



Computational Thinking Challenge

Solutions Guide 2019



digital**careers**

Thank You

CSIRO's Education & Outreach Program, Digital Careers, is privileged run to the Bebras Challenge in Australia. We would like to thank the International Bebras Community for their support of the Australian Challenge as well as, of course, developing the challenge questions. This year, Bebras Australia runs with a similar set of questions to the UK and US and we would like to thank Bebras UK and US for their generous support.

We would like to thank the Australian Government for funding CSIRO Digital Careers which manages Bebras Australia, and the Australian Curriculum, Assessment and Reporting Authority (ACARA) for providing advice on Bebras questions. Special thanks go to Eljakim Schrijvers and Daphne Blokhuis for their ongoing support for the Australian Challenge. In addition, we would also like to thank the following individuals for their help and support: Peter Millican, Chris Roffey, Andrea Schrijvers and Sue Sentence.

International Community

Bebras is an international initiative whose goal is to promote computational thinking to students in years 3-12. Questions are developed as an international community which means you can see what country created a question by looking at the flag in the top left corner of each question page.



UK Beaver



South Korea Beaver

What is a Solutions Guide?

Computational thinking skills underpin many careers of the future. Creating opportunities for students to engage in activities that utilise their problem solving skills is the foundation for further learning. The Bebras Challenge is an engaging way for students to learn and practice these skills in the classroom environment.

On the following pages you will find the 41 tasks used in the Australian Bebras 2019 Challenge. Above each question is noted the age groups and level. Level A is easy, level B is medium and level C is hard.

After each question there is an answer, an explanation of how the answer is obtained, plus a section on how the questions are related to computational thinking. We have also mapped the questions to the Australian Digital Technologies Curriculum Concepts.

Computational Thinking Skills	Concepts
Decomposition - DE Break problems into parts Pattern recognition - PR Analyse the data, look for patterns to make sense of the data Abstraction - AB Remove unnecessary details and focus on the important data Modelling and simulation - MS Create models or simulations to represent processes Algorithms - AL Create a series of ordered steps taken to solve a problem Evaluation - EV Determine effectiveness of a solution, generalise and apply to new problems	Abstraction - Ab Underpins all content Data collection - Dc Properties, sources, collection of data Data representation - Dr Symbolism, separation – how digital systems represent data i.e. Binary Data interpretation - Di Patterns and contexts Specification - Sp Descriptions and techniques Algorithms - Al Following and describing Implementation - Imp Translating and programming Digital systems - Ds Hardware, software, networks, Internet Interactions - In People, digital systems, data and processes Impacts - Ims Sustainability and empowerment

Contents

Simon Says	page 7	Flipping Cards	page 52
Shoe Prints	page 9	Connections	page 53
Honey Pot	page 10	Park Walk	page 55
Pizza	page 11	Treasure Maps	page 57
Pile of Clothes	page 12	Medical lab	page 59
Trees in a Circle	page 14	Longest Word Chain	page 61
Subway	page 15	Gifts	page 63
One Hour One Task	page 16	Simon Debugging	page 65
Rubbish Robots	page 18	Arrows	page 68
Three Friends	page 20	Ballroom Floor	page 70
Bird Colours	page 21	Optical Fibre	page 72
Lemonade Party	page 23	Buried Treasure	page 74
Email Message	page 25	Truth or Lies	page 76
Flowers	page 27	Secret Secrets	page 78
Roomsharing	page 29	Scheduling Rehearsals	page 80
Infinite Ice-cream	page 31	Switch On	page 82
Mutation of an Alien	page 33		
Beaver Land	page 35		
Arrow Maze	page 37		
Colouring In	page 39		
Shortish Program	page 41		
Board Jump	page 43		
Elevator	page 44		
Passcode	page 46		
Rows and Columns	page 48		
Toll Roads	page 50		

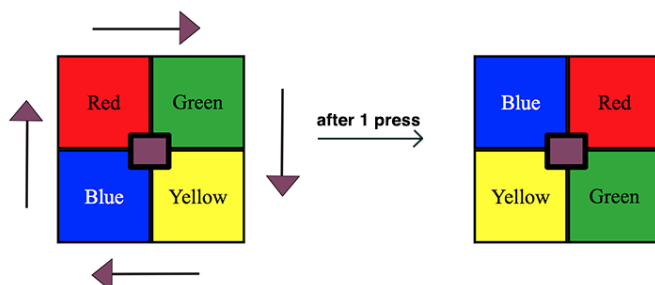
Matrix

2019 Questions	Years	Level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Spinning square	3-4	A	X	X	X		X	X
Shoe prints	3-4	A	X	X	X		X	
Honey pot	3-4	A	X		X	X	X	
Pizza	3-4	A	X		X		X	
Getting dressed	3-4	A	X		X		X	X
Trees in a circle	3-4	B	X		X	X	X	
	5-6	A						
Rubbish robots	3-4	B	X		X	X	X	X
Anthill Scramble	3-4	B	X		X		X	
	5-6	A						
Snake Samba	3-4	B	X	X	X		X	
	5-6	A						
Rainbow Parrots	3-4	B	X	X	X		X	X
	5-6	B						
Subway	3-4	C	X		X	X	X	X
	5-6	A						
Three Friends	3-4	C	X		X	X	X	
	5-6	B						
	7-8	A						
Crocodile Scales	3-4	C	X	X	X		X	
	5-6	B						
Flowers	3-4	C	X	X	X		X	
	5-6	B						
Colouring in	3-4	C	X	X	X	X	X	X
	5-6	C						
	7-8	A						
Shortish program	5-6	A	X	X	X	X	X	X
	7-8	A						
Ring toss	5-6	B	X		X		X	

2019 Questions	Years	Level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Lemonade party	5-6	C	X		X	X	X	
	7-8	A						
Room sharing	5-6	C	X		X	X	X	X
	7-8	B						
	9-10	A						
Tree lopping	5-6	C	X		X	X	X	X
	7-8	A						
Elevator	5-6	C	X		X	X	X	
	7-8	B						
	9-10	A						
One hour one task	7-8	B	X		X	X	X	X
	9-10	A						
Arrow maze	7-8	B	X		X	X	X	
Park walk	7-8	B	X		X	X	X	X
	9-10	A						
Mutation of an alien	7-8	C	X		X	X	X	
	9-10	B						
	11-12	A						
Passcode	7-8	C	X		X	X	X	X
	9-10	B						
	11-12	A						
Toll roads	7-8	C	X		X	X	X	X
	9-10	B						
	11-12	A						
Optical fibre	7-8	C	X		X	X	X	
	9-10	B						
Medical lab	7-8	C	X		X		X	
	11-12	B						
Infinite ice cream	9-10	A	X	X	X		X	

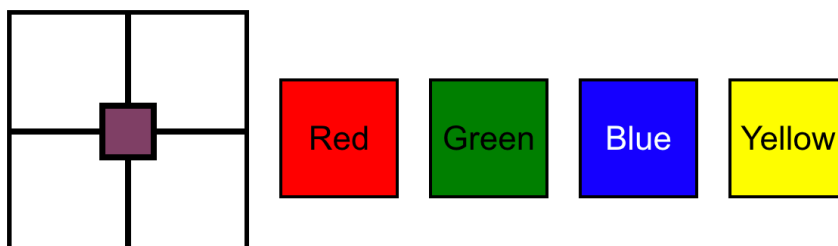
2019 Questions	Years	Level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Signup debugging	9-10	B	X		X	X	X	X
	11-12	A						
Rows and columns	9-10	C	X		X	X	X	X
	11-12	B						
Flipping cards	9-10	C	X	X	X	X	X	
Connections	9-10	C	X		X		X	
	11-12	B						
Arrows	9-10	C	X			X	X	X
	11-12	B						
Switch on	9-10	C	X		X	X	X	
	11-12	C						
Box jump	11-12	A	X				X	X
Treasure maps	11-12	B	X		X		X	X
Longest word chain	11-12	C	X		X	X	X	X
Gifts	11-12	C	X		X	X	X	
Ballroom floor	11-12	C	X	X	X	X	X	X
Buried treasure	11-12	C	X		X	X	X	

Here is a simple push square game. The colours rotate every time the centre button is pressed:

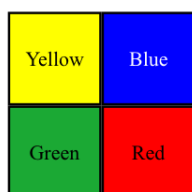


Question

If we press the button one more time, where would the red, blue, green and yellow squares be? Drag and drop the colours into their correct places.



Answer:



It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR) Abstraction (AB), Algorithms (AL), Evaluation (EV)

Concepts – Abstraction (Ab), Algorithms (AI)

This problem involves following a series of steps and keeping track of:

- the current state
- whether that state is what the current orientation looks like
- or what is in the top-left corner
(which can be used to determine the position of the other three colours).



Shoe Prints

Year 3+4: A

Year 9+10

Year 5+6:

Year 11+12

Year 7+8:

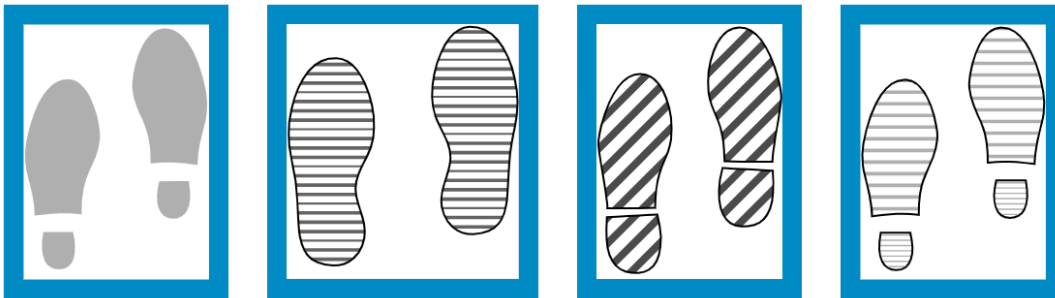
Four footprints have been found in the mud.

The police are looking for a robber who wore shoes with these properties:

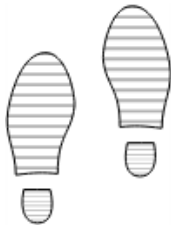
- The soles have a stripy pattern
- The heel is thin

Question:

Choose the set of shoe prints that could belong to the robber.



Answer:



Explanation:

If you look at all the shoe prints in the question, only the shoe print shown above has both thin heels and a stripy sole.

It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR), Abstraction (AB) Algorithms (AL)

Concepts - Abstraction (Ab), Data Interpretation (Di), Specification (Sp), Algorithms (Al)

It is important to be able to see patterns in computer science. In this task, it is best to find a solution by comparing a defined pattern with other patterns. Similar processes are also used in areas such as pattern recognition and image detection in Computer Science.

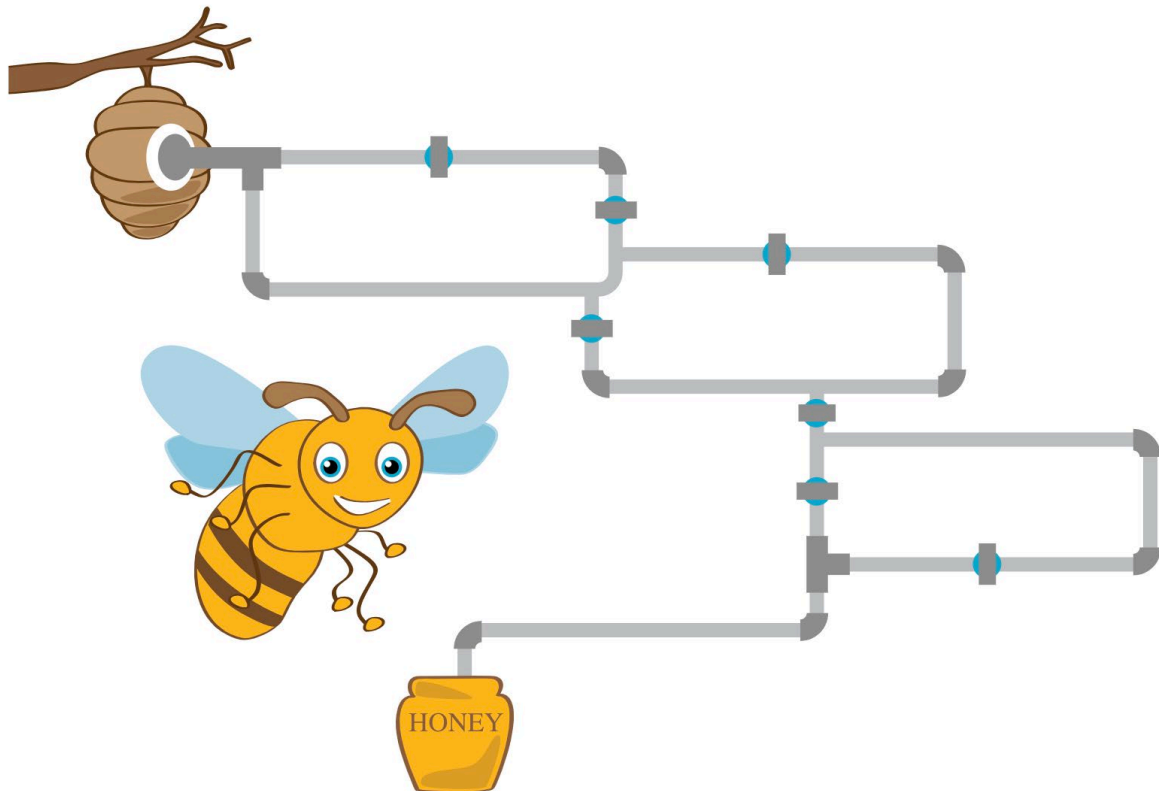
Bertie the bee has to fill the honey pot as quickly as he can.

Bertie can open and close the taps () in the pipes below by clicking on them.

By doing this he can control which way the honey flows.

Question:

Click on the taps to make the honey reach the pot by the shortest possible path.



Honey Pot

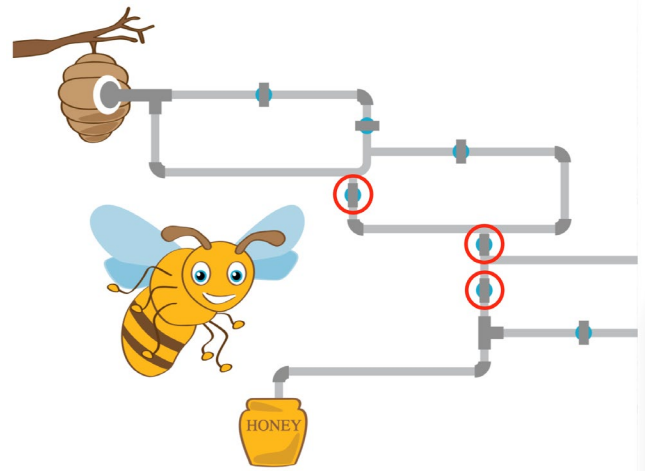
Answer:

The image on the right shows the three taps that need opening to allow the honey to flow along the shortest path. All of the other taps should be closed.

Explanation:

This task requires the students to analyse a model and solve two problems:

- find the shortest path in a network of pipes
- work out how to arrange a series of valves to ensure this path is followed.



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL),

Concepts - Abstraction (Ab), Data Representation (Dr), Algorithms (Al), Digital systems (Ds)

Computers are composed of various chips which are made of even smaller parts: electronic circuits. Electronic circuits are in turn composed of logic gates. Logic gates act like valves, except that instead of water they conduct electricity and instead of pipes they have wires. This means that our modern electronic devices (including complex ones like computers and smartphones) are built up from simple logical operations.

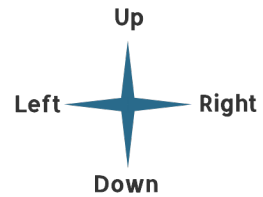


Year 9+10

Year 11+12

Year 3+4: B	Year 9+10
Year 5+6: A	Year 11+12
Year 7+8:	

Edna the echidna wants to go to the anthill. To get there she must collect all the red ants (larger ants). Help her by watching her walk straight and telling her where to walk (Right, Left, Up and Down as shown below).



Question

Which commands should you give Edna?

Right, Up, Right, Down, Right, Up, Right

Right, Up, Right, Up, Right, Up, Right, Down, Right, Down, Right

Right, Up, Right, Up, Left, Down, Right

Right, Up, Right, Down, Right, Down, Right

Answer

A

Explanation

A) Edna walks right from the starting point. When the straight path ends, she walks up, and walks all the way to the top before going right. When she catches the next red ant, she goes down again until the line ends. Next she needs to go right, then up, and when she reaches the red ant she needs to walk right again.

B) can't be right as the path described does not collect all red ants.

C) and D) can't be right as the paths described do not follow the line of ants and rule stated.

The only right answer can be A as it follows the rules stated.

It's Computational Thinking

CT Skills - Decomposition (DE), Abstraction (AB), Algorithms (AL)

Concepts - Abstraction (Ab), Data Representation (Dr), Specification (Sp), Algorithms (Al)



Snake Samba

Year 3+4: B

Year 9+10

Year 5+6: A

Year 11+12

Year 7+8:

Sally the snake is coming up with a dance.



Question:

Which image comes next in the dance?



Answer

C

Explanation

A, B & D can't be right as it does not follow the pattern:

- In the images the tail of the snake change the positions in each dance step: it is either a straight or bent after the other.
- The small black stripe is rotatory over or under the wide black stripe.
- In each second dance step the snake rotate (90 degrees) in a clockwise direction.

The next dance step the head of the snake looks in new direction (down), with the tail remaining unmoved (straight), and the black small stripe moves over the wide black stripe.

The only right answer can be C as it follows the same rules.

It's Computational Thinking

CT Skills – *Decomposition (DE), Pattern Recognition (PR) Abstraction (AB), Algorithms (AL)*

Concepts - *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al)*

Data can take many forms, for example, pictures, text or numbers. When we look at data in this question, we are looking for a sequence of images that will assist in solving the problem. In the

sequence each image has attributes and some of them will be changed on the next image in the sequence and some of them only on each second, 3rd, ... image. By identifying these changes we can make predictions, create rules and solve more general problems.



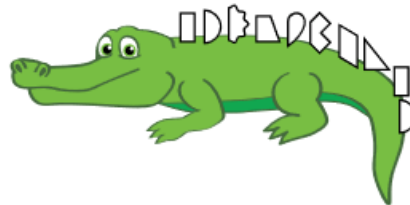
Crocodile Scales

Year 3+4: C
Year 5+6: B
Year 7+8:

Year 9+10
Year 11+12

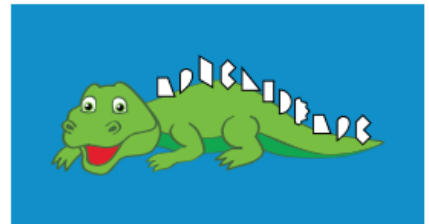
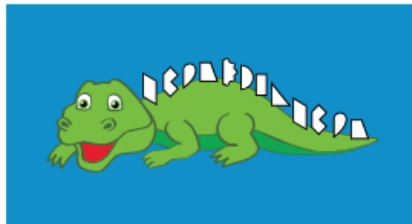
Cheryl and Charlie Crocodile belong to the same family of crocodiles that always have the same sequence of scales on their spine.

