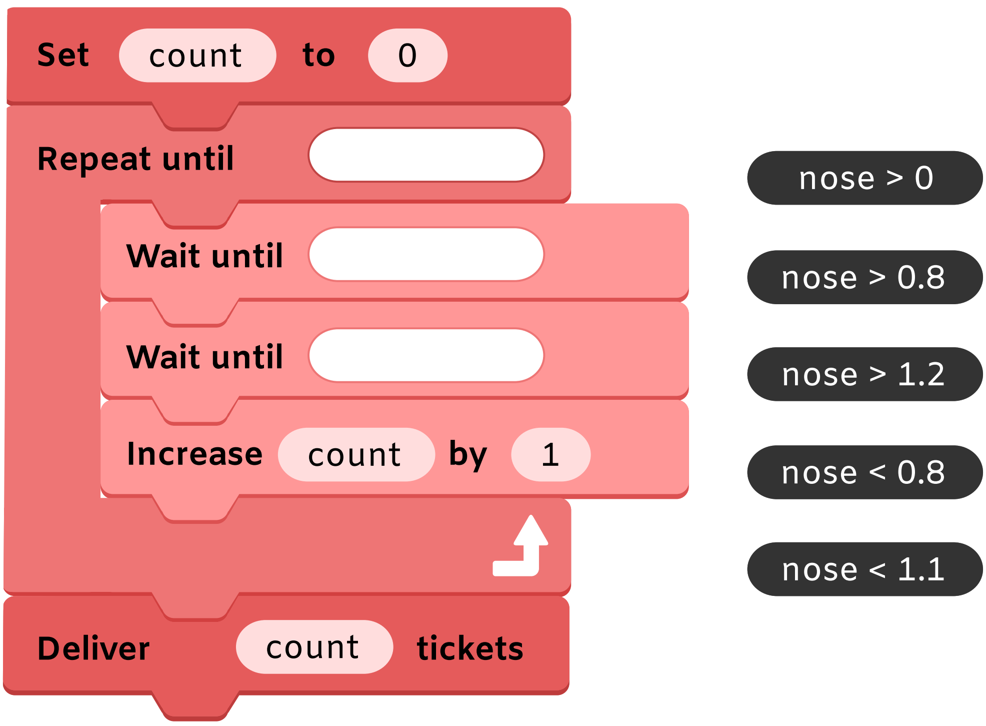
If the value of *nose* is 1, the head is in its normal position.	
When the customer nods and the head goes down, the value of *nose* gets greater than 1, because the nose appears to be longer.	
When the head is raised, the value of *nose* gets lower than 1.	

The control program is started when a customer stands in front of the vending machine and the head is in its normal position.

Question

A skeleton of the control program is shown below.

Complete the program by dragging the appropriate condition blocks from the right to the gaps in the program.



Continued on next page

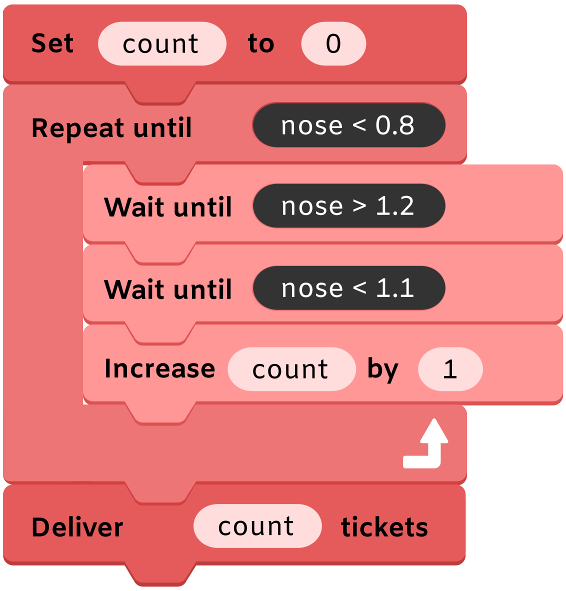
This question comes from Germany

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Medium

Counting by Nodding – cont’d

EXPLANATION

Answer



Explanation

The program uses two variables named *count* and *nose*. The variable *count* contains the number of nods, and *nose* represents the visible nose bridge (see task) and is automatically updated by the CV system.

A sequence of three commands is repeated in a loop until the head is raised and therefore nose gets a value smaller than 0.8. These repeated commands manage the counting: First the system waits until the head goes down (*nose* > 1.2) and then waits until it goes up again (*nose* < 1.1). This is one complete nod. The value of the variable count is increased by 1.

When the loop is finished (because the person has raised their head such that *nose* < 0.8), the variable count contains the number of nods and count tickets are delivered.

The program has to use inequalities, as in real life applications, it will be difficult for the user to get the exact value of nose = 1 to signify the end of each nod. Similarly, the values used to trigger an action should be significant enough that it can be considered a deliberate action of the user. For example, if nose > 1 is used to measure a nod, minuscule head movements can be counted as one nod even though it is not the intention of the user.