Sometimes they lose a scale, but the scales can regrow.



Question

Select **all** crocodiles that are from the same family as Cheryl and Charlie.



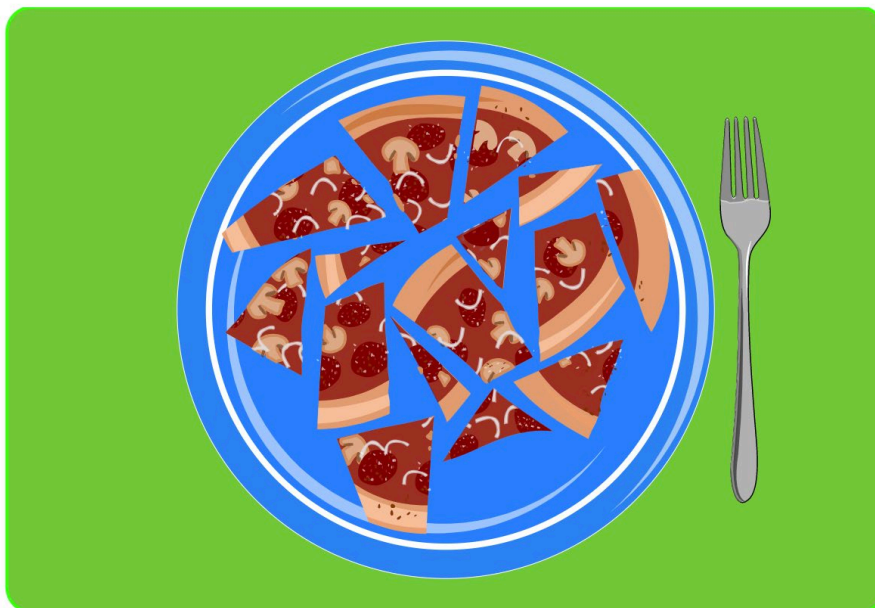
Lucilla is learning how to eat pizza. Here are her Mum's rules:

Pieces with crust should be eaten with hands.

Pieces without crust should be eaten with a fork.

Question:

Select all of the pieces of pizza that Lucilla should eat with her fork.



Answer:

The pizza pieces to be selected are the three pieces without crust, as shown on the right.



It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR) Abstraction (AB),

Algorithms (AL)

Concepts - Abstraction (Ab), Specification (Sp), Algorithms (AI),

For each of the pizza pieces, Lucilla needs to perform a simple test of whether it has a crust or not.

When designing computer programs, there are often points that require the program to make choices. In programming, a choice like this is known as a selection, and is programmed using IF statements.

Getting Dressed

Year 3+4: A Year 9+10
Year 5+6: Year 11+12
Year 7+8:

These are Bruno's clothes:

shirt	vest	trousers	underwear	braces	socks	shoes
						

Beaver Dad carefully arranges little Bruno's clothes, into four piles.

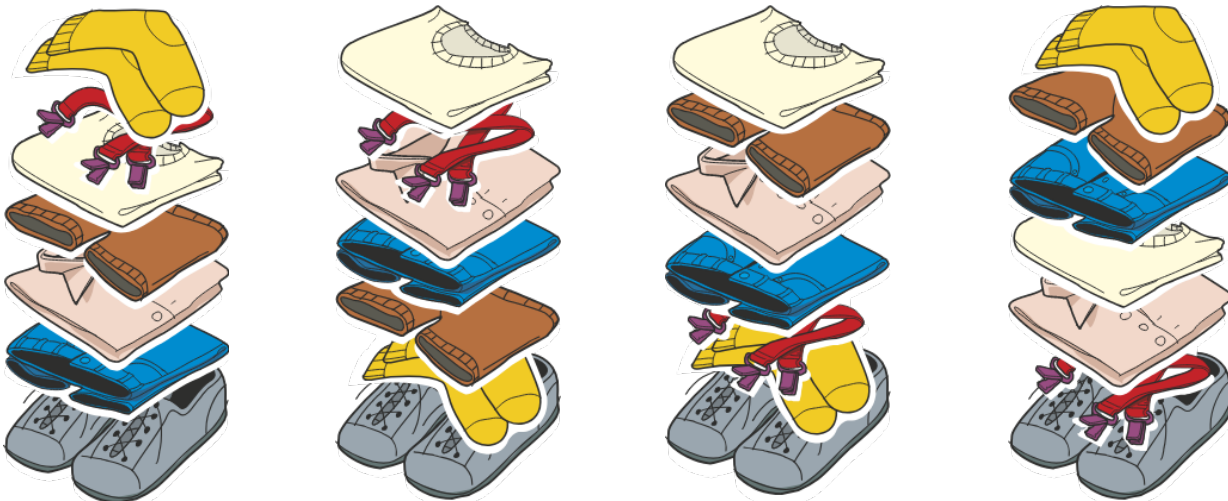


Bruno puts on his clothes in the order that they are in the pile, starting from the top.

Bruno wants to wear the braces over his shirt.

Question:

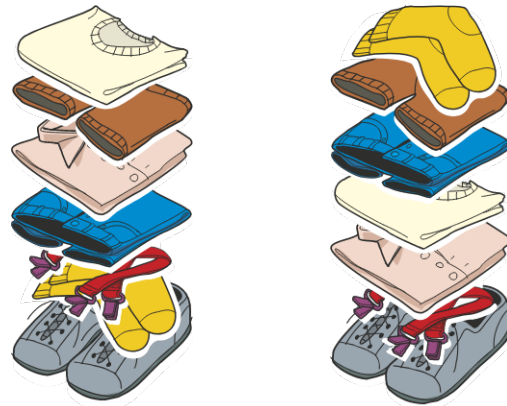
Choose a pile of clothes that Bruno will be happy with.



Getting Dressed

Answers:

There are two possible correct answers:



Explanation:

To find the solution we should look at the piles carefully starting from the top item and check that the order is correct according to the rule: shirt before braces.

The other two piles are wrong because the braces would be put on before the shirt.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data representation (Dr), Specification (Sp), Algorithms (Al)

This is an example of a constraint: "Before you can go into a room to open your toy box you have to open the door." The door must be open before you can go in to get what you want. This task can be solved by checking which lists satisfy the constraint: "Item A must be put on before item B." If the list satisfies the constraint, then it is correct, otherwise it is not correct.

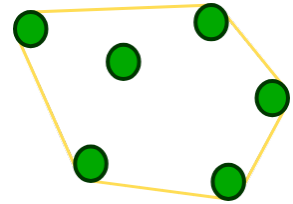
Checking if lists or objects satisfy constraints is a very common problem in computer science, and can be difficult for complicated objects such as apps and computer games.

When the ordering constraints apply only to some items in a list (such as only to a shirt and braces) we call this a partial ordering.

Example:

The green circles on the right show the position of six trees. Joni has tied a rope around them shown by the yellow line.

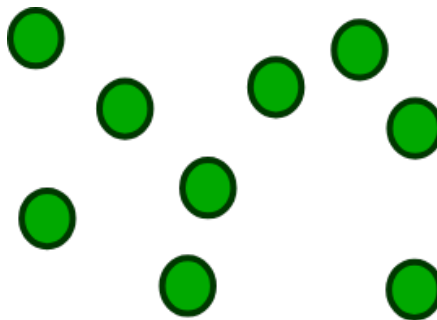
Only 5 trees are touched by the rope.



Question:

If the trees shown below are wrapped with a rope in the same way, how many trees will be touched by the rope?

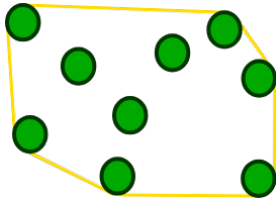
4 5 6 or 7



Trees in a Circle

Answer and explanation:

Six trees are touched by the rope as shown here:



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp)

This problem is known as finding the convex hull of a set of points. One way to define this is the polygon with the smallest area which contains all the points in the set. The word convex means extending outward or curving out, and any two points within a convex shape, when drawn on paper, can be connect with a straight line which is inside of the shape. The word hull here is used to describe the "shell" or "encasement", as in the hull of a ship or the hull of a seed, such as corn.

Finding the convex hull of a set of points is used in a variety of computation settings:

pattern recognition: is there a face in an image

processing written text: is that handwritten character the letter B?

geographic information systems: what is the size of a floodplain or river system?

packaging: what is the least amount of packing required to tightly wrap a three-dimensional object?

There are many algorithms to solve this problem, and solving this problem efficiently is a very practical and useful application in computer science.



Subway

Year 3+4: C

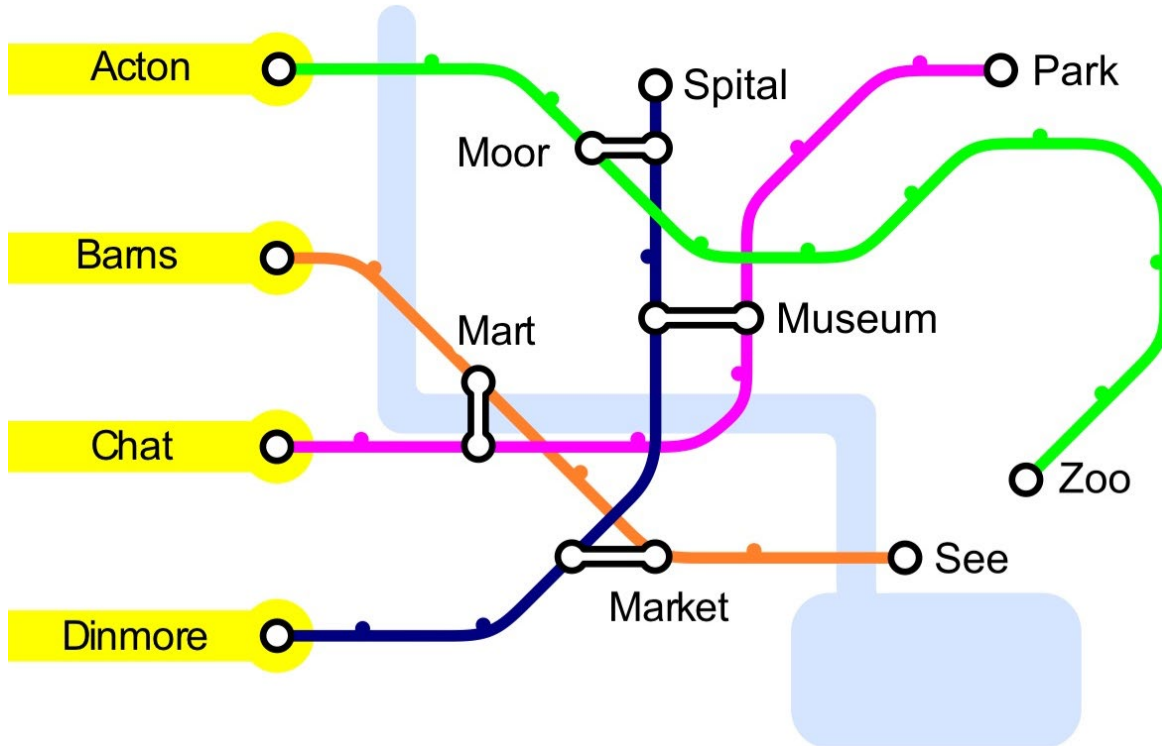
Year 9+10

Year 5+6: A

Year 11+12

Year 7+8:

A train system consists of four train lines that start at the stations:
Acton, Bams, Chat, and Dinmore.



There are also four interchanges: Moor, Museum, Mart and Market.
Interchanges allow passengers to change from one train line to another.

John went to the Zoo. He changed train lines exactly once.

Question:

At which station did John start his journey?

Acton Bams Chat or Dinmore

Answer:

The answer is Dinmore.

Explanation:

No transfer is needed if John starts at Acton.

2 transfers are needed if John starts at Bams.

2 transfers are needed if John starts at Chat.

Only 1 transfer is needed if John starts at Dinmore.

It's Computational Thinking:

CT Skills - *Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)*

Concepts - *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al)*

To solve a common sudoku exercise, we must look at all of the rows and all of the columns.

One way to do this is to look at all the rows first, and then all the columns.

Another way to do this is to look at all the columns first, and then all the rows.

If you find an incorrect row or column, it means that the box is incorrect, so you can immediately eliminate the box from the possible answers.

In life, elimination can help you to solve many problems. For example, when answering a question with multiple choice answers in a test, you can eliminate incorrect ones to minimise the possible choices.



The beavers have a robot that can perform many tasks.

In any one hour, the robot can work on only one task. At the end of each hour it checks if it has received a new task:

If yes, the robot must begin work on that new task immediately.

If no, the robot continues working on the task which has been left the longest.

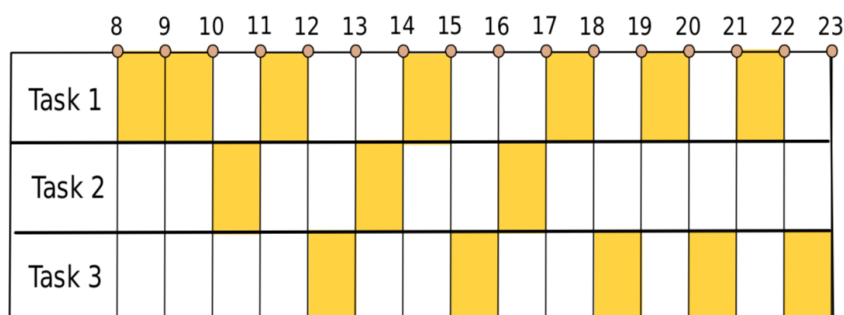
Example:

The following timetable shows an example of the robot's work in one day.

Task 1: a 7-hour task, received at 8:00

Task 2: a 3-hour task, received at 10:00

Task 3: a 5-hour task, received at 12:00



The robot finished Task 1 at 22:00, Task 2 at 17:00, and Task 3 at 23:00.

Question:

The robot was given 4 new tasks:

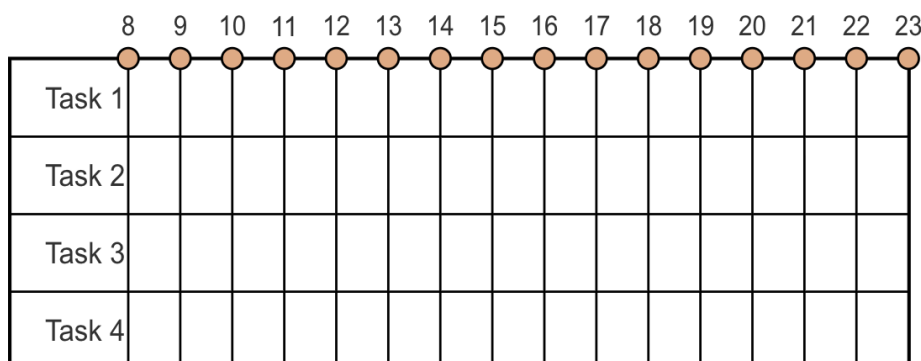
Task 1: 5-hours long, received at 8:00

Task 2: 3-hours long, received at 11:00

Task 3: 5-hours long, received at 14:00

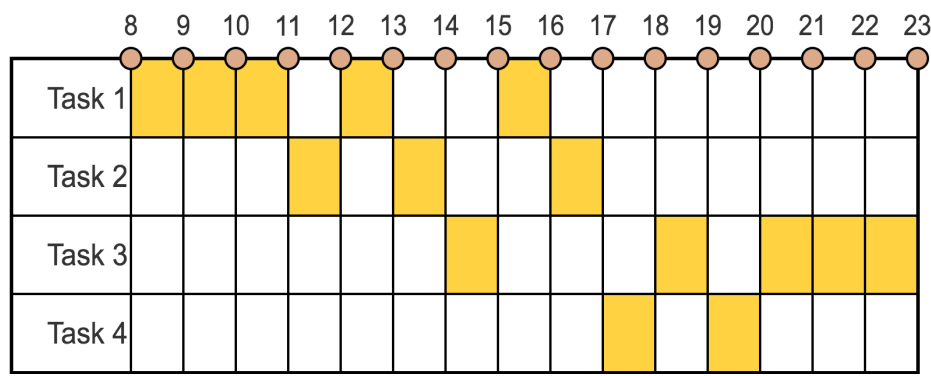
Task 4: 2-hours long, received at 17:00

Show the schedule the robot will follow by clicking on the appropriate cells below.



One Hour One Task

Answer:



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Digital Systems (Ds), Interactions (In)

When solving this we are simulating the use of a time line chart according to the given rules or procedure.

The problem tries to expose the actual process management system based on Round Robin Scheduling and the use of Gantt's Chart (the activities versus time line).



Rubbish Robots

Year 3+4: B

Year 9+10

Year 5+6:

Year 11+12

Year 7+8:



The picture shows the map of a park divided into rectangles. The number in each rectangle represents the number of pieces of rubbish left there by visitors.

The park keepers have two robots, Anton and Boris, who collect all the rubbish they find in every rectangle they enter.

The robots were given the following instructions:

First, robot Anton was sent: \uparrow = upwards \uparrow = upwards \leftarrow = left

Next, robot Boris was sent: \uparrow = upwards \uparrow = upwards \leftarrow = left

1	3	1
0	2	6
0	1	3
1		
Anton		Boris

Question:

How many pieces of rubbish will Boris collect?

3 9 11 or 12

Rubbish Robots

Answer:



9



Explanation:

Anton will go first and he will collect $1 + 2 + 0$ pieces of rubbish, which is 3 altogether, using the instructions given.

Next, Boris will collect $3 + 6 + 0$ pieces of rubbish, which is 9 altogether, using the instructions given.

Note that in the last square on Boris's path there used to be 2 pieces of rubbish before both robots started but Anton cleans this square first.

1	3	1
	0	6
0	0	3
1		
Anton		Boris

1	3	1
0	2	6
0	1	3
1		
Anton		Boris

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data Collection (Dc), Data Representation (Dr), Data Interpretation (Di), Algorithms (Al)

In computer science and robotics, it is often important to know in which order instructions in a program are processed (or how machines do their jobs).

We need to think about how the actions of our program or robot affects the environment. In this task it is how many pieces of rubbish will be left in each square of the park after Anton and Boris have followed the instructions they have been given.

Another goal might be to create an efficient program, for example, not to send a robot to collect rubbish from somewhere where there is no rubbish to be collected.