BACKGROUND INFORMATION

Computer vision (CV) makes it possible to communicate with a machine by gestures. An e-book reader with a clever CV control system enables a person who cannot use thier hands to turn pages by head movements. For programming languages there exist special program libraries like OpenCV supporting CV. These libraries contain special commands that make it possible to detect parts of a face like the eyes or the bridge of the nose in a camera image.

In the task, the program shown is described as a “skeleton of the control program” because it is not a finished program. It will need further real life testing to check that the variables chosen work in a good variety of situations, that it produces reliable data in a high percentage of occassions, and that there are no bugs that lead to the vending machine behaving in unexpected ways.

This question comes from Switzerland

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Medium

Playing with Hats

A beaver likes to play a game by placing circular pebbles on square paving stones.

The beaver moves from left to right, one square at a time.

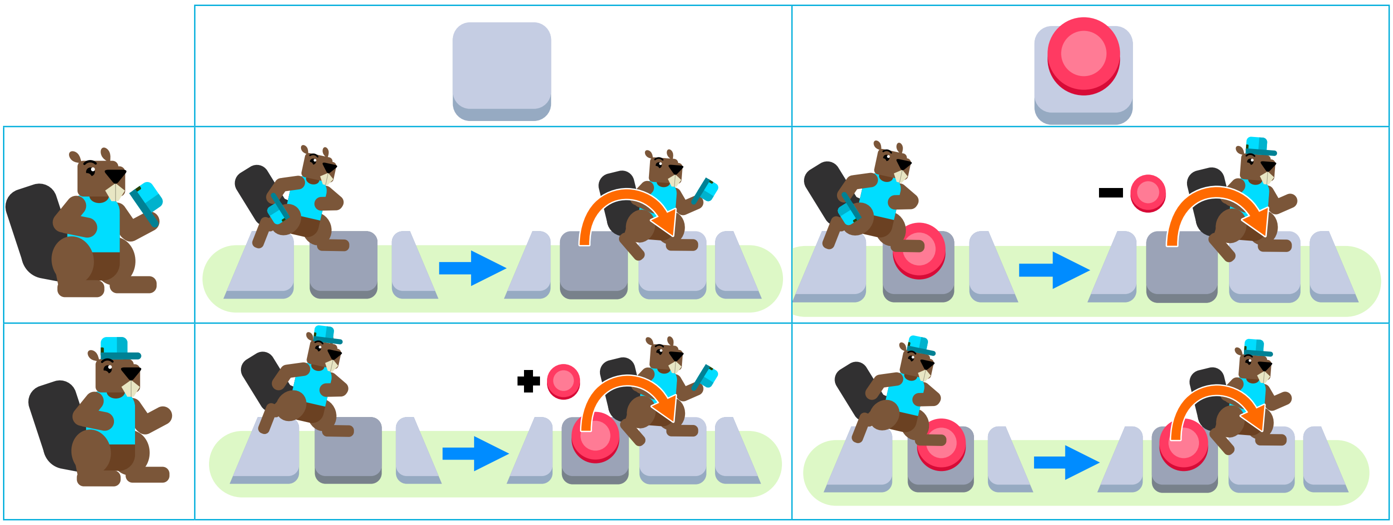
The beaver has a hat and behaves differently depending on whether they have the hat in their hand or on their head.

The rules that the beaver follows are listed below: :

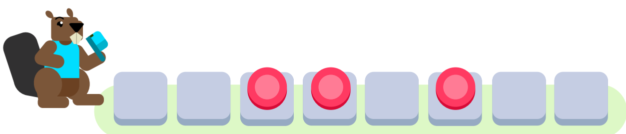
- If the beaver has the hat in their hand and steps on a square with no pebble, they continue on with no change.
- If the beaver has the hat in their hand and steps on a square with a pebble, they take the pebble and put the hat on their head before moving to the next square.
- If the beaver has the hat on their head and steps on a square with no pebble, they place a pebble on the square and take the hat off before moving to the next square.
- If the beaver has the hat on their head and steps on a square with a pebble, they continue to the next square with no change.

The pictures in the table below show the rules of the game. The changes for each situation are shown "before → after".

Rules:



At the beginning, the beaver has the hat in their hand and three pebbles are on the squares in the positions shown below.



Question

Show which squares have pebbles on them after the beaver has moved over all of them and left the last square.