Three friends want to meet.

The map below shows where they are at the moment.

Bob is with his bike, Alice is on her skateboard and Jenny is with her scooter.

They want to meet at either the square, triangle, circle or diamond.

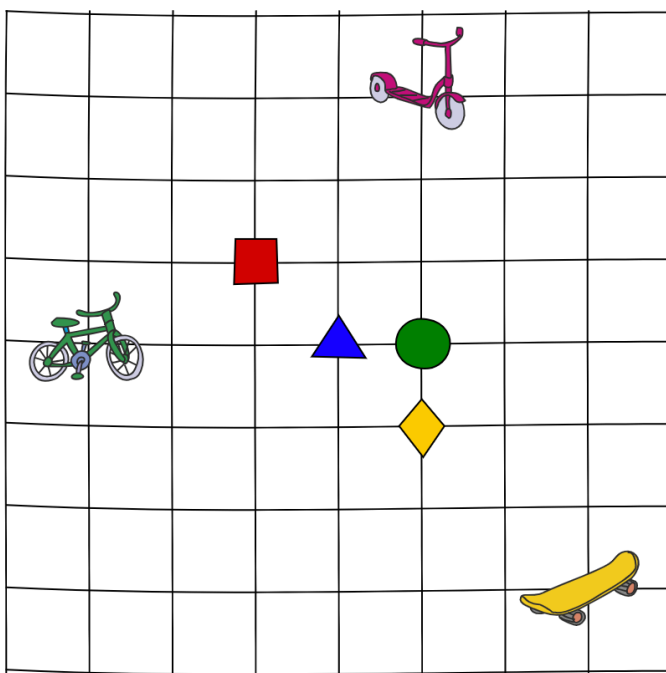
Distances are measured by: Adding up the distance travelled horizontally and vertically along the gridlines.

Example:

The distance from Alice (on her skateboard) to the blue triangle is 6.

Question:

Which meeting place should they choose so that the total distance the three friends must travel is the shortest possible?



Three Friends

Answer:

They meet at the Green circle.

Explanation:

The total distance from their homes to the red square is: $4 + 3 + 8 = 15$.

The total distance from their homes to the blue triangle is: $4 + 3 + 6 = 13$.

The total distance from their homes to the green circle is: $3 + 4 + 5 = 12$.

The total distance from their homes to the yellow diamond is: $4 + 5 + 4 = 13$.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL),

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al)

In Computer Science, local search can be used on problems that can be formulated as finding a solution maximising a criterion among a number of candidate solutions.

Local search algorithms move from solution to solution in the space of candidate solutions (the search space) by applying local changes, until a solution deemed optimal is found or a time bound is elapsed.

To find the best meeting point in this task, add all the distances covered by the three characters to each meeting point. The smallest of them shows the correct answer. Using this method you are using a local search algorithm.



Ring Toss

Year 3+4:

Year 9+10

Year 5+6: B

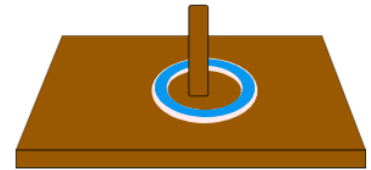
Year 11+12

Year 7+8:

Sarah is playing the game *Ring Toss*.

For each round she has 5 rings to try to throw around a peg.

Every ring that successfully lands around the peg scores points, but not every throw is worth the same number of points:



Throw	Points
1st throw	5
2nd throw	4
3rd throw	3
4th throw	2
5th throw	1

Any ring that misses the peg scores 0 points.

Question:

Sarah threw her five rings as shown.
How many points did she get?

Answer:

Sarah got 6 points.



Explanation:

Toss	ring	Hit / Miss	points
first toss	yellow	Miss	0
second toss	blue	Hit	4
third toss	red	Miss	0
fourth toss	green	Hit	2
fifth toss	black	Miss	0
TOTAL			6

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Algorithms (AL),

Concepts - Abstraction (Ab), Data Collection (Dc), Data Interpretation (Di), Specification (Sp), Algorithms (Al)

The rings represent data packets managed in a data structure called a stack. This problem is a sequencing problem. A computer analyses data in sequence. Data needs to be organised so that it can be processed to help determine solutions.

The Rainbow Parrot, has had four chicks.



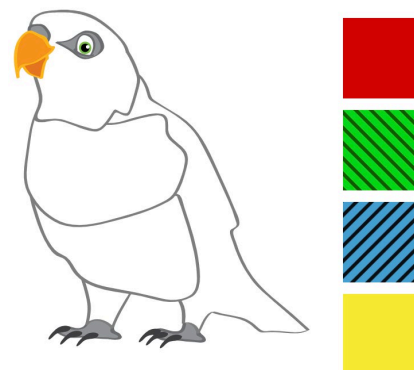
Each young parrot has four colours: red, blue, green and yellow.

Each colour in a parrot must be in a different part of the body to the other parrots.

Question:

Based on the first three chicks, what colour patterns will the fourth have?

Drag and drop the colour patterns onto the body on the right.



Answer:



Explanation:

Other combinations of colour patterns cannot be right as they do not follow the progressive colour pattern: red, blue, green and yellow.

Each colour fills in only one body part of the Lorikeet: head, chest, wing, tail. The only right answer can be the one above as it follows the colour and body part pattern established.

It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR), Abstraction (AB) Algorithms (AL), Evaluation (EV)

























Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al)

Janet made 37 litres of lemonade at home and now she needs to put it in bottles to take it to school for a party.

She has several empty bottles of various sizes, but she wants to use the smallest number of bottles AND all of the bottles have to be full.

Question:

Click on the bottles that she should use.

1 litre				
2 litre				
4 litre				
8 litre				
16 litre				
32 litre				

Lemonade Party

Expected Answer:

The best solution is a 32 litre bottle, a 4 litre bottle and a 1 litre bottle.

Explanation:

This is a binary counting problem. Note that it never makes sense to select multiple bottles of a single size; if you select 2 bottles of 8 litres, it makes more sense to select a single bottle of 16 litres.

There are therefore a minimum of three bottles required.

Alternative Answers:

As the question did not stipulate that the bottles had to be full it is possible to take the 37 litres in two partially filled bottles in one of these three combinations:

32 litre and 32 litre

32 litre and 16 litre

32 litre and 8 litre

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL),

Concepts - Abstraction (Ab), Data Representation (Dr), Specification (Sp), Algorithms (Al)

The binary number system is central to computation and to computers. The volume of the bottles keeps doubling in this task, just like the values of individual bits in a binary number. In informatics it is important to be able to convert numbers in one base (i.e. base ten) to another base (i.e. binary). When converting to binary, a simple trick is to select the largest bit that still fits in the number you are trying to convert.

To solve the question above, you would reason through the following steps:

You have 37 litres to put into bottles

The largest bottle that fits for 37 litres is size 32

Now you have 5 litres left

The largest bottle that fits for 5 litres is size 4. 5. Now you have 1 liter left

The largest bottle that fits for 1 is size 1

$32+4+1=37$, so you are finished.

A flower shop sells the following types of flowers:



gladiolus



lily



tulip



rose

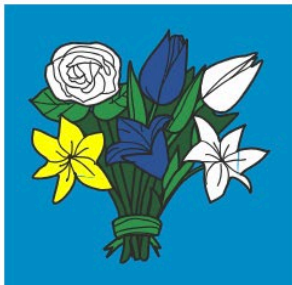
The flowers come in white, yellow or blue.

Clara wants a bunch of six flowers. She tells the florist:

- There must be two of each of the colours yellow, white and blue
- Flowers of the same type must not have the same colour
- There should be no more than two of each type of flower.

Question:

Which of the following bunches will Clara be happy with?



Flowers

Answer:



Explanation:

Wrong answers:

In this bunch there are three white flowers:



In this bunch there are three roses:



and in this bunch there are two flowers of the same type that have the same colour:



It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR), Abstraction (AB), Algorithms (AL)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al)

Computer science often involves solving problems that are specified by a set of constraints. The task is to find a solution that satisfies all these constraints or as many as possible.

One can consider more complex tasks where the constraints are combined by logical operators like conjunction (A and B means both constraints A and B have to be satisfied, like the three rules in our task) or disjunction (A or B means satisfying just one of them is enough).

The members of the Girls' Computer Club are planning a weekend trip.
 They will stay in a hostel with large rooms, that can take a maximum of six guests each.
 But who will share rooms with each other?

Each girl submits her room sharing wishes on a card saying:

- which other girls she absolutely wants to share a room with (+)
- which other girls she definitely does not want share a room with (–)

The club president wants to keep all members happy. So she must assign the girls to rooms and fulfil all their room sharing wishes.

Question:

Help the club president assign the girls to their rooms, drag the wish-cards into one of the rectangles.
 (The rectangles represent different rooms.)

<u>Emma</u> +: -: Alina	<u>Lara</u> +: -: Emma	<u>Alina</u> +: Lilli -:	<u>Mia</u> +: Emma, Zoe -:	<u>Lilli</u> +: -: Lara	<u>Zoe</u> +: -: Alina
-------------------------------	------------------------------	--------------------------------	----------------------------------	-------------------------------	------------------------------

Room Sharing

Answer:

One possible answer is:

<u>Emma</u> +: -: Alina	<u>Mia</u> +: Emma, Zoe -:	<u>Zoe</u> +: -: Alina	
<u>Alina</u> +: Lilli -:	<u>Lilli</u> +: -: Lara		
<u>Lara</u> +: -: Emma			

Explanation:

To fulfil all wishes, you could focus on the positive wishes:

1. If there is an empty room,
 - a. then drag a girl's card to this room
 - b. else drag a girl's card to a room where this girl is not on a – list of the cards already in this room (and
FAIL if there is no such room)
2. Drag all cards of girls who are on the + list of the first card to the same room. If there is no more card, then you are DONE.
3. Go back to step 1.

Applying the above procedure results in this room assignment. There is no other assignment that would fulfil all wishes: Alina must be together with Lilli. But Alina (and hence also Lilli) can be together with neither Emma nor Zoe; hence she also cannot be together with Mia, who must be together with Emma and Zoe. Lara must go into her own room, as she can be with neither Lilli (and hence Alina) nor Emma (and hence Mia and Zoe).

Room Sharing

It's Computational Thinking:

CT Skills - *Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)*

Concepts - *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Interactions (In)*

In order to solve the problem of keeping all girls happy, the club president is looking for an assignment of girls to rooms that is in accordance with all preferences. Such an assignment is a solution to the president's problem.

But how to find a solution? A simple way is to look at all imaginable assignments, one after the other, and see whether the preferences are satisfied. But as there are many possible assignments, this might take quite some time. It is more efficient to consider the preferences at first hand and to construct an assignment that satisfies the given constraints and, therefore, is a solution.

Constraint satisfaction problems like the "girl-to-room-assignment" problem are quite well known in computer science. They may appear, like in this task, when resources need to be used without conflicts – for instance, when many airplanes need to be scheduled to use the take-off or landing strips of an airport. Computer systems, using efficient algorithms, are able to solve such problems efficiently.

There are two ice-cream sellers. They use the same four flavours:



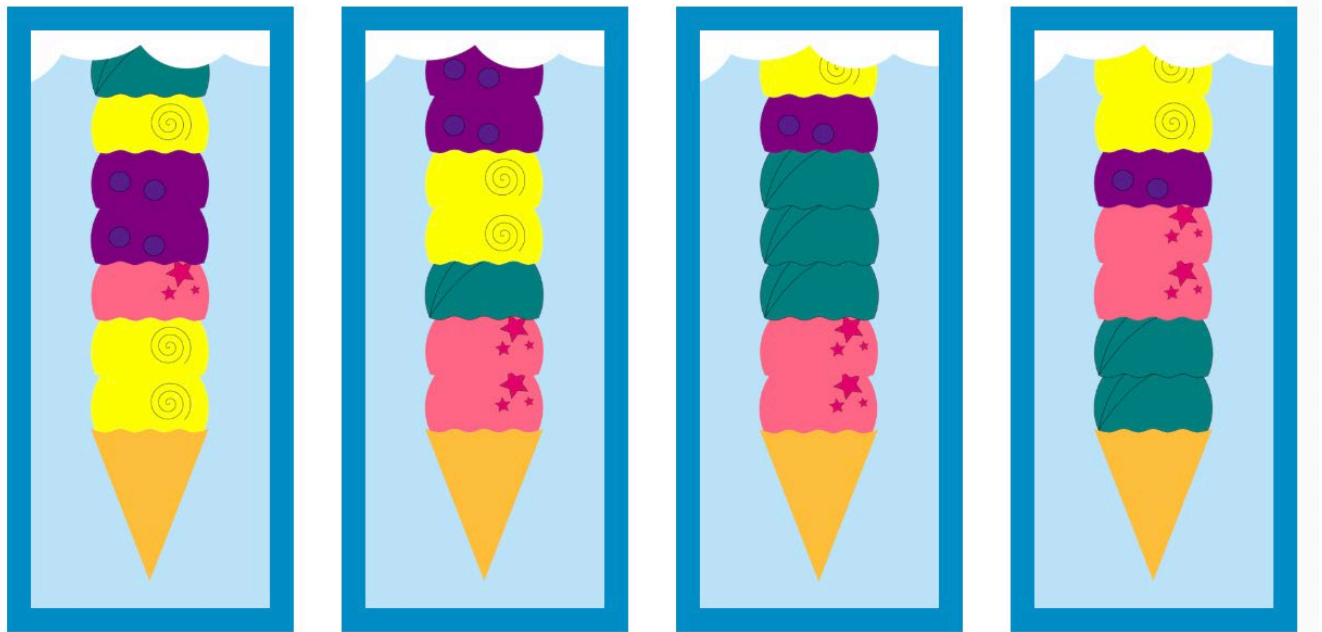
The first seller uses the following instructions to make ice-cream:

1. Start with an empty cone.
2. Pick a flavour at random, and add two scoops of that flavour.
3. Add one scoop of any different flavour.
4. If the requested height is reached, stop, otherwise go to Step 2.

The second ice-cream seller does not follow any instructions.

Question:

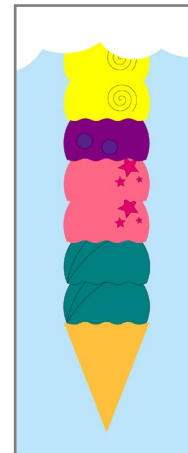
You can only see the first few scoops of the ice-cream cones below. Which one is certainly from the second seller?








Infinite Ice-cream

Answer:

The only cone that clearly does not follow the instructions is:



Explanation:

It starts off correctly by placing two of the same flavour   followed by one of a different flavour  but then it adds two scoops of different flavours   when it should have added two scoops of the same flavour.

The other cones follow the instructions of first ice-cream seller, *at least as far as we can see*.

It's Computational Thinking:



CT Skills - Decomposition (DE), Pattern Recognition (PR), Abstraction (AB), Algorithms (AL)

Concepts – Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di)

Patterns in ice-cream cones, or words, or images, can be created by short lists of instructions. Recognising patterns, and recognising where patterns break, are everyday jobs for computer scientists.

Sometimes these patterns are repeating, for example:



which is a simple pattern that just repeats  

These are easier to spot. This task is slightly more difficult, because the pattern is not repeating.

There is also a trap a computer (or a computer scientist) may fall into: Instructions may sometimes seem like they are being followed just by accident. Indeed, the second machine does occasionally choose the flavours randomly in a way which seems to follow the instructions. You may recognise that the instructions are being broken. But, just by observation, you can never be sure that they are followed. Luckily, in this task we knew for sure that only one of the ice-creams is from the second machine.



Mutation of an Alien

Year 3+4:

Year 5+6:

Year 7+8: C

Year 9+10: B

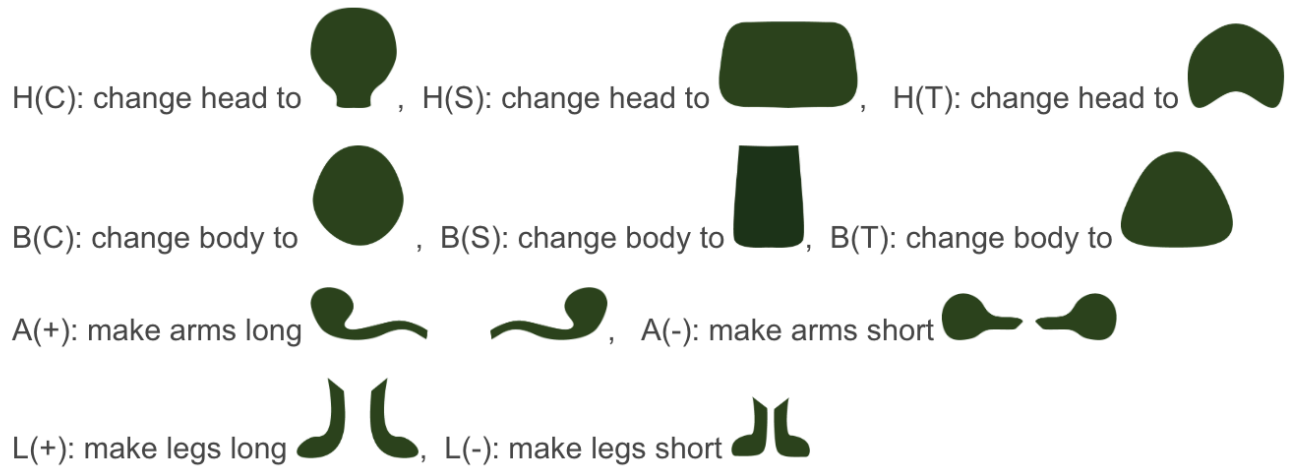
Year 11+12: A

An alien has a head, a body, two arms, and two legs.

The alien can be transformed through the following mutation commands:

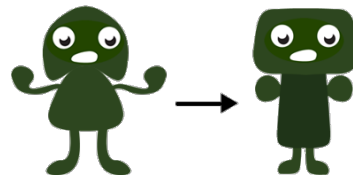
(It is possible that the shape of a part is mutated more than once.)

Mutation Commands



Example:

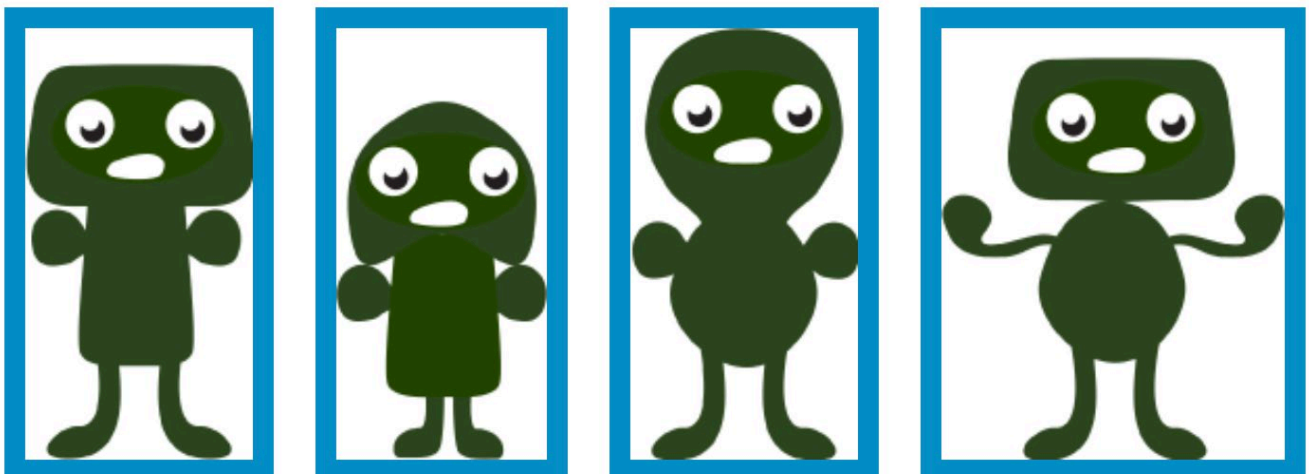
Transformation example for H(S), B(S), A(-), L(-):



Question:

What will the alien look like after receiving the following commands?

H(T), L(+), B(T), A(+), H(C), A(-), B(C)



Mutation of an Alien

Answer:



Explanation:

For each part of the alien, the last mutation command will overwrite the result of the previous one. Therefore, the final result is a head with circle shape, a body with a circle shape, short arms, and long legs.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL)

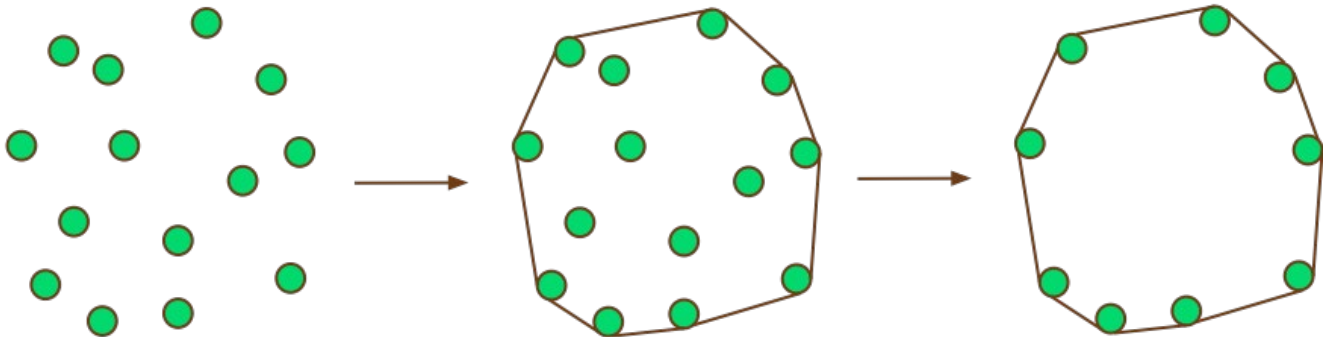
Concepts - Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al)

When performing a program, the instructions are executed in a sequence.

The head, body, arms, and legs are like the variables or functions used in a program.

The setting of the shapes: C for circle, S for square, and T for triangle is like the value assigned to variables or is like the parameter passed to a function.

Beavers surround their village with a rope around the trees on the outside of the village. They then cut down the trees that are not needed to support the rope:



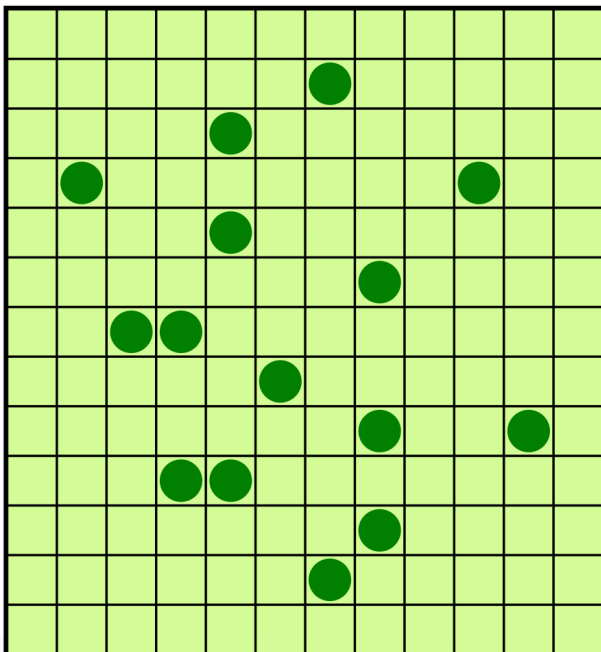
First they make a map the trees on a grid of squares like the one shown below. Then they select the smallest number of trees needed to support the rope.

Notes:

- Trees are shown as green circles.
- To make this easier, the beavers assume that all trees have the same diameter (thickness) and are in the centre of the squares in the grid.

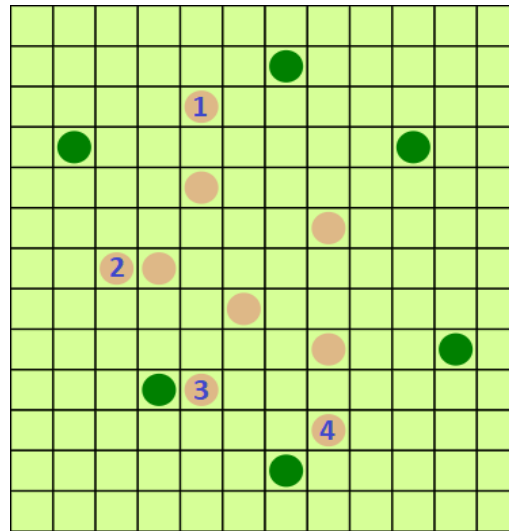
Question:

Select all the trees that must NOT be cut down by the beavers in the plan below.



Tree Lopping

Answer and Explanation



The main problem is to decide whether a tree is inside the boundary, on the boundary or on the boundary but not required to support the rope.

The trees that can be cut are shown below as light brown circles. We can cut most of the trees without much thought. A few trees (with blue numbers) still require some analysis. Because of the grid system, whether the tree is needed to support the rope can be worked out by counting boxes and looking at triangles:

- Trees 1,4,and 3 are therefore just inside the boundary.
- Tree 2 can be seen to be on the boundary but is not required to support the rope.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Impacts (Ims)

The smallest convex polygon containing a finite set in a plane is called a "convex hull" of that set. This is an important idea and is usually solved with a mathematical algorithm. The grid provided in this task makes this a lot simpler for humans (or beavers) to solve.

The problem of finding convex hulls has practical applications in pattern recognition, image processing, statistics, geographic information system, game theory, construction of phase diagrams, and static code analysis by abstract interpretation. It also serves as a tool, a building block for a number of other computational-geometric algorithms such as the rotating callipers method for computing the width and diameter of a point set.



Arrow Maze

Year 3+4:

Year 9+10

Year 5+6:

Year 11+12

Year 7+8: B

Help Smilie get through the arrow maze and back to his house.

When standing in a square with an arrow, Smilie must move to the next square by following the direction of the arrow.

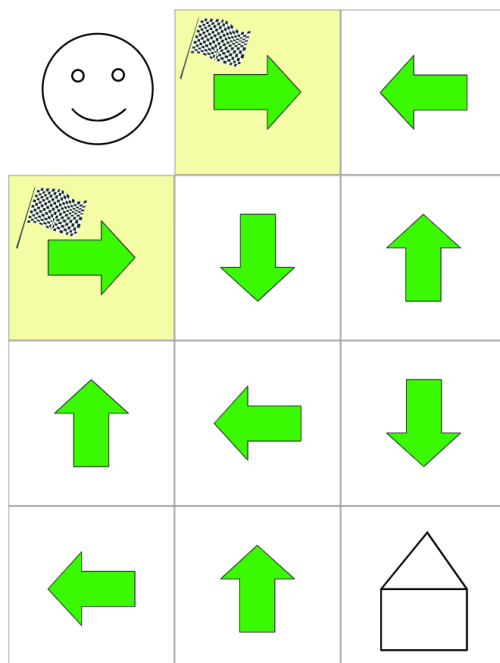
Smilie can start from either yellow square with a start flag.

At the moment it is impossible for Smilie to reach his house.

Question:

Change the direction of only one arrow so that Smilie can follow the arrows to his house.

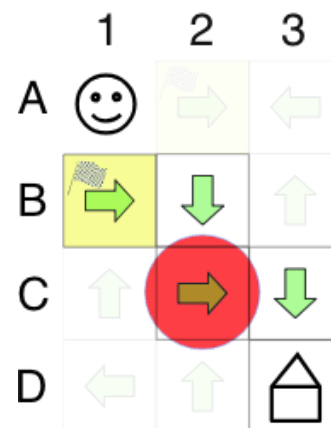
(Click repeatedly on an arrow to change which way it points.)



Arrow Maze

Answer:

The arrow to change is marked below with a red circle:



Explanation:

The path for Smilie is highlighted (start A1 – B1 – B2 – C2 – C3 – D3 Smilie's house).

Here is a proof that this solution is the only possible solution:

Starting from the target cell D3 and going backwards, Smilie can get to cell D3 from 2 directions: D2 and C3.

The arrow in D2 is not pointing at the target cell, so to produce a correct solution it needs to be changed. Because no neighbouring arrow is pointing at D2 a second arrow should be changed, which is not allowed. Therefore, the target is accessible only from the cell C3.

There is no arrow pointing at C3, so we need to change an arrow in cell B3 or C2. Because no arrow is pointing at B3, there is no way to go to cell C3 through B3 without changing the direction of another arrow.

Cell C2 is accessible from the starting cell (A1 – B1 – B2 – C2) without changing another arrow. We need only change an arrow in cell C2 to solve the maze.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL)

Concepts - Abstraction (Ab), Specification (Sp), Algorithms (Al)

To solve this task, you must understand how to behave in a maze and what the arrows mean. Even if finding a correct way through the maze is easy, proving that there is no other solution is difficult.

Backtracking is a good idea to go through all possible combinations effectively. Backtracking is an approach which takes candidates of solutions and abandons the candidates which cannot possibly be a valid solution.

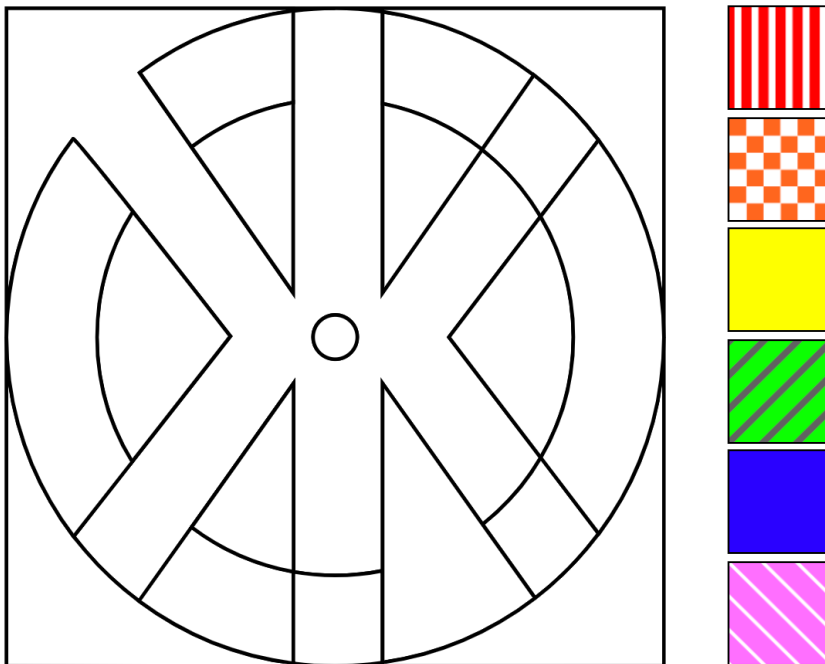
The pattern below needs colouring in!

There are two rules:

1. Use as few colours as possible
2. No two areas that share an edge can be the same colour.

You can try colouring in the pattern yourself.

To do this drag any of the 6 colours provided on to the pattern.



Question:

What is the minimum number of colours required?

Colouring In

Answer:

The task can be completed with only 3 colours.

Explanation:

There are many possible answers depending on which colour is used as a starting colour and which segment is chosen to colour in. Here is a solution created by choosing the first colour and filling in as many regions as possible starting in the top left corner:



The second colour is then used to fill in as many regions as possible starting now from the bottom left corner:



By selecting the third colour, it can be seen that all the remaining regions can be filled in while still obeying the required no-shared-edge rule:



This showed that three colours suffice. We can now see that we cannot solve the problem with less than three colours; consider the region marked with an "x", coloured in yellow: it shares an edge with two regions, so we have three regions that must all have different colours. Therefore, we cannot colour the pattern with less than three colours.



It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts – Abstraction (Ab) Data Interpretation (Di), Specification (Sp), Algorithms (AI)

This is a task that refers to the Four Colour Theorem, which states that given a two dimensional shape, such as a map, that is divided into any number of regions of any shape, no more than four colours is required to fill in the regions so that no two adjacent regions have the same colour.

Graph colouring has important applications in computing. For example, in scheduling of flight plans, in the assignment of flying corridors to aircrafts so as to avoid collisions during landing or taking off of aircraft, and in the assignment of frequencies for mobile networks.

The program shown on the right drives a triangular robot.


Following these instructions, the robot drives in a small square twice and ends up back where it started. (You can try it out below if you wish.)

The whole program uses only 5 code blocks (3 x purple, 1 green, 1 grey).

Question:

Write a program that gets the robot to the green square using 12 blocks or less.

Slow
Normal
Fast



Run
Erase

Move forward

Turn left

Turn right

repeat 10 times
 do

Programming Instructions:

Drag the code blocks into the workspace on the right.

Clip them together to make your program.

Try out your program by pressing Run.

Shortish Program

Answer:

Here is one solution:



Explanation:

This was arrived at by writing producing a simple but long answer and then pattern spotting to see which combination of loops were possible and then which combination of possible loops required the least number of code blocks.

Maybe you found it difficult to spot patterns in this task. If you can recognise the patterns, you can check and analyse the series of the code blocks one by one from the back. There are many available combinations of instructions (code blocks) because of a lot of paths at the front, but there is only one path from the middle point to the end. This makes spotting patterns easier. (See the pictures below).



Step 1



Step 2



Step 3



Step 4

If all the blocks consist of a minimal pattern repetition in your program, the program has used the least number of instructions.

It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp),

This task is about spotting patterns and using loops to solve a problem. This not only saves the programmer too much typing, but it can also make fixing errors quicker. Without loops available, this task would need 29 code blocks to be used!

Pattern recognition (spotting patterns) in computer science and programming is key to determining appropriate solutions to problems and knowing how to solve certain types of problems. Recognising a pattern, or similar characteristics, helps break down the problem and also build a construct as a path for the solution. Ever find yourself saying, 'Where have I seen this before?' This could be a significant step in computational thinking. The patterns which you recognise can be iterative structures in algorithms and programs.

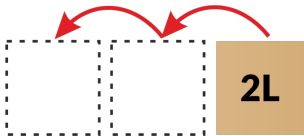
There are 8 boxes on a board. The positions of the boxes are labelled from 1 to 8.

One of three types of movement rule is placed in each box.

An example of each rule type is given below:

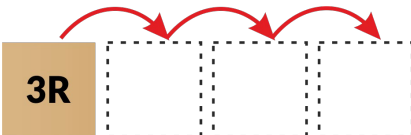
1. Movement to the Left

e.g. 2L means move two boxes to the left:



2. Movement to the Right

e.g. 3R means move three boxes to the right:



3. Do not move

If the rule says "0", do not move from this box at all.

Question:

Consider this board:



Which box should you start in so that, by following the rules, every box is visited?

2 3 5 It is not possible to visit every box

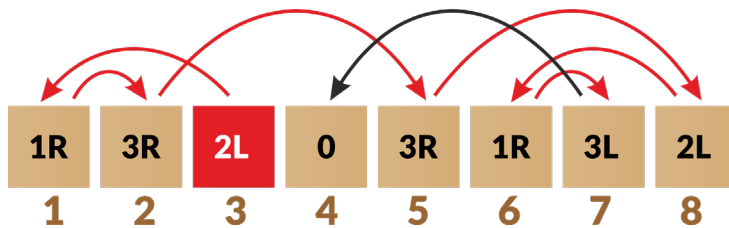
Box Jump

Answer:

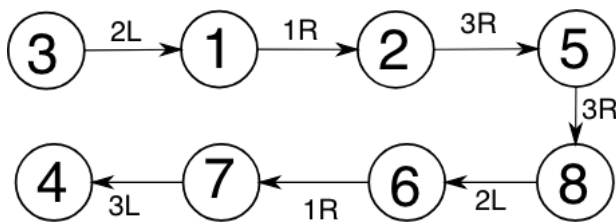
3

Explanation:

Working backwards, we can see that column 4 is reached by column 7, which is reached by column 6, which is reached by column 8, which is reached by column 5, which is reached by column 2, which is reached by column 1, which is reached by column 3:



We can also draw this as a graph, with the label of the nodes being the column, and the label of the edges being "how to move between columns". This graph can be drawn starting at any node, and is complete when all columns have been written down.



It's Computational Thinking:

CT Skills - Decomposition (DE), Algorithms (AL), Evaluation (EV)

Concepts - Specification (Sp), Algorithms (AI), Implementation (Imp), Digital Systems (Ds)

This problem is about following pointers: We can think of a move like "3R" from column 2 to column 5 as following a pointer. Given this collection of pointers, which is really a directed graph, we are looking for the "head" or "parent" node of this collection.

Following a sequence of pointers is important in memory management by the operating system (or Java garbage collection) so that memory which is no longer being used can be recycled and "reclaimed" for other programs to use. Many software errors can be found by tracing back problematic calculations/instructions back to their parent origin.