Continued on next page

This question comes from Switzerland

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Medium

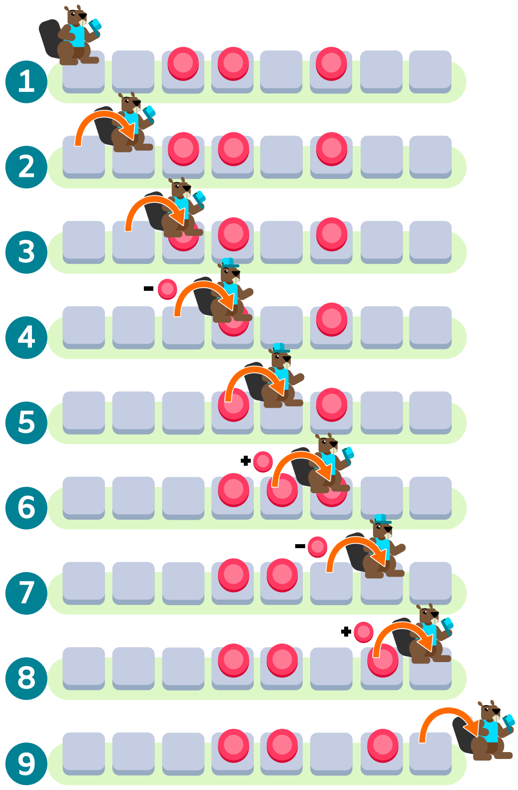
Playing with Hats - continued

EXPLANATION

Answer

Explanation

The solution can be found by step-by-step analysis. We show this in this picture, using the rules below:



BACKGROUND INFORMATION

The beaver has two states:
- hat in hand
- hat on head

Depending on its state, the beaver behaves differently. The beaver with its rules behaves like a Turing machine. A Turing machine is a useful model for computation in computer science. Although it is very simple, it is as powerful and as efficient as any programming language. This means any software program can be converted into a Turing machine and, conversely, any Turing machine into a program. It was first described in 1936 by the English mathematician and computer scientist Alan Turing. Turing machines are one of the most important formal models in computer science.

- A Turing machine has various necessary components:
- A long tape divided into squares. Normally it is said to be infinite.
- A finite alphabet of symbols, e.g., 0, 1. In our example we used a pebble and no pebble.
- A read/write head: this would be able to look at a square and read its symbol. After reading and proceeding according to the rules the head would then move left or right one square at a time. In our case the beaver represents the read/write head.
- A finite set of states: we used two states: hat in hand and hat on head.
- A set of rules (transition rules): to specify how the machine operates (see task description).

This question comes from
Switzerland

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Medium

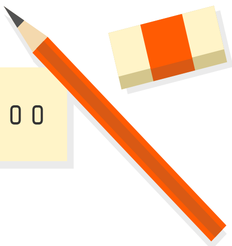
Compact Representation

Beaver Xavier wants to be able to send secret messages to his friends, so he creates a code to represent some letters with binary digits 1 and 0. He notices that letters T and E are more frequent, thus he decides to give them a shorter representation and thus code the letters T, E, A, K, C, and R as follows:

Letter	T	E	A	K	C	R
Code	1	00	0010	0110	1010	1110

Xavier sent this coded message to Yvonne:

1 0 0 1 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 1 1 1 0 0 0



Question

In letters, what is the complete message written by Xavier?

EXPLANATION

Answer

The correct solution is: TAKECARE.

Explanation

Here is the correspondence between characters and their binary representation in Xavier's message:

Letter	T	A	K	E	C	A	R	E
Code	1	0010	0110	00	1010	0010	1110	00

To reconstruct the message, Yvonne must find a way to segment the whole message into a sequence of letter. This is not so easy to do starting from the left as multiple options for decoding quickly appear.

For example, starting from the leftmost digit – Yvonne can quickly identify that the first letter is T and corresponds to a "1". This is because there are no letters that correspond to the sequences "10", "100", or "1001".

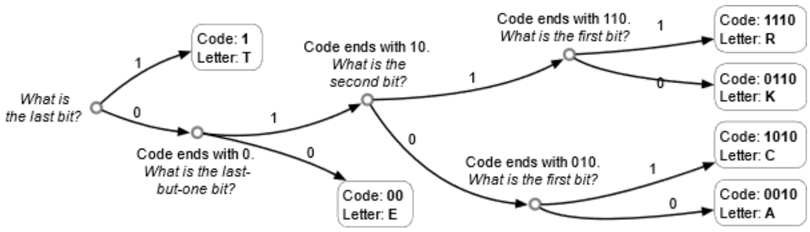
The second letter is a problem: it could either be E, represented by "00", or A, represented by "0010". At this stage, Yvonne can't know for sure. The message Xavier sent is unambiguous, though: when decoding, if Yvonne made the wrong choice at the second position and choose E, she will be stuck later on and realise that guessing E was incorrect – therefore the only possibility was A.

Continued on next page

Compact Representation – cont'd

However, Yvonne never has to "make a guess" when decoding a letter if she starts from the last digit in the sequence. This is because the code is suffix-free: there is no code word which ends in a sequence of 1s and 0s that would itself be another code word. Thus, one can easily reconstruct the text unambiguously by reading the binary code from right to left. When a code of a letter has been found, one can exchange the code for the letter.