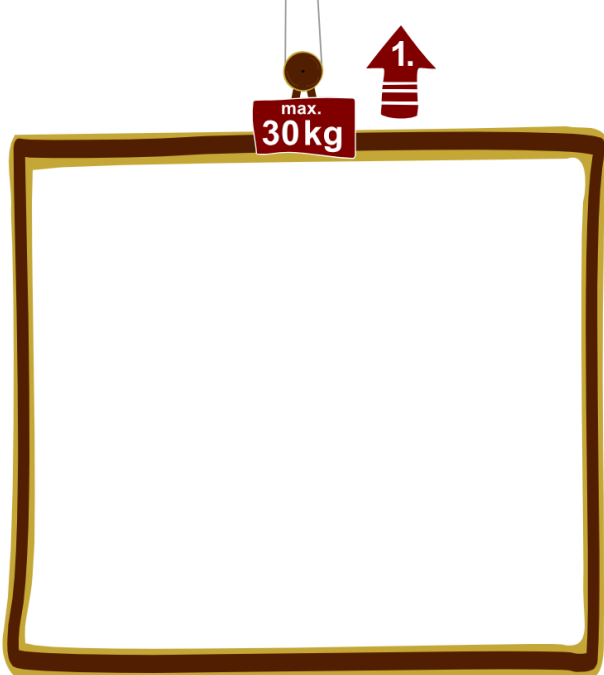
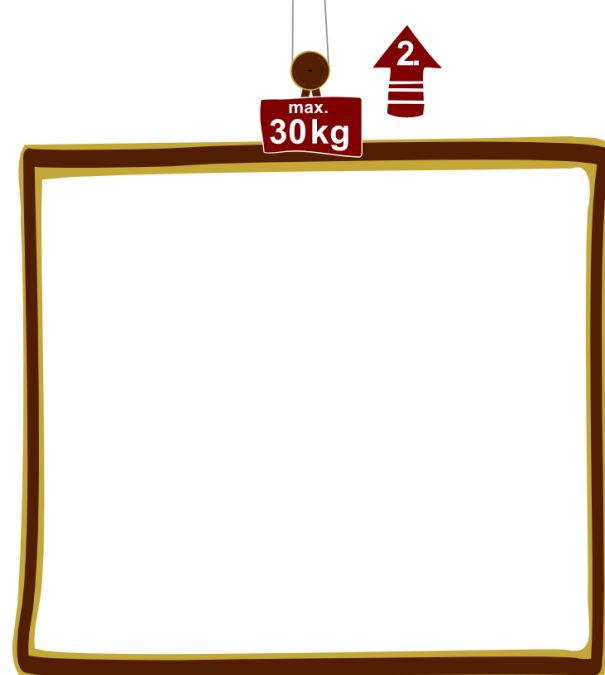


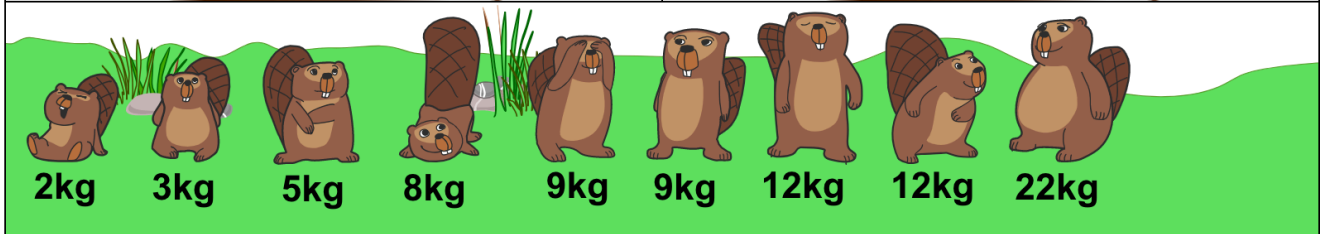
The goto statement (or jump statement in assembly language) can be modelled by this game. A goto statement indicates that instead of executing the "next" instruction, move to some other part of the program and continue execution. In this task, the "program" is a sequence of goto instructions, where the only "terminating" instruction is the one at position 4.

A lot of beavers need to use two lifts.

Each lift can only take a maximum of 30kg.

Question:

Drag the beavers in to the lifts so that as many beavers as possible can take the lifts at once.

Elevator

Answer:

One of the lifts can take the beavers and luggage that weigh: 2kg, 3kg, 5kg, 8kg and 12kg.

The other lift can take the beavers and luggage that weigh: 12kg, 9kg and 9kg.

Explanation:

The first idea that might come to mind could be to pack into the first lift as many beavers as possible with the smallest weights:

$$2 + 3 + 5 + 8 + 9 = 27 \text{ kg}$$

$$\text{and into the second cabin } 9 + 12 = 21 \text{ kg}$$

This is 7 beavers in all but it is possible to pack 8 beavers into the lift as shown above.

If we move the 9 kg beaver in the first lift into the second lift, this uses the total weight allowed in the second lift (30 kg) and now there is room for another 12kg beaver in the first lift.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL)

Concepts - Abstraction (Ab), Data Interpretation (Di), Specification (Sp), Algorithms (Al)

This problem sees us having too many possibilities, and it is impossible to check them all in reasonable time. We have to find the 'best possible' solution to the problem, however, this may not always be the best solution!

In Computer Science there are often problems that are practically unsolvable. However, we can follow a clever plan such as starting by trying to place as many beavers as possible into the first lift. In Computer Science such a plan, that leads to a good solution but possibly not the best solution, is called a heuristic.

Beaver Daniel received a chest of gold that is locked with an electronic lock. The lock can be opened by entering a code of 9 digits.

Daniel has received the following hints about the code:

- The only digits in the code are 2, 6, 7 and 9
- The digit with the highest value is used the lowest number of times in the code.
- The digit with the lowest value is used the highest number of times in the code.
- The code looks the same in reverse.
- All consecutive digits are different.
- The last digit entered is odd.

With the information given above, can you determine the pass code?

Question:

Drag the digits to the passcode area and click on 'Save answer' when you think you have the correct code.

Answer:



7 2 6 2 9 2 6 2 7



Explanation:

Hints 1, 2, 3 tell us that number 9 appears once, number 6 and 7 appears twice and number 2 appears 4 times.

Hint 4 tells us that number 9 must be in the middle.

Hint 6 (combined with hint 4) tells us that 7 must be at the beginning and the end.

Hint 5 tells us that the number 2 must be at places 2, 4, 6, 8.

Now you know where to place the 6's

It's Computational Thinking:

CT Skills - Decomposition (DE), Pattern Recognition (PR), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp), Interactions (In)

While this task originally appears to not give you enough information to work out the code, it turns out that if you take small steps, looking at the hints, you actually have enough information and you don't have to try out all of the combinations.

Computational Thinking is about analysing problems and trying to think of a clever way to find a solution. You could just have a computer try out all possible combinations and then check if the combination follows the clue. But if you do some logical thinking you will see that you can reason your way to a single solution.

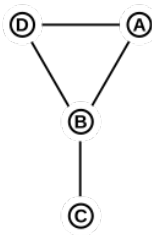
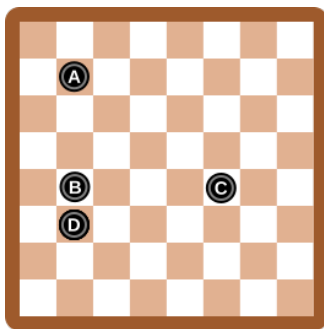
You can see the reasoning you do here as knowledge based reasoning. You have some knowledge and with that knowledge, and your reasoning skills, you are then able to infer new facts and solve other problems.

Below on the left you see a picture of a game board with 4 pieces placed on it.

The diagram to the right of the board represents the positions of the pieces.

It is drawn in the following way:

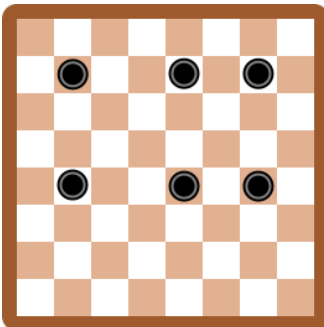
- For each piece on the board, draw a circle.
 - If two pieces are in the same row on the board or in the same column on the board, then draw a line between their circles in the diagram.
- (Do not draw any other lines in the diagram.)



Letters have been placed on the pieces and the circles so you can easily check that the diagram is correct.

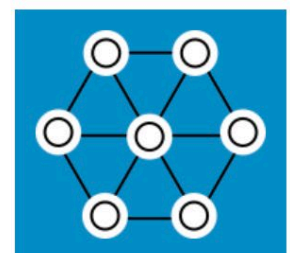
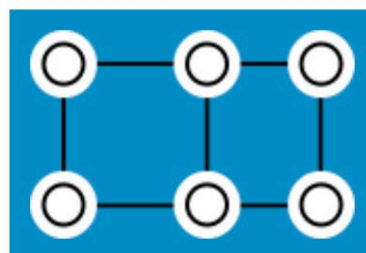
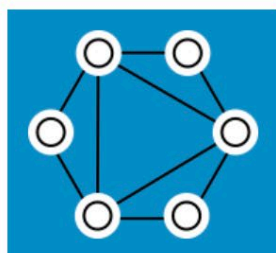
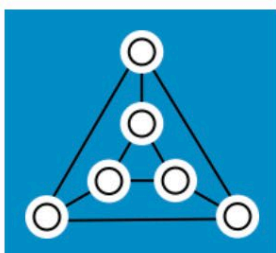
A new board with six pieces is shown below.

A new position diagram for this board is drawn in the same way.



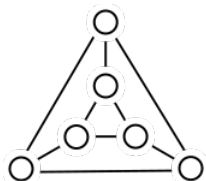
Question:

Which of the four diagrams below were drawn?



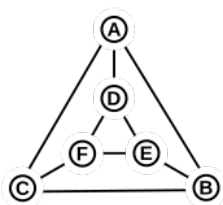
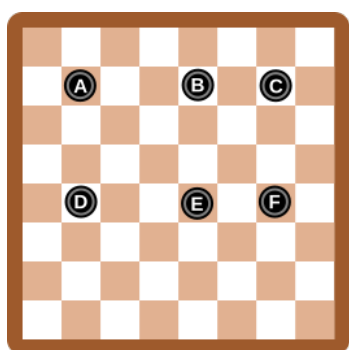
Rows and Columns

Answer:



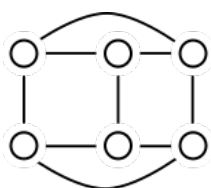
Explanation:

You can easily verify this with the picture below where we have put letters on the pieces and on the circles in the diagram.



The easiest way to see that the other three diagrams are not correct, is to note the following: On the board each piece has two other pieces in the same row and one other piece in the same column. Hence, in the diagram each circle must be connected to $2+1=3$ other circles. (Also note that answer C has 7 circles, which is one too many.)

Maybe you thought that diagram B was correct? After all, it looks very much like the pieces on the board. However, the leftmost and rightmost circles in the diagram are connected to only two other circles, so the diagram cannot be correct. To make it correct, you could add two lines, as shown in the following figure:



Rows and

It's Computational Thinking:

CT Skills - *Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)*

Concepts - *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp), Interactions (In)*

In computer science diagrams like this are often used to represent the essential information of a problem. Such a diagram is called a graph. The circles in the diagram are called vertices or nodes of the graph.

In a graph it is only important whether two vertices are connected or not. Where the vertices are drawn is immaterial. The same graph can often be drawn in different ways, as is illustrated in the explanation section: Both answer A and the last picture in the explanation are correct diagrams for the board and represent the same graph.

Whether a graph can be a useful representation of a computer science problem depends on what information you need to solve the problem. If, for instance, you need to know whether a white piece on the board is in the same row or column as a black piece, then our graphs are not a good representations: The colour of the pieces is not represented in the nodes. If you need to know whether there are sufficient pieces left on the board so that a game can still be won, then a graph is 'overkill': You only need to keep track of how many pieces are left on the board at each moment, and there is no need to record which piece is in the same row or column as another piece. On the other hand, the graph is a good representation for answering questions such as: 'What is the minimal number of pieces you have to remove so that no piece is in the same column or row as any other piece?'

Finding the correct representation for a problem is one of the challenges a programmer or computer scientist encounters regularly in their job.



Toll Roads

Year 3+4:

Year 9+10: B

Year 5+6:

Year 11+12: A

Year 7+8: C

Bob decided to drive from Hamper to Mug.

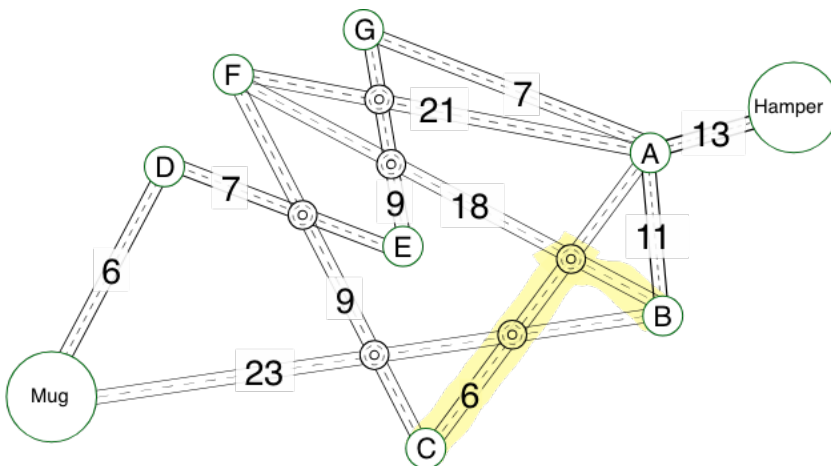
In the map below, circles with letters are cities and lines are two-way roads.

Roads also have roundabouts where they intersect.

The number beside a road is the toll that cars must pay every time they enter the road.

Cars can change their route at roundabouts but they need to pay the full toll for the road they enter.



For example, to drive from city B to city C you can take road 18 and road 6 thus the toll fee is 24.



Question:

What is the minimum toll fee Bob should pay to drive from Hamper to Mug?

A row of cards is laid out in front of you.

The cards may lie face up:  face down: 

How to play:

For each step in the game you:

- examine the cards from right to left
- if the current card is face down, you turn it face up and stop
- if the current card is face up, you turn it face down and move to the next card
- when you run out of cards, you stop.

The images below show the effect of such a step:

Before: 

After: 

Here are 7 cards lying face down:

(You can flip them over by clicking on them.)



Reset

Question:

How many steps will it take, using the system above, to make all 7 cards face up?

- It will take 10 steps or less.
- It will take more than 10 steps but at most 100.
- It will take more than 100 steps but at most 1000.
- It will take more than 1000 steps.
- You can never reach the situation with all 7 cards facing up.

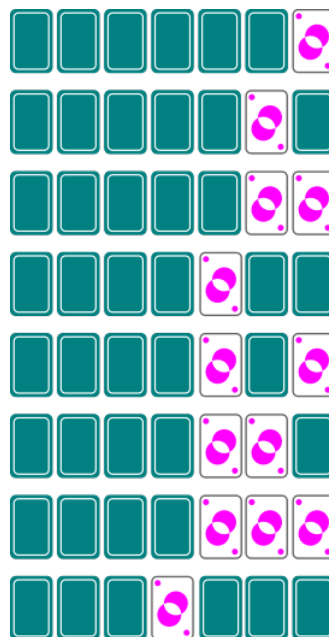
Flipping Cards

Answer:

It will take more than 100 steps but at most 1000.

Explanation:

There are various way to see (or guess) that this is the correct answer. If you perform the first few steps you will see the following patterns:



Notice that the rightmost card is turned over at the 1st step, the second card from the right is first turned over at the 2nd step, the third card from the right is first turned over at the 4th step and the fourth card from the right is first turned over at the 8th step.

Is there anything remarkable about the sequence 1, 2, 4, 8, ... ?

Indeed, every number in the sequence is double the number that precedes it. So you can guess that it will take 16 steps before the 5th card from the right is turned over, 32 steps for the 6th card and 64 steps for the 7th card. (And the 7th card from the right is the first from the left.) So we need *at least* 64 steps.

It is a little bit more difficult to see that you in fact need **127** steps to turn all cards facing up. For that you need to realise that after step 7, just one step before the 4th card is turned over, the pattern contains 3 consecutive cards facing up. Similarly, after step 15, one step before the 5th card is first turned over, the pattern contains 4 consecutive cards facing up. So, 7 cards facing up happens one step before the 8th card would be turned over (if there were 8 cards), which is after step $2 \times 64 - 1 = 127$.

In the *It's Computational Thinking* section you can read about a different explanation that involves *binary numbers*...

Flipping Cards

It's Computational Thinking:

CT Skills - *Decomposition (DE), Pattern Recognition (PR), Abstraction (AB), Algorithms (AL)*

Concepts - *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Algorithms (Al)*

Inside a computer, numbers are represented in so-called *binary notation* where instead of the digits 0-9, a number is 'written' using only zeroes and ones (called *bits*).

The representations of the first few numbers are as follows:

1 is represented as 0000001

2 as 0000010

3 as 0000011

4 as 0000100

5 as 0000101

6 as 0000110

etc.

(We use 7 bits for a number here, but on modern computers the total number of bits used is usually 32 or 64.)

Do you recognize these patterns? Indeed, if you use 0 for a card that is placed face down, and 1 for a card that is lying face up, than you obtain the same patterns as for the first 6 steps of our game. And if you know that 1111111 represents the number 127, then you see that indeed 127 steps are needed to end up with the requested pattern.

The 'step' which we use in our game is what is used by the actual electronics inside a computer to increase a binary number by 1.

Your job is to colour in some of the circles in the picture below.

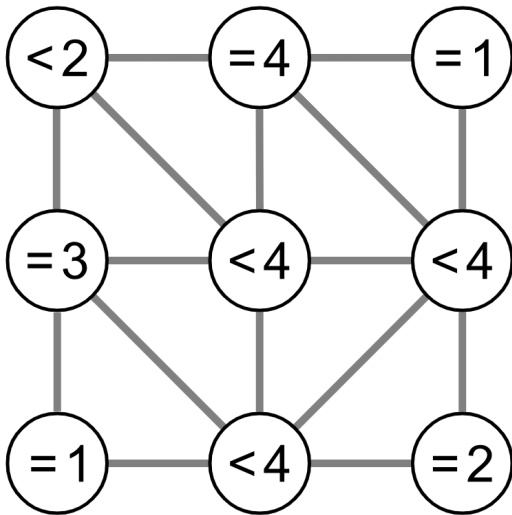
The circles have connections to some of their neighbours.

The numbers inside each circle indicates the number of neighbours which need to be coloured in. For example, the circle marked with " $=3$ " must have exactly 3 of its 4 neighbours coloured in.

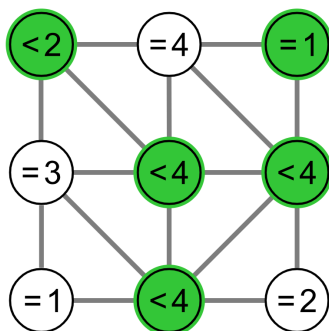
Similarly, the circles marked with " <4 " must have less than 4 of their neighbours coloured in.

Question:

Colour in the required circles by clicking on them:



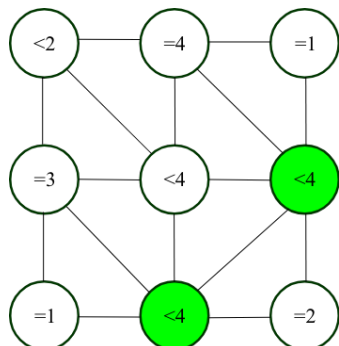
Answer:



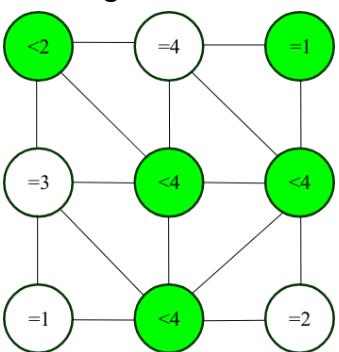
Connections

Explanation:

Looking at the bottom-right most circle, which contains "=2", we know that both of its neighbours must be filled in to obtain:



Noticing the circle labelled "=4", all four of its neighbours must be filled in, yielding:



At this point, all circles are satisfied. Examining each of the remaining circles, we see that they cannot be filled in. Specifically

if the "=1" circle was filled in, then the "=3" circle would be incorrect

if the "=2" circle was filled in, the "<4" circle above it would be incorrect

if the "=3" circle was filled in, the "<2" circle above it would be incorrect

if the "=4" circle was filled in, the "=1" circle to its right would be incorrect

Note that we can also start at the "=4" circle and arrive at the same solution after similar reasoning.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Algorithms (AL),

Concepts - Abstraction (Ab), Data Interpretation (Di), Algorithms (Al)

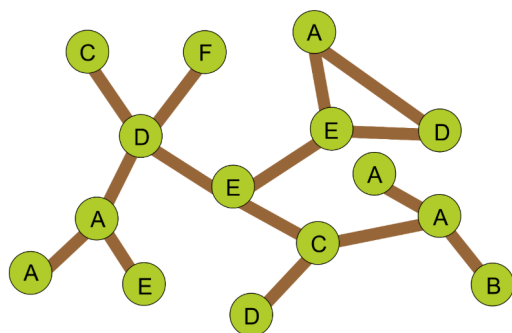
This problem requires logic and demonstrates that brute-force solutions are not effective. By brute-force, notice that for each of the 9 circles, they are either filled in or not: therefore, there are 2 choices for each circle. The total number of choices for filling in these circles is $2^9=512$. So, one solution is to try all 512 different ways of filling in these circles. However, using reasoning, and in particular, a sequence of reasoned deductions as shown in the solution, it is possible to substantially reduce the number of possibilities that need considering to something much more manageable.



Year 9+10: A

Year 11+12

This is the map of a park:



The green circles with letters represent the trees and the brown lines are paths. Note that some letters are used to label more than one tree.

Walking from tree F to tree B can be described as **F D E C A B**.

Last Sunday two families walked in the park.

The Wilde family's walk was **B A A A C E D E E D A**.

The Gilde family's walk was **F D C D A E A D E D A**.

Both families started their walks at the same time.

Walking from one tree to another tree, down one path takes the same amount of time.

Question:

How many times did the two families meet at a tree?

Once Twice Three times They never met at any of the trees

Park Walk

Answer:

They never met at any trees.

Explanation:

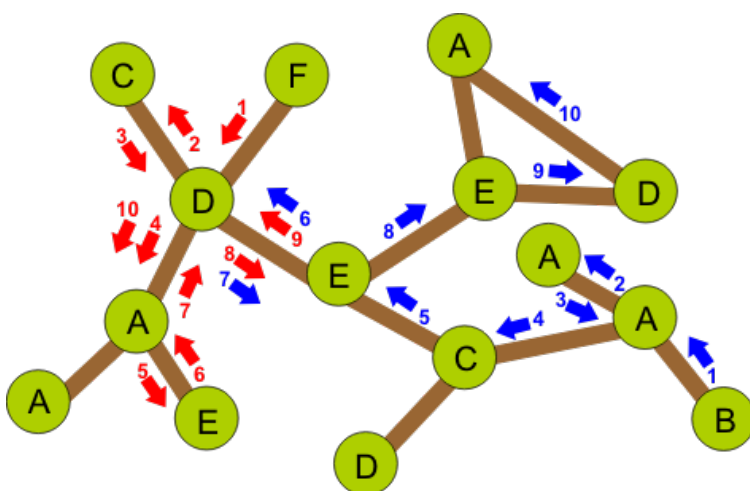
It is not enough to find the same letter at the same position (i.e. same time) during the walk, because the same letter can denote a different tree of the same kind. For example, both families ended their walk at a tree labeled A, but if we follow both walks step by step then we can find out that actually they ended at different trees.

Note that we can follow each step of each family quite easily because the neighbours of each tree (the trees that are connected to it by a path) are always labeled with different letters. However, to follow the two walks in parallel is not that easy (but doable).

So let's mark out first the walk of the Wilde family (starting at F, by blue colour) and number the visited trees by the time in which they visited it (blue numbers 1 to 11). Then let's mark out, in a similar way, the Gilde family walk with a red colour and red numbers.

The two families would meet at some trees only if that tree were numbered by the same red and blue number. But we cannot see any such trees in our picture.

Note that there are only two trees visited by both families, so we can focus only on those two. The tree D (the leftmost one in the park) was visited by the Wilde family only at time 7 and by the Gilde family at times 2, 4, 8 and 10. The tree E (the one neighbouring the leftmost D) was visited by Wilde family at time 6 and 8, but by the Gilde family only at time 9.



Park Walk

It's Computational Thinking:

CT Skills - *Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)*

Concepts – *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Interactions (In)*

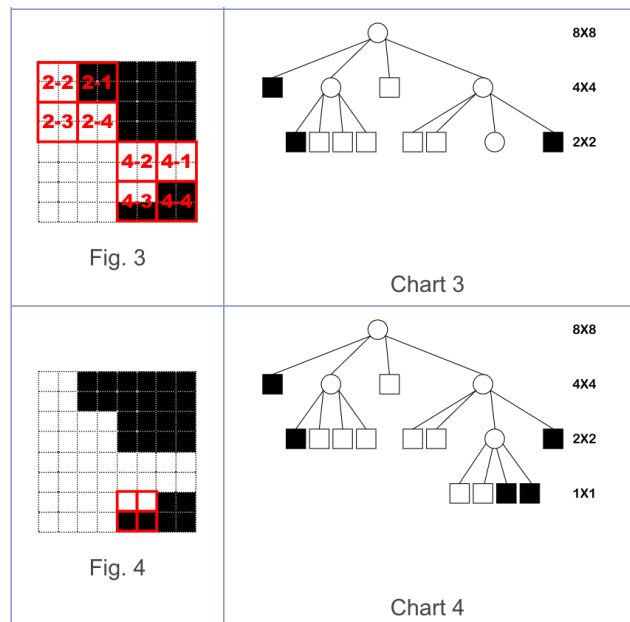
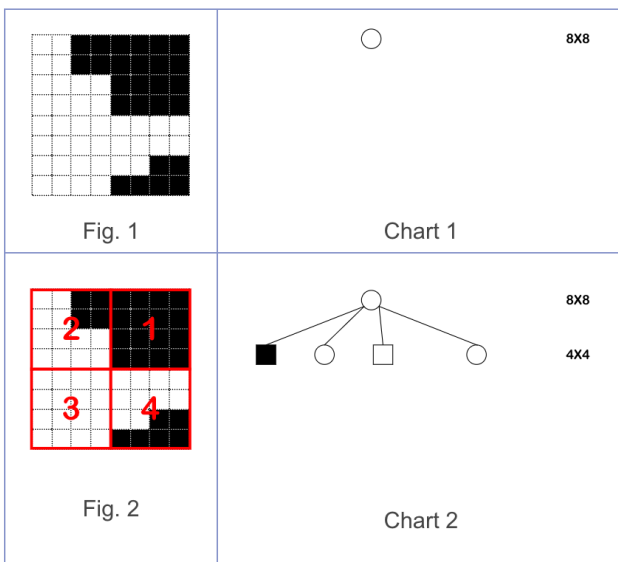
Computer scientists and programmers often use graphs (and then they speak about vertices and edges instead of trees and paths connecting them). The Graph theory is an important and interesting part of both mathematics and computer science. Both these sciences use it in their own way for slightly different purposes.

Another interesting point about this task is the representation of the walks in the park. Despite the fact that some of the trees (vertices) are marked by the same letter, the walks that start from B or F can be unambiguously described by the sequence of letters along the walk. It means that one sequence of letters describes only one walk. It is because the neighbours of each tree (neighbours of tree X are the trees that are directly connected to X by a path) are always labeled with different letters. So if we know where we are at some moment of the walk and we see the next letter in the walk's representation then there is no doubt which tree we should visit next.

This labelling system may seem odd at first but when we look at road maps we find that, in many countries, there are towns with the same name!

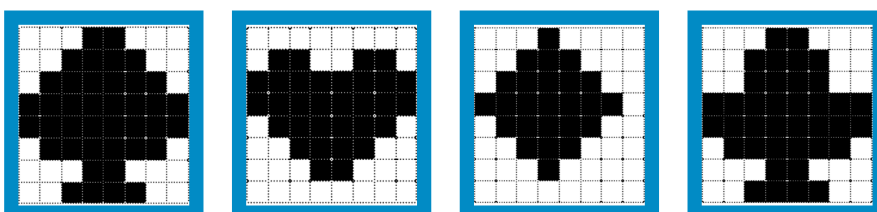
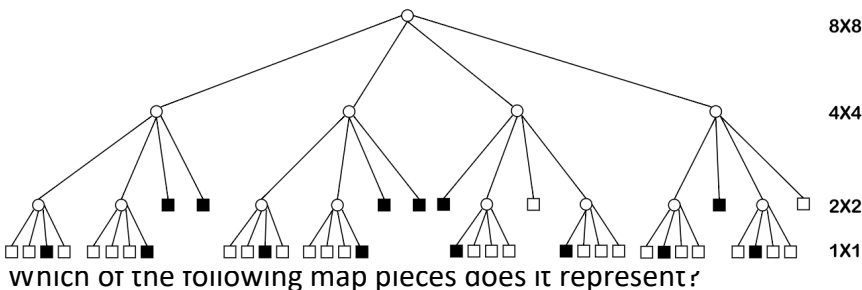
Pirate Beaver has a very large treasure map which is cut into smaller pieces. Each map piece shows a region of 8 units wide and 8 units long (Fig. 1). However, Pirate beaver has a very small boat and cannot take all the map pieces with him. Smart as she is, Pirate beaver finds a way to document each region (map piece) into a small chart in her note book. Here is how:

1. If all units in the region are in the same colour, she marks a “square” on her notebook with the same colour.
2. Or else, she marks a “circle” (shown as Chart 1) and then divides the region into 4 subregions (shown as Fig. 2) according to its central point.
3. Repeat step 1 and 2 until all units are recorded (shown as Chart 4).



Question:

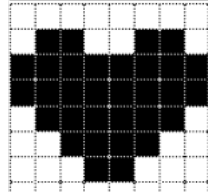
Here is another chart in Pirate Beaver's notebook:



Treasure Maps

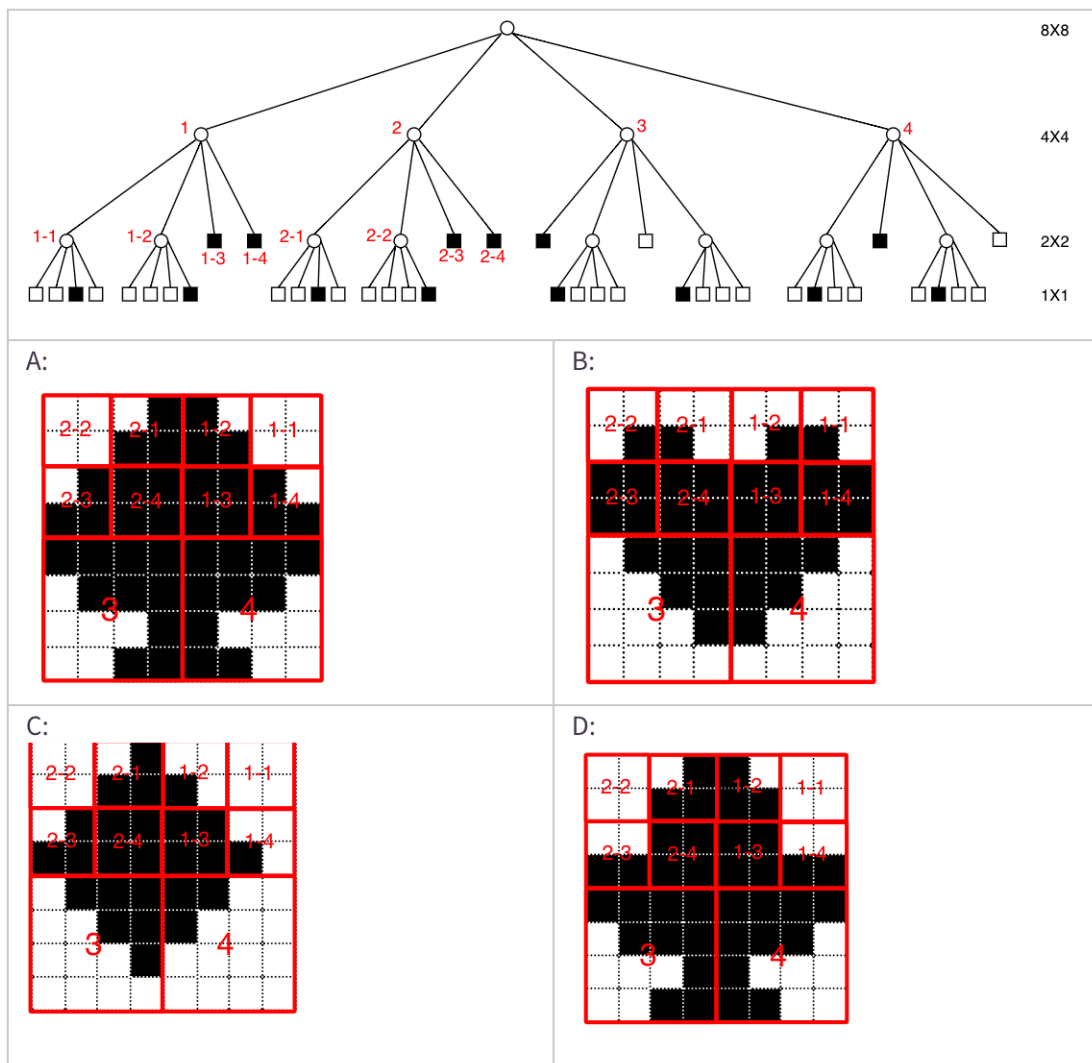
Answer:

The correct answer is:



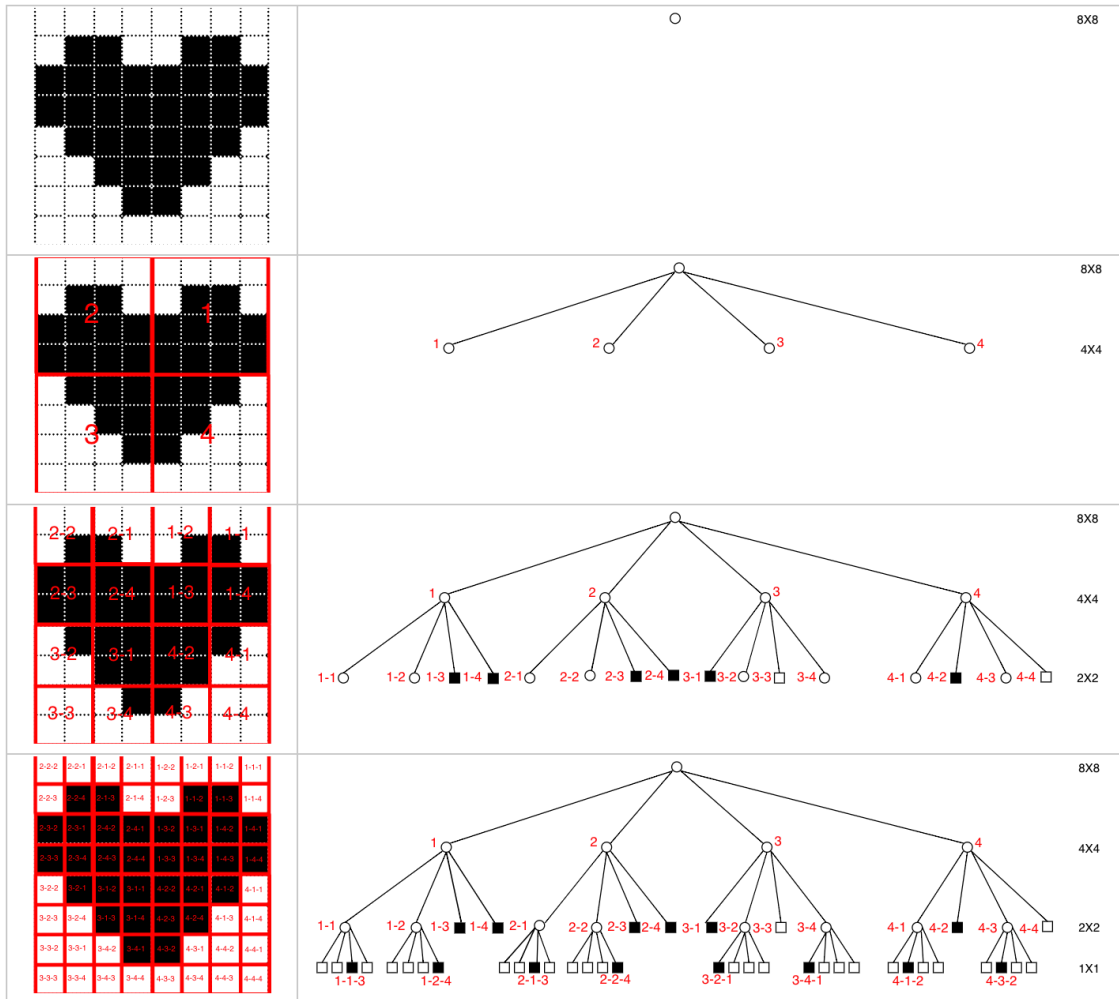
Explanation:

In the chart, we can mark the points that represent 4X4 regions with number 1 to 4, and mark sub-points of point 1 with number 1-1 to 1-4, and so on and so forth. The corresponding positions of regions in the map and points in the chart are shown as below. 1-3, 1-4, 2-3 and 2-4 are marked in black in the chart, which means that the colour of these four 2X2 regions in the map is black. Only (B) satisfies this criteria.