The diagram below illustrates how the binary message can be read from right to left, yielding letters unambiguously:



Had Xavier wanted it to be possible for someone to decode this message unambiguously at all stages from left to right, this would have required a *prefix-free code*, i.e., a code where no code word begins with a sequence of 1s and 0s that is itself another code word. Xavier's code is not prefix-free, as the code word "0010" for A begins with "00", which is itself the code for E.

BACKGROUND INFORMATION

All objects a computer is working with must be described as sequences of bits. This is also true for texts. One always expects that the original object can be reconstructed from its binary representation, but this is only possible if it never happens that two or more different objects have the same binary representation. Computer scientists are asked to develop such systems of codes that one can efficiently reconstruct the original object (for instance, a text) from its binary representation.

If one wants to compress a text (to get a binary representation of the text that is as short as possible), then a good strategy is to take shorter binary codes for the more frequent letters and use longer codes for the letters that are rare. One has to take care in this case to choose codes that guarantee an unambiguous decoding (reconstruction of the original text) efficiently. Very good choices for this purpose are *prefix-free codes* and *suffix-free codes*, whose principles are described in the answer explanation.

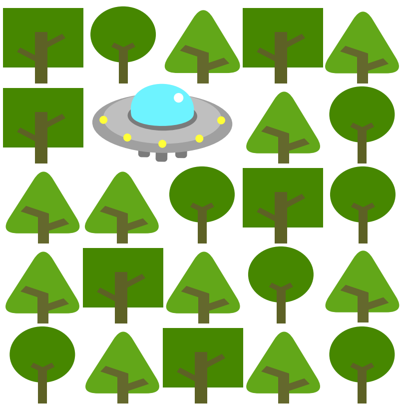
This question comes from Russia

Save the Trees

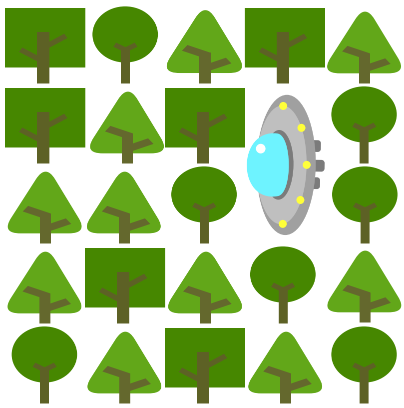
On Planet Bebras there are many beautiful trees but no places for spaceships to land. A developer wants to cut down trees in order to build spaceship landing zones. A ranger wants to save the trees. They come to the following agreement:

- The ranger is allowed to mark 3 trees, which cannot be cut down, and
- A tree can only be cut down if needed to build a landing zone. That is, a tree cannot be cut down for no reason.

The trees are arranged in a 5-by-5 grid. Spaceships are rather large and they require the space occupied by 2 adjacent trees in order to land. The trees can be horizontally or vertically adjacent, but not diagonally.



and



illustrate the two valid ways a spaceship could land.

Question

Assuming the ranger marks three trees wisely in order to save the most trees, and assuming the developer cuts down trees wisely in order to build the most landing zones, how many landing zones will be built?

11 | 10 | 9 | 8

EXPLANATION

Answer

The correct answer is 9.

Explanation

Because a spaceship always requires two adjacent squares to land, the grid of trees can be thought of as a chess board:

1	2	3	4	5
10	9	8	7	6
11	12	13	14	15
20	19	18	17	16
21	22	23	24	25

Continued on next page

Save the Trees – cont'd

Thinking of the grid this way, a spaceship always takes up one black square and one white square. In the grid there are exactly 13 white squares and 12 black squares, allowing a maximum of 12 spaceships to land. If the rangers marked one tree on a white square, that would bring the total of white and black squares to 12 and 12, still allowing a maximum of 12 spaceships to land. From this it follows that the ranger should only mark trees on black squares. If the ranger marks three black squares, they can block three potential landing zones, bringing the number of landing zones to at most 9 (as there are now only 9 black squares despite there being 13 white squares).

It can also be proven that marking 3 black squares cannot block more than 3 potential landing zones. This shows that the answer is exactly 9. To do this, number the squares on the chess board as shown in the diagram. A spaceship can always land with one side on an even-numbered square with number n, and the other side on square n+1.

If the squares are 'unfolded' from the board, in the order that they were numbered, the following chain is made:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Marking a tree on a black square basically means ripping the chain into two parts at the place of the black square that you marked. If a chain is ripped with n black squares into two parts, there is a left side and a right side. Each side starts and ends with a white square and has alternating white and black squares. The total number of black squares is now n-1.

For instance, if tree 4 is marked first, this creates two chains of 1-3 and 5-25. If tree 8 is then marked, the second chain is split up into 5-7 and 9-25. If tree 10 is then marked, the rightmost chain is split into 9-9 and 11-25.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

For each of the chains the developer can still land exactly as many spaceships as there are black squares.