Treasure Maps

Explanation continued:



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Digital Systems (Ds), Interactions (In)

The 4-region chart is called "Quadtree" in Computer Science. A quadtree is a tree data structure in which each internal node has exactly four children. All forms of quadtrees share some common features:

- They decompose space into adaptable cells.
- Each cell (or bucket) has a maximum capacity. When maximum capacity is reached, the bucket splits.
- The tree directory follows the spatial decomposition of the quadtree.

There are some common uses of quadtrees, such as image representation (this example), image processing, mesh generation, ...etc..

A diagnostic device in a medical lab must repeatedly shake specimens taken from patients.

The device works according to a computer program, which is written on numbered lines. The device reads the program line by line. It always reads one line and then executes it immediately. If the line contains the command go to X, the device jumps to the line X and continues reading and executing.

The Program:

```
1. set A to 0
2. add 1 to A
3. go to 6
4. if A equals 60 go to 8
5. set A to 0
6. add 1 to A
7. go to 2
8. shake the specimens A times
9 end
```

The program is able to store a number A, to add 1 to the number stored in A, and compare its value with another number.

Question:

How many times will the device shake the specimens when this program is run?

The specimens will never be shaken.

The specimens will be shaken once.

The specimens will be shaken 60 times.

The program will not stop the specimens being shaken.

Answer:

The specimens will never be shaken.

Explanation:

The program always jumps from line 3 to 6 and from line 7 to 2. Except at the beginning, the program visits only lines No. 2, 3, 6, 7. The instruction to shake the specimen is on line No. 8, which is never visited. This means the device will never shake anything according to the program. Moreover, the instruction on line No. 9 is never executed, so the program continues forever.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Algorithms (AL)

Concepts - Abstraction (Ab), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp), Digital Systems (Ds)

The first programming languages, developed in the 1940's and 1950's, looked like the one in our task, and are called assembly languages. Program lines were numbered and special commands, known as go to instructions, were used for jumping to a different line than the next one. It was really difficult to read these programs and find mistakes but easy to make them. The error-proneness of such programming languages was the reason why modern programming languages were developed starting in the 1950's. These modern languages are not line-oriented and contain structures like loops, procedures and selections, instead of go to instructions.



Longest Word Chain

Year 3+4:

Year 5+6:

Year 7+8:

Year 9+10

Year 11+12: C

Beavers often play a word chain game.

Game Rules:

Place 9 word cards face up in front of both players.

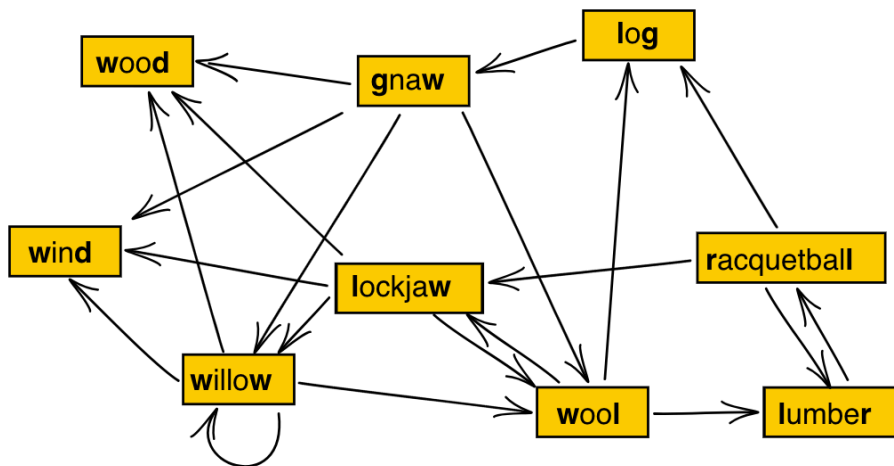
Player one starts by saying one of the words.

Player two says another word which begins with the last letter of the previous word.

Player one says another word which begins with the last letter of the previous word and which has not been used yet.

Play continues in this way until there is no new word available.

Below is a set of 9 cards. Arrows have been added to help with this question:



Question:

What is the largest possible number of words that can be said in one game?

Longest Word Chain

Answer:

The beavers can use at most 8 words in one game.

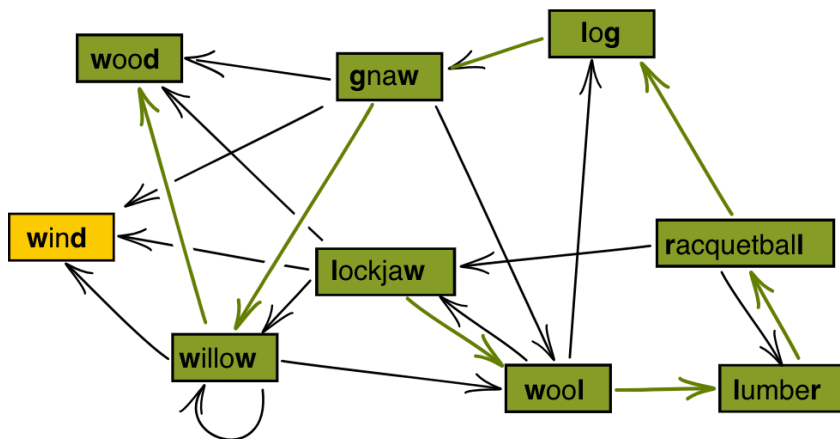
Explanation:

One example is:

lockjaw-wool-lumber-racquetball-log-gnaw-willow-wood

Can you find another word chain of the same length?

It is good to have an example. It means we can be sure that one game can use at least 8 words. But we do not know yet whether it is possible to use all 9! However, that would mean using all the words. Now consider the words *wood* and *wind*. There is no word beginning with *d*, so if any of these words will be used, it must become the last word of the game. But there cannot be *two* last words. Hence we cannot use all 9 words in one game.



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp)

This task requires you to understand a set of rules (how to play word chain), a clear and practical representation of the data (the graph), and then to find an optimal solution in the given system. All of these are typical tasks computer scientists enjoy.

With some extra knowledge of computer science, you may recognise the given problem as searching for the longest path in a directed graph. This is closely related to the well-known Traveling salesman problem, but also a well-known problem on its own. Computer science tells us that finding the longest game for bigger vocabularies (mind that you know thousands of words!) is not feasible, it takes too long with even the best algorithms we have to date. Maybe you can come up with a better one?

Some friends are planning a party.

The diagram below shows some of the friendships in the neighbourhood – two people are friends with each other if a line connects their names.

For each pair of friends, only one of them is to buy and bring a gift for the other.

The numbers in the diagram show how many gifts each person can buy.

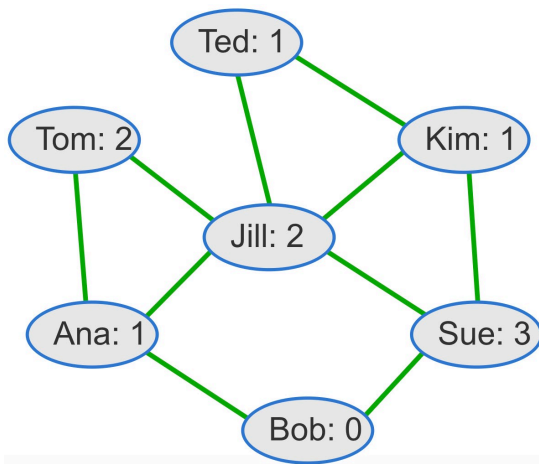
No friend is supposed to buy more gifts than this number.

Question:

Click on the lines in the diagram to show who gives and receives presents.

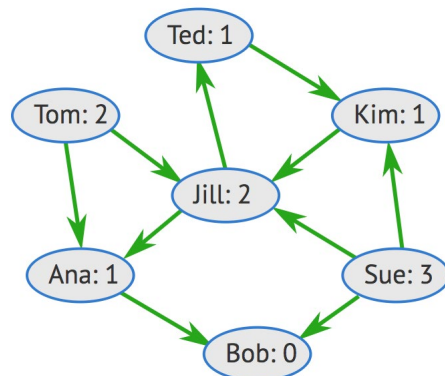
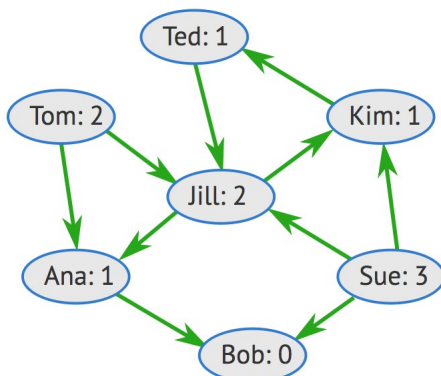
Click again to change the arrow direction. Click once more to remove the arrow.

(Click Save when you are satisfied with your answer.)



Answer:

There are two possible answers that obey all the given rules:



Gifts

Explanation:

In order to choose the directions of the arrows correctly, we start with Bob. This is because he has no money. Bob will receive gifts from his friends Ana and Sue. Thus, Ana has already spent all her money, so she will receive gifts from her friends Jill and Tom. For these friends, there is no choice.

Now we seem to have many options. But there is a critical choice: Friends Ted and Kim can only buy 1 gift. For this pair, we need to decide who will buy the gift for the other.

- If we choose Kim to buy a gift for Ted, then Kim needs to receive a gift from his friend Jill. Then Jill has spent her money and will receive gifts from Ted, Tom, and Sue. Sue also needs to buy a gift for Kim. This solution is shown on the left.
- If we choose Ted to buy a gift for Kim, Ted needs to receive a gift from Jill. Again, Jill cannot buy any more gifts and will receive gifts from Kim, Tom, and Sue. Again, Sue also needs to buy a gift for Kim. This solution is shown on the right.

So each choice for Ted and Kim leaves no further choices for the remaining friends. As there are only two choices for Ted and Kim, there are only two correct answers.

It's Computational Thinking:

CT Skills - *Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)*

Concepts - *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Digital Systems (Ds), Interactions (In)*

The friends and the friendships between some of them together form a network of friendships with nodes (people) and links (friendships). This reminds us of the well-known social networks many people are using. But there is an important distinction among these networks: In some of them, there are mutual "friendships" (i.e., links without direction), like in this Bebras task. In other networks, there are "followers", such that links have a direction: You may follow some celebrity, but the celebrity may not follow you in return.

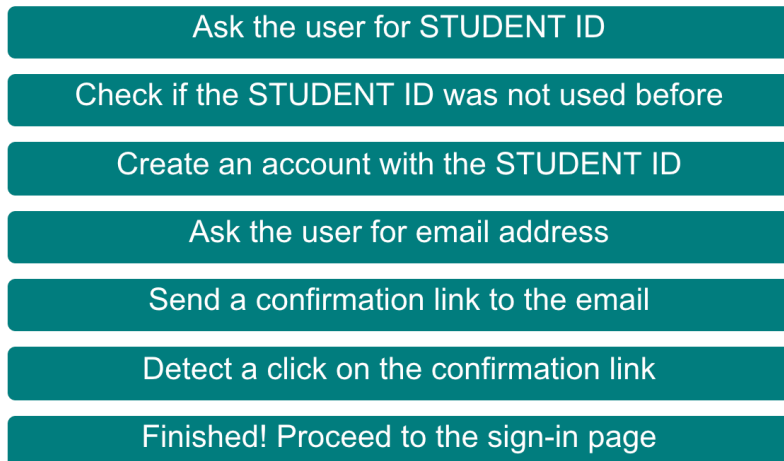
In this Bebras task, the bidirectional friendship links have to be made directed "giving" links. This is again different from following, as there are limits for giving but not for following. We want to see as many gifts going from one friend to another (hopefully one gift in each friendship), but without exceeding any friend's limit. This is similar to a well-known problem in Computer Science: In a network, the capacities of links are limited, but within these limits we want to have as much flow through the links as possible. The difference is that in this task, nodes have capacities, while in "network flow problems" links have capacities. However, it is not difficult to change this task's problem into a real network flow problem. Then, it is possible to solve this task, even if there are many friends and friendships; efficient algorithms for network flow problems are well-known in Computer Science.



Sara is developing a website for her friends to share their art projects.

She designed a sign-up process so a user can first create an account with a unique STUDENT ID as a username and then enter a valid email address for further communication.

The following diagram shows the steps of her design:



During testing she found a critical design bug:

If a user mistypes their email address, they do not receive the email with a confirmation link and consequently they can not sign-in. On the other hand, they can not start the process all over again as their STUDENT ID has already been submitted and it is not available any longer.

Question:

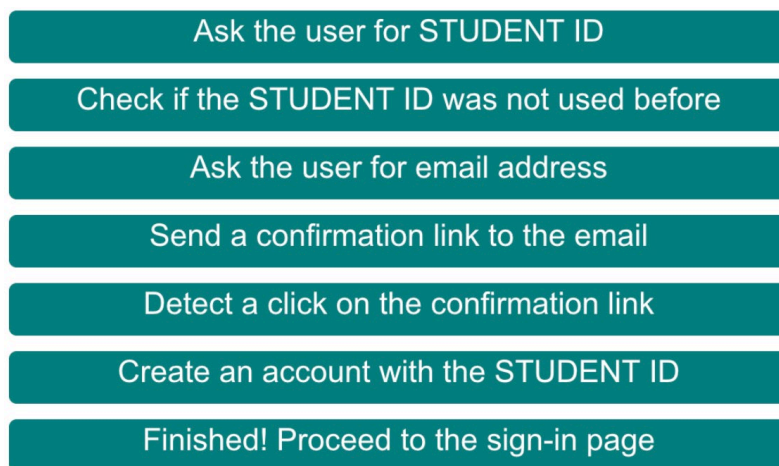
Help Sara by re-arranging the sign-up design steps to remove the bug.

(You can re-arrange the program by dragging and dropping the instructions in the sequence.)

Signup Debugging

Answer and Explanation:

The correct approach is to create the account only after the confirmation is received (detecting a click on the confirmation link sent to the given email). So the 3rd step should move to the 6th position (just before finish) as shown below:



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Digital Systems (Ds), Interactions (In), Impacts (Im)

The motivation of this task introduces system design which is the process of defining and developing information systems to satisfy specific requirements of the users. One of the means of system design is control logic. It responds to commands from the user and it also acts on its own to perform automated tasks that have been structured into the program, thus controlling the operations of the program.

Control flow analysis helps to debug the designed system, i.e. find and resolve defects or problems in a computer program that prevent its correct operation.

In more detail, the process analysed in the task is essentially about two phase locking. The lock phase happens, as initially designed, when the STUDENT ID is chosen, which also locks the particular STUDENT ID. The second phase, the lock release, happens when the confirmation is received by the system. Unfortunately, according to the initial design, this can never happen. The updated design avoids this problem, as the lock phase does not happen before the e-mail is confirmed (simultaneously with a click detection).

Note however, that in a parallel environment this solution will still not work as two students might request the same ID simultaneously causing a race condition to happen. In that case, to alleviate the problem the whole process is split into two sub-processes: (i) Establishment of a communication channel (e-mail, phone, SMS, ...); and (ii) setting up the username (i.e. STUDENT ID).

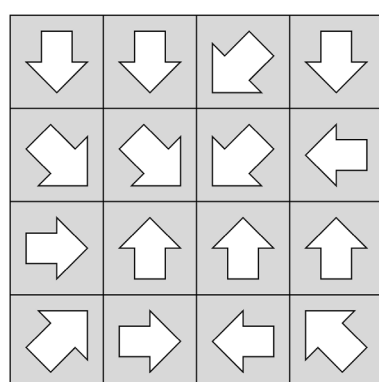
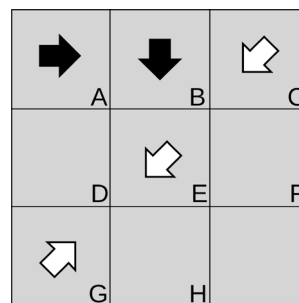
In the figure on the right, the black arrow at A points at one black arrow (at B) and one white arrow (at C).

The white arrow at C points at exactly two white arrows (at E and at G).

In the figure below, you can change arrows from black to white or white to black by clicking on them.

Question:

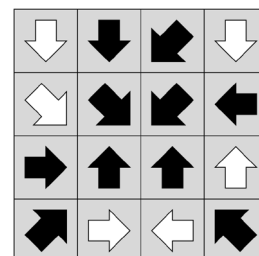
Create a set of arrows so that all white arrows point at exactly one other white arrow and all black arrows point at exactly two other black arrows.



Answer and Explanation:

To find the solution you could start the search, go step by step, and, if you find you are in a dead end, step back and try another possible step.

A possible solution is shown on the right:



It's Computational Thinking:

CT Skills - Decomposition (DE), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)

Concepts - Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp), Interactions (In), Impacts (Ims)

To find the solution you could start the search, go step by step, and, if you find you are in a dead end, step back and try another possible step.

In computer science this is called a backtrack. Backtracking can be used to solve puzzles or problems such as the eight-queens-puzzle, sudoku. In Computer Science it can also be used to solve combinatorial optimisation problems such as parsing and the knapsack problem. Some logic programming languages such as Icon, Planner and Prolog use backtracking internally to generate answers.

Beaver has to tile a big ballroom floor that is made up of 16 rows of tiles and 31 columns.

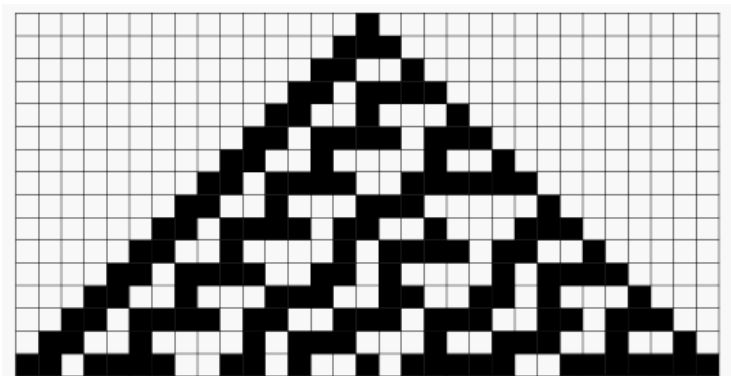
She decides to program the pattern, based on a set of rules. Here is an example "rule set":



When laying a new tile, Beaver looks at the three tiles directly above the new tile and finds which rule in the rule set it matches. This then shows her whether the new tile should be black or white.

To allow her rules to work at the edges, Beaver decides to imagine that all the tiles that are outside of the floor are also white and remain white.

Beaver decides to start with one black tile in the middle on the top row. Using the example rule set, the following pattern is formed:



She wants the last row of the floor to start and end with a black tile.

It must also alternate between black and white tiles like this:

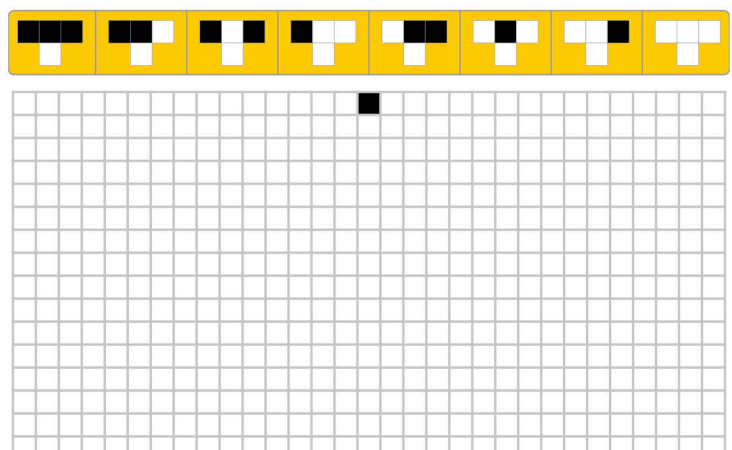


Question:

Create a new set of rules that will produce the required result.

Do this by clicking on the squares in the rule set to turn them either black or white.

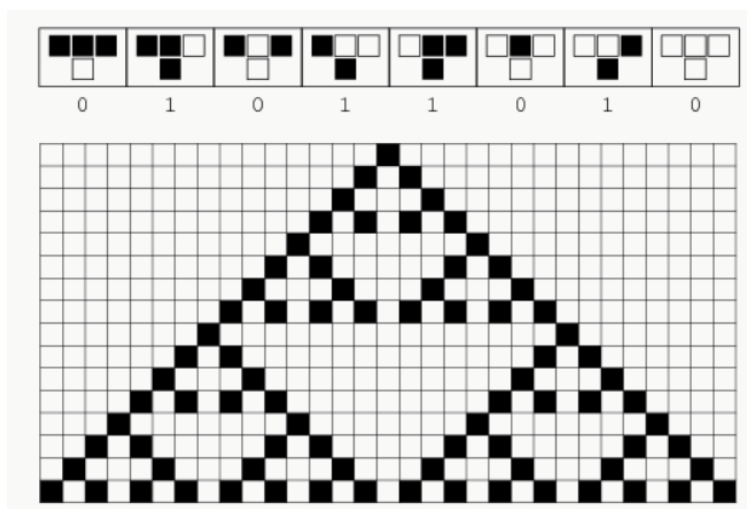
Remember, you have to start with one black tile in the middle of the top row.



Ballroom Floor

Answer:

The following image shows a solution. This is a solution based on the 'XOR' function. In this case, the value of a tile is not dependent on its own state; it is only dependent on the state of both the neighbours. If exactly one neighbour is black the cell will be black.



Fortunately, there are multiple correct solutions.

Explanation:

An interesting feature of this task is that there are at most 256 different rule sets. You can see that you have to indicate what happens to the colour of a cell based on the three cells above it (directly above and the two diagonal ones). There are eight different ways the three cells are coloured black or white (www, wwb, wbw, wbb, bww, bwb, bbw, bbb). For each of these eight situations you have to decide on the colour of the cell below. This means you only have to make 8 choices of black and white. For 8 binary choices you have 2^8 possibilities, so 256.

In the solution diagram we have added '1' and '0' below the choices.

The table on the right shows all valid solutions.

Some of these are easy to find (put a black square somewhere only if the square above it is white and either one or two of the diagonals are black), and some are very interesting to find.

Having so many valid solutions means you can just play with this task to see the patterns emerge without having to spend too much time searching for the solution.

00010010
00011010
00110010
00111010
01010010
01011010
01110010
01111010
10010010
10011010
10110010
10110011
10111010
11010010
11011010
11110010
11111010

Ballroom Floor

It's Computational Thinking:

CT Skills - *Decomposition (DE), Pattern Recognition (PR), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL), Evaluation (EV)*

Concepts - *Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp), Digital Systems (Ds), Impacts (Ims)*

This game is actually related to Conway's Game of life.

Conway's Game of Life was invented by John Conway. It was based on the work of Johan von Neumann, who was trying to find a hypothetical machine that could create copies of itself. Conway's Game of Life is a cellular automaton, which is basically a system where some very simple rules are applied to cells on a grid. With these simple rules, many very complex machines can be built. In fact, it has been proven that any calculation can be done using Conway's Game of Life.

This 1 dimensional version is just another version of the Game of Life. In this version, the student has to come up with his own rules that he can simulate as well.

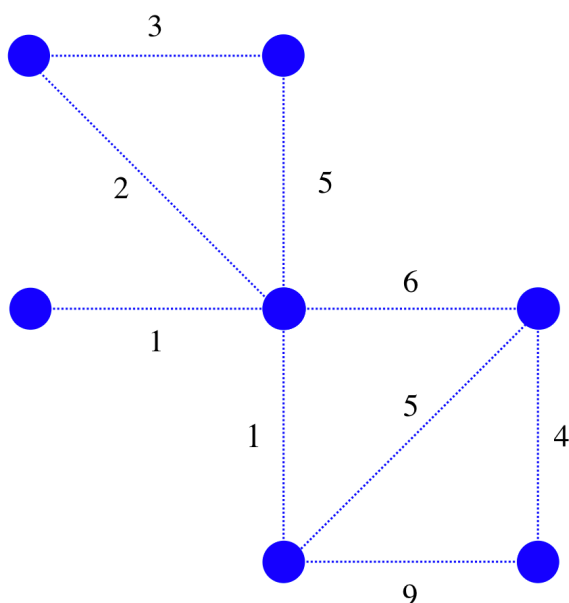
An internet service provider (ISP) wants to set up a new network.

There are seven cities which have to be connected so that every city can send and receive messages from any other city.

The company has to pay to setup links between cities. The costs are shown on the lines linking the cities below.

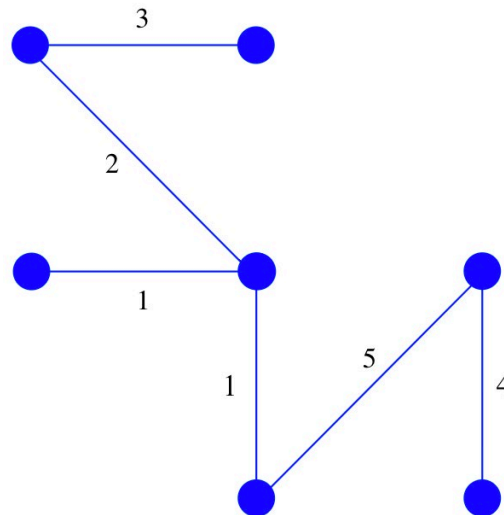
Question:

Select the links that should be built to connect the cities with the least cost.



Optical Fibre

Answer:



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp), Digital Systems (Ds), Impacts (Ims)

The key problem of this task is finding a way to connect all the cities with the least cost. All the possible links between the cities is a graph, where each city is a vertex and each link is an edge. We have to find a subset of edges, which form a tree, so that every city has a path to any other city. We also need to find a tree which has the least cost, which is called a minimum spanning tree.

In computer science, Prim's algorithm is a greedy algorithm that finds a minimum spanning tree for a weighted undirected graph. This means it finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized. The algorithm operates by building this tree one vertex at a time, from an arbitrary starting vertex, at each step adding the cheapest possible connection from the tree to another vertex.

The algorithm was developed in 1930 by Czech mathematician Vojtěch Jarník and later rediscovered and republished by computer scientists Robert C. Prim in 1957 and Edsger W. Dijkstra in 1959. Therefore, it is also sometimes called the DJP algorithm, Jarník's algorithm, the Prim–Jarník algorithm, or the Prim–Dijkstra algorithm.

The algorithm may informally be described as performing the following steps:

- Initialize a tree with a single vertex, chosen arbitrarily from the graph.
- Grow the tree by one edge: of the edges that connect the tree to vertices not yet in the tree, find the minimum-weight edge, and transfer it to the tree.
- Repeat step 2 (until all vertices are in the tree).

Buried Treasure

Year 3+4:

Year 9+10

Year 5+6:

Year 11+12: C

Year 7+8:

There are 5 Forest Imps living in a wood.

Each imp wants to bury its treasure somewhere in the wood.

Forest Imps only bury their treasure under trees.

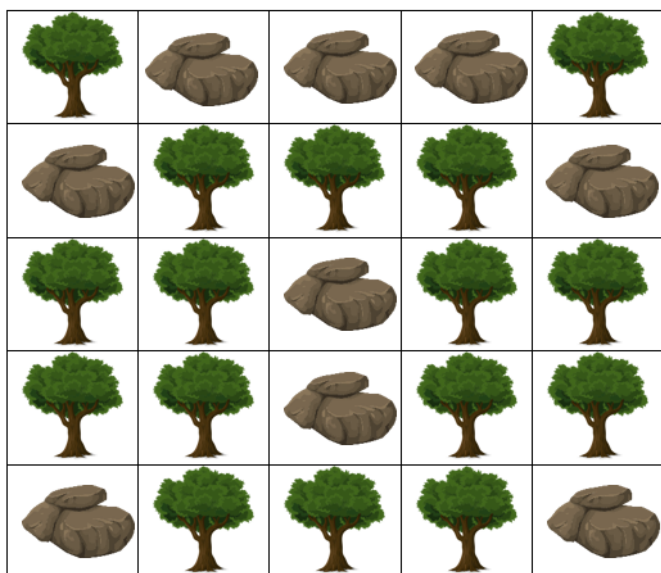
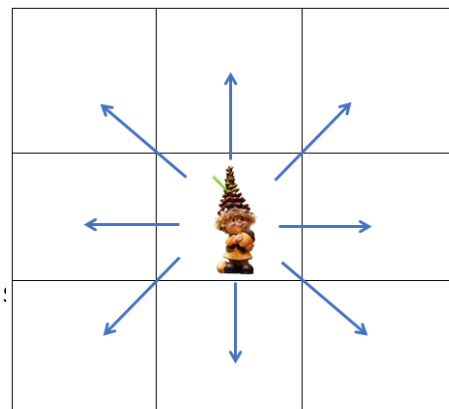
They all have a map with the wood divided into squares.

Each square contains a tree or a rock.

Forest Imps can see in all directions as shown on the right and can :

Forest Imps cannot see through rocks.

The forest map:



Question:

In how many ways can the 5 Forest Imps bury their treasure in the wood without seeing each other?

1 2 3 4 or 5

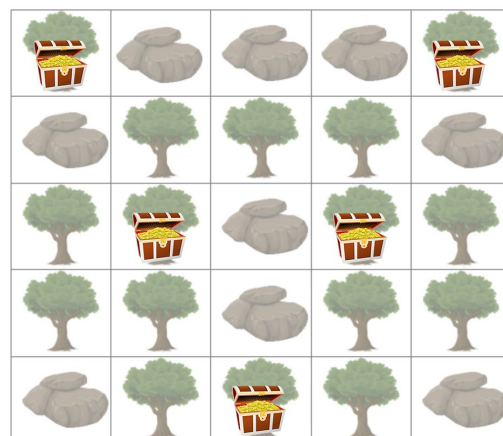
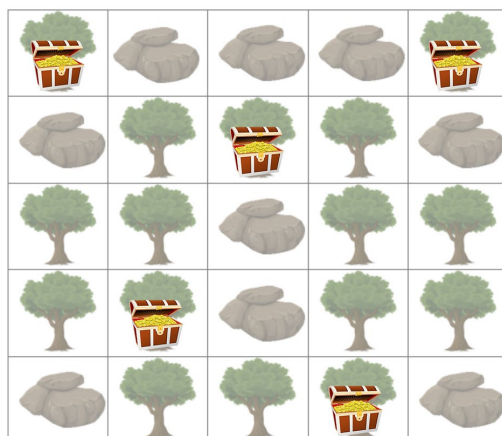
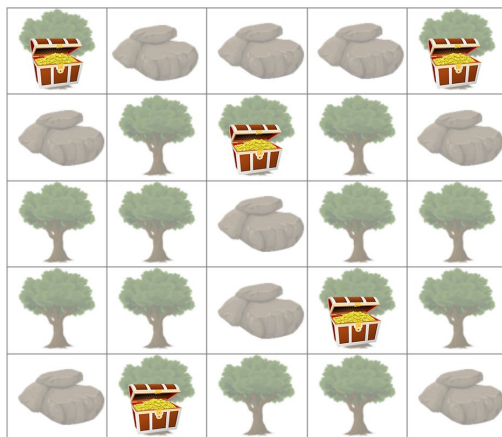
Buried Treasure

Answer:

4

Explanation:

There are 4 possible arrangements, as shown below:



It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp)

In order to solve this task, all the possibilities of arranging the 5 treasures on the map must be generated. This could be achieved using the Backtracking programming technique. It is easier to start with the first line of the map where there are only two trees located.

If there is any ambiguity in the explanation about what constitutes a line of sight, this is removed by the fact that it is a multiple choice question.

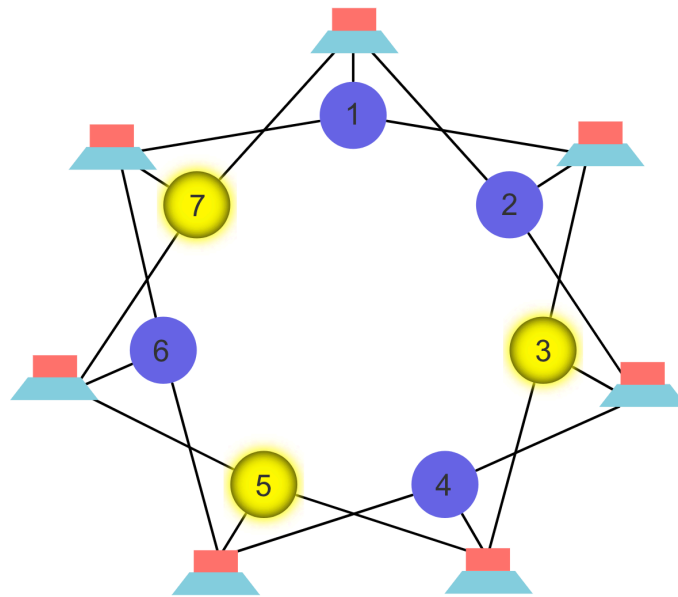
Below is a network of lights and switches.

When you use any of the switches, the three lights connected to this switch will change their state: from off to on, or from on to off.

Question:

Switch on all the lights!

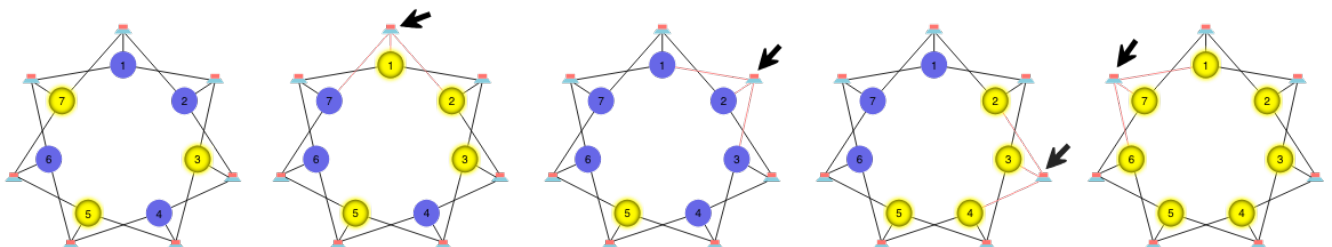
Click on one switch at a time in order to use it.



Answer:

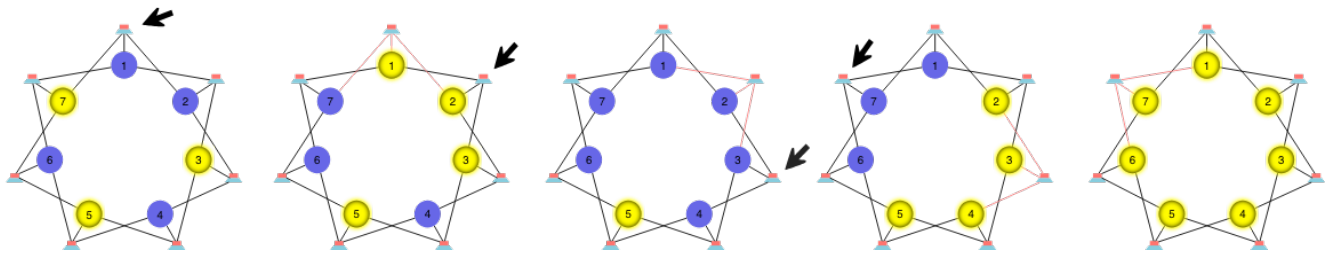
The switches next to light 1, 2, 3, and 7 may be used, in any order, to switch all the lights on.

Explanation:



It can be useful to reason backwards from the desired final state of the network. In order to make the last switch switch on its three lights, all these lights must be off before. Hence, at first we need to switch off a few lights. By using the switches close to light 1 and then light 2, light 7 and then lights 1, 2, and 3 are switched off. Afterwards, only light 5 is on. Now, the switches close to light 3 and light 7 can be used to switch all other lights on.

Switch On



To search for a solution several properties of the network might help:

- Using twice (or an even number of times...) a switch doesn't help: Any second switch action "undoes" the first
- The effect of using any switches X and Y (in this order), is the same of using Y before X.

Thus, the number of sequences of switches to try is in fact limited: At worst all the subsets of the seven switches. There are 128 subsets of a set of 7 elements; it is also worth noting that each of these sequences has a different total effect on the network. Therefore, since there are only 2^7 (= 128) different configurations of the network, each of them can be obtained from any initial configuration.

It's Computational Thinking:

CT Skills - Decomposition (DE), Abstraction (AB), Modelling and Simulation (MS), Algorithms (AL)

Concepts - Abstraction (Ab), Data Representation (Dr), Data Interpretation (Di), Specification (Sp), Algorithms (Al), Implementation (Imp), Digital Systems (Ds), Interactions (In), Impacts (Im)

For many tasks, we know the goal precisely, and we know the initial state we are starting with. In this Bebras task, the goal state of the light network is the one with all lights on, and the initial state is the one with lights 3, 5, and 7 on (and all others off).

In the early days of "Artificial Intelligence", problem solving by computing systems often was understood as a search for a sequence of actions from an initial state to a goal. Such a perspective makes sense in particular if there is a finite set of possible actions, and a finite set of objects that can be manipulated by these actions. For realising this search process, many techniques and strategies were developed. "Means-ends analysis", for instance, chooses an action that reduces the difference between a current state and a goal state. In this Bebras task, however, we could see that it may be helpful to also look for actions that transform some other state into the goal state. Searching from both the initial state and the goal state is called "bidirectional search".