So this shows that if the ranger marks 3 trees in black squares, they will block exactly 3 landing zones.

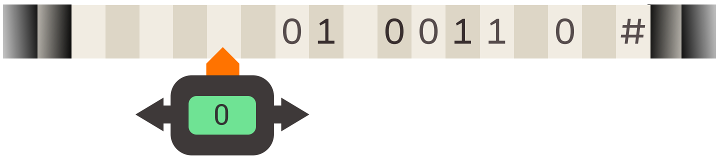
BACKGROUND INFORMATION

Solving this task by trying out every combination of marking 3 trees and then figuring out how many spaceships could still land is called a *brute force* strategy and would take a very long time. This task highlights the power of some logical thinking tricks that involve abstraction. In order to solve this task, the spaceships need to be abstracted as 2x1 rectangles that are placed on a board filled with squares. Often times these 2x1 rectangles are called 'dominoes', just like in the game.

In addition to what is asked in this question, there are some clever algorithms that exist that can show how many different ways the spaceships could land.

Turing Machines

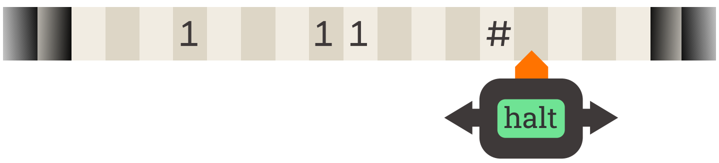
A Turing machine is a computational model of a computer. It consists of a state (shown on the green screen) and a read/write head (orange arrow) that operates on a tape containing symbols. The head can move left (L) or right (R) one symbol at a time. The machine always starts in state “0” (initial state), as shown below.



Our version of the Turing machine can run programs and each line of such program consists of 5 elements as shown in the following short program:

Current State	Current Symbol	New Symbol	Direction	New State
0	-	-	R	0
0	1	1	R	1
0	0	-	R	0
0	1	1	R	halt
1	0	0	R	1
1	1	1	R	1
1	-	-	R	1

When the program is executed, the first line (from the top) with matching current state and symbol at the current head position is used to determine a new symbol to overwrite the tape at the current position, a direction for moving the head and a new state for the machine. “_” denotes the space character. The “halt” state stops the program. Empty lines are only used for formatting. The program above removes leading zeroes from binary numbers, arriving to this final state:



From the initial state, the head moves right until it hits the first 0, remaining in state 0. Here, given the 3rd rule, the machine replaces this first 0 with a space, moving right and remaining in state 0. Now the head reads “1”, so the machine leaves this alone on the tape, moving right and into state 1. The next symbol is a space, so according to the program, the machine leaves the space on the tape and moves back to state 0. The process continues this way until the machine reaches the final state shown above.

Continued on next page

This question comes from
Austria

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Hard

Turing Machines – continued

Question

What does the following program do?

Current State	Current Symbol	New Symbol	Direction	New State
0	1	-	R	1
0	*	*	R	0
1	1	-	R	2
1	*	f	R	halt
2	1	-	R	1
2	*	t	R	halt

"*" in column 2 means "any character".

"*" in column 3 means "same character that was read"

A) The program replaces "1"s with "2"s.

B) The program replaces "2"s with "1"s.

C) When finding a group of "1"-symbols the program prints "t" (true) if the number of "1"s is even. Otherwise it prints "f" (false).

D) When finding a group of "1"-symbols the program prints "t" (true) if the number of 1s is odd. Otherwise it prints "f" (false).

EXPLANATION

Answer

C: When finding a group of "1"-symbols the program prints "t" (true) if the number of "1"s is even. Otherwise it prints "f" (false).

Explanation

The correct answer is option C: When finding a group of "1"-symbols the program prints "t" (true) if the number of "1"s is even. Otherwise it prints "f" (false).

In the starting state, the line [0 * * r 0] means that all symbols except the "1"s are skipped. The first line changes to state "1" if a "1"-symbol is detected.

Being in state "1" means, that an odd number of "1"s was detected so far. If another "1" is detected, the machine changes to state "2". If anything other than "1" is detected, the machine prints out "t"

Continued on next page

Turing Machines – continued

and halts. Being in state “2” means that an even number of “1”s was detected so far. If another “1” is detected, the machine changes to “state 1”. If anything other than “1” is detected, the machine prints out “f” and halts.

This is exactly the result that was suggested by option C).

Option A: The program replaces “1”s with “2”s.

If the program replaced all “1”s with “2”s, we would expect a line in the program with “1” in the second column and “2” in the third column. This is not the case. Option A) can be ruled out.

Option B: The program replaces “2”s with “1”s.

Similarly, as above, to replace “2”s with “1”s, we should have “2” or ‘*’ in the second column (current symbol) and a “1” in the third column (new symbol). But we do not have such lines. So B) is incorrect.

Option D: When finding a group of “1”-symbols the program prints “t” (true) if the number of 1s is odd. Otherwise it prints “f” (false).

The correct answer (C) implies this option is incorrect. In detail: an even number of “1”s leads us to state “1” (either from state “0” by reading a “1” or from state “2” by reading another “1”).

In either case, from state “1”, with an even number of “1”, we can either:

a) read another “1” leading us to state “2” but having now an even number of “1”s; or b) read a different symbol, terminating the program with output “f” (false). Hence, option (D) is incorrect.

BACKGROUND INFORMATION

A Turing machine is a computational model for a computer, developed by the British mathematician Alan Turing in 1936. Although Turing machines are a simple concept, researchers agree that any algorithm that runs on a classic computer can also run on a Turing machine, although not very efficiently. Conversely, if one can show that an algorithm can not be run on a Turing machine, that algorithm can not be run on a classical computer either. So we can think of a Turing machine as a computer that is reduced as much as possible to draw conclusions and find general properties of classical computers. Quantum computers have to be seen as an exception, they do not fall in the same category as classical computers.

Mastermind

Isla is playing Mastermind on her computer: The computer makes up a password from four distinct digits. The player can submit several guesses of this password. Each time, the computer responds with the number of *correct* digits, which appear both in the guess and in the password. Also, it tells the player whether they placed these digits in the right positions.

Isla made some guesses (?). From the responses given, she was able to discover the password.

?	?	?	?	
5	7	2	0	One of the digits is correct and in the proper position.
6	0	3	1	One of the digits is correct, but not in the proper position.
1	4	8	5	Two digits are correct, but they are not in the proper positions.
1	5	9	6	None of the digits are correct.
8	1	2	5	One of the digits is correct, but not in the proper position.

Question

What is the password?

EXPLANATION

Answer

The password is 3748.

Explanation

To arrive at this number we must systematically follow a procedure according to the information available. This process must be iterative. One of the ways to solve it is to subdivide the problem and first try to discover the digits and then the correct order.

From the **first row** (guess plus response) we know that only one of the given digits is part of the password. The second and third rows provide similar information.

The **fourth row** then helps to extract more information from **rows one to three**: We can discard 5 from the first guess, 6 from the second guess, and 1 and 5 from the third guess. From this last step we know that 4 and 8 are parts of the password. Then the correct digit in the fifth guess is 8; from that we can conclude from **row five** that 1, 2 and 5 are not parts of the password.

If we go back to rows **one and two**, the possibly correct numbers are 7 or 0 in **row one**, and 3 or 0 in **row two**. Thus, we have 5 possible numbers 0,3,4,7 and 8. If 0 was in the password, then both 7 and 3 would need to be discarded given that only one number is correct for **rows one and two**. This would leave only 0,4,8 as possible numbers in the password. Given that the password must contain four unique numbers, 0 cannot be correct. Therefore, the other digits that complete the code must be 3 and 7.

Continued on next page

Mastermind – continued

Now we consider the information of the position or order. From **rows three and five**, we know that 8 can be in positions 2 or 4:

?-8-?-8

and that 4 can be in all positions but 2:

4-?-4-4

From **row one** we know that 7 must be in position 2; then, 8 must be in position 4. From **row two** we know that 3 cannot be in position 3, so that it must be in position 1. Hence, 4 is in position 3. This gives the only possible option for the password which is:

3-7-4-8

BACKGROUND INFORMATION

Logic plays a key role in computer science (databases, computational complexity, programming languages, artificial intelligence, hardware and software design and verification, etc.), and is undoubtedly one of the foundations that provide the maturity and agility to assimilate the future computer science concepts, languages, techniques, etc.

There are some complex algorithms that use the same logic flow to detect intruders in computer networks, or errors in logic circuits. As it can be hard to examine each part individually, a batch of test questions is sent and based on the answers the presence of an intruder/error can be determined.

This question comes from the United States

Truth Table

A truth table describes the output (○ or ●) for one or more input variables. The diagram below describes what the output will be for all different combinations of x, y and z. This truth table can also be described as a formula by just listing exactly for which input values the output would be ●:

x	y	z	output
○	○	○	○
○	○	●	●
○	●	○	○
○	●	●	○
●	○	○	●
●	○	●	●
●	●	○	●
●	●	●	●

(x=○ and y=○ and z= ●) or

(x= ● and y=○ and z=○) or

(x= ● and y=○ and z= ●) or

(x= ● and y= ● and z= ○) or

(x= ● and y= ● and z= ●)

In this case there are exactly 15 input symbols.

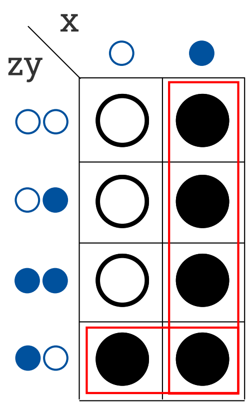
Analysing this table shows that the output is always ● when x= ●, so this formula can be reduced to four symbols:

(x=○ and y=○ and z= ●) or (x= ●)

This can be shortened further - this table can be represented with just three symbols:

(y=○ and z= ●) or (x= ●)

This shortest representation of three symbols can be drawn in the following diagram:



Here, (y=○ and z=●) is represented by the 1x2 red rectangle and (x=●) is represented by the 4x1 red rectangle.

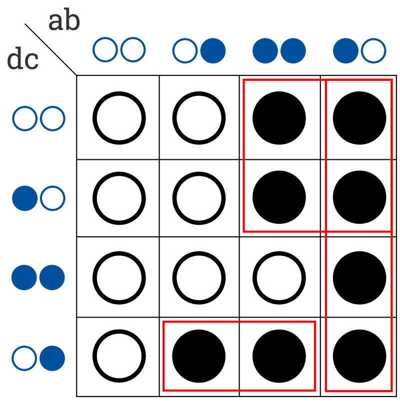
Continued on next page

Truth Table – continued

Question

Here is a much bigger table, again paired with a matching diagram with the same information (note the order of the ‘c’ and ‘d’):

a	b	c	c	output
○	○	○	○	○
○	○	○	●	○
○	○	●	○	○
○	○	●	●	○
○	●	○	○	○
○	●	○	●	○
○	●	●	○	●
○	●	●	●	○
●	○	○	○	●
●	○	○	●	●
●	○	●	○	●
●	○	●	●	●
●	●	○	○	●
●	●	○	●	●
●	●	●	○	●
●	●	●	●	○



What is the lowest number of symbols that needs to be used to describe this truth table?

Truth Table - continued

EXPLANATION

Answer

The correct answer is 7.

Explanation

You can find this by looking in the diagram with the circled groups. These diagrams are called Karnaugh maps.

The size of the circled output symbols basically indicates how many symbols you need to describe each group.

For this table that has inputs for four symbols:

To describe a single output symbol, you will need four input symbols. To describe two adjacent symbols, you will need three input symbols. To describe four adjacent output symbols (either in 4x1, 1x4, or 2x2) you will need two input symbols, and to describe eight adjacent symbols (2x4 or 4x2) you need only a single input symbol.

This diagram shows three groups, two groups of 4 output symbols (2x2 and 4x1), and 1 group of 2 output symbols (1x2). So the answer is 2 + 2 + 3 = 7.

There should be a rule for each of the three groups in the Karnaugh map - the 2x2 group, the 4x1 group and the 1x2 group. The final notation of input values would be the following three rules with a total of 7 input symbols:

(a= ● and c= ○) or

(a= ● and b= ○) or

(b= ● and c= ● and d= ○)

BACKGROUND INFORMATION

Karnaugh maps are used to simplify logic tables, so they can be expressed using a minimum number of logical gates. When building circuits, the cost of the circuit depends on how many elements are needed in the circuit. Any way by which fewer inputs can be used was seen as positive. Karnaugh maps help find this minimal set of inputs.

Karnaugh maps are useful because they have been designed so that in order to move from one square to an adjacent square, never more than one bit has to be changed. Do note that the squares wrap around! From the top square in a column to the bottom square in the same column is also a single change. Because of this special property, if two adjacent squares both contain an output of ‘1’, the bit that flips to move between them is not needed in the final expression, and both rows can be combined from the truth table.

Karnaugh maps used to be used to also find potential problems in logical circuits called race conditions.

Karnaugh maps work well up to 4 variables. This may not seem like a lot, but many small circuits have 4 or fewer inputs, or can be split into multiple circuits of fewer than 4 variables.

This question comes from Germany

Years 3+4
Years 5+6
Years 7+8
Years 9+10
Years 11+12 Hard

Log Sort

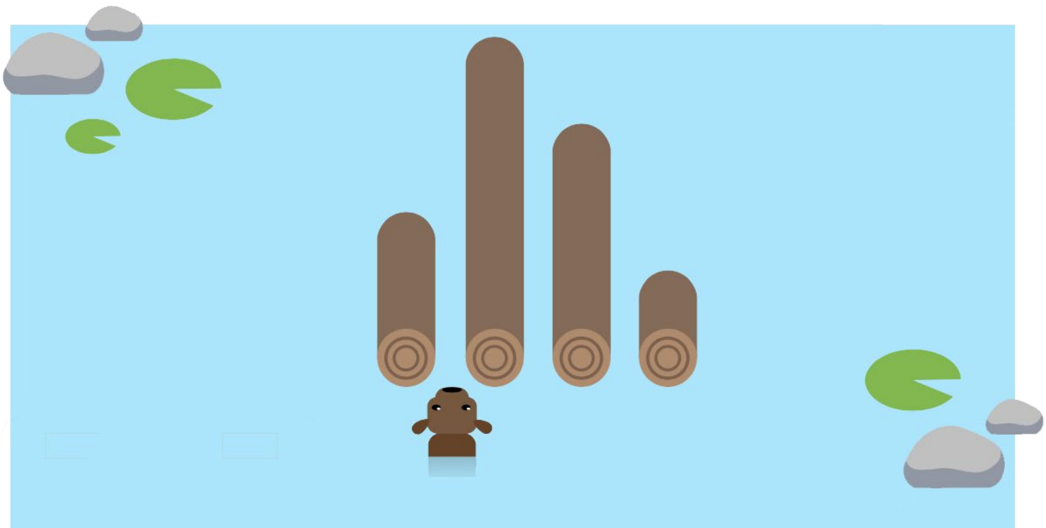
Tree logs of different sizes are in a river. Beaver Hamid's task is to sort the logs by size. Hamid moves along the riverbank, always taking a position between two logs. Hamid compares these two logs by size and swaps them if necessary.

Hamid knows the logs can be sorted in the following way, no matter how the logs come in initially:

Start at the position on the right of the leftmost log.
Repeat the following until you are on the right of the rightmost log:

- If the log on the left is smaller than the right log: move to the right by one log.
- If the log on the right is smaller than the left log:
  - swap these logs;
  - unless you are at the starting position: move to the left by one log.

See how Hamid sorts 4 logs in this way. In this example, Hamid has to move 9 times.



The number of times Hamid has to move to sort a group of logs depends on how the logs come in initially. In the worst case, Hamid would have to move 25 times to sort 6 logs. In the best case (when the 6 logs are already sorted) Hamid would still have to move 5 times.

Question

Which of the ranges below is the *smallest* range that will always include how many times Hamid has to move when sorting all starting arrangements of 60 logs?

0...60 | 10...90 | 59...300 | 59...3,600 | 59...216,000

Continued on next page

This question comes from
Germany

Log Sort – cont’d

EXPLANATION

Answer

59...3,600.

Explanation

[0...60] is wrong. Even in the best case, when the logs are sorted, Hamid has to make 59 moves. More moves are required if the logs are out of order quickly taking the moves required to beyond 60.

[10...90] and [59..300] are also wrong. To prove it, we need to explore the worst case, when logs are sorted in the opposite order. In this case, Hamid reaches a log on the k^{th} position and moves it to the first (leftmost) position, then goes for a log on the $(k+1)^{th}$ position. So for the log on the k^{th} position, we need to move k-2 times right to reach it, and k-2 times left to put it in the beginning. Thus, we obtain the sum 2(1+2+...+58) and we need to add 59 moves from the leftmost to the rightmost position at the end of the algorithm. The sum is exactly 592=3481. This number does not belong to any of these two ranges.

[59...3600] is correct. To see it, we need to prove that the worst case is really the worst one. When Hamid reaches the log in the k^{th} position, all previous logs are already sorted properly, so he needs only to put this new log in the correct position among the previous ones. Then he goes to the log on the $(k+1)^{th}$ position. So, the smaller is the log on k^{th} position, the greater number of move it requires.

[59...216,000] does include how many times Hamid will have to move when sorting the logs no matter what their starting positions but this range is not the smallest range that does this.

BACKGROUND INFORMATION

In Computer Science, sorting algorithms are used to put a sequence of objects in a certain order. The most frequently used orders are numerical order (for numbers) and lexicographical order (for all kinds of data based on an ordered alphabet). Efficient sorting is important for optimising the efficiency of other algorithms, such as search algorithms that require input data to be sorted. Also, sorting can be useful for canonicalising data and for producing human-readable output.

One of the most well-known sorting algorithms is the *gnome* sort. It is conceptually simple - by working with one item at a time, the algorithm gets each item to its proper place by a series of swaps. If the list is initially almost sorted, it works with n swaps for n objects. The *gnome sort* (sometimes dubbed “stupid sort”) was originally proposed by the Iranian computer scientist Hamid Sarbazi-Azad (professor of Computer Science and Engineering at Sharif University of Technology) in the year 2000.

When speaking of an algorithm we always need to consider how ‘fast’ it is, i.e. the number of operations it requires to sort items in the worst or most disordered case, depending on the number of elements needed to be sorted. For this algorithm, if we have n objects, we need approximately n^2 operations. This is called a quadratic relationship. The other answers in this task represent other possible relationships: constant (independent) for [0...60], linear for [10...90], log-linear for [59...300] and really big (actually increasing by the power of 3) for [59...216,000].

This relationship for a particular algorithm is called the *algorithm complexity* and is studied in computational complexity theory.

We would like to thank the International Bebras Committee and community for their ongoing assistance, resources and collaborative efforts. Special thanks to Eljakim Schrijvers, Alieke Stijf and Dave Oostendorp for their support and technical expertise.

If you would like to contribute a question to the International Bebras community, please contact us via the details below.

Contact us

CSIRO Digital Careers
digitalcareers@csiro.au
csiro.au/Digital-Careers

Australia's National Science Agency

digital careers