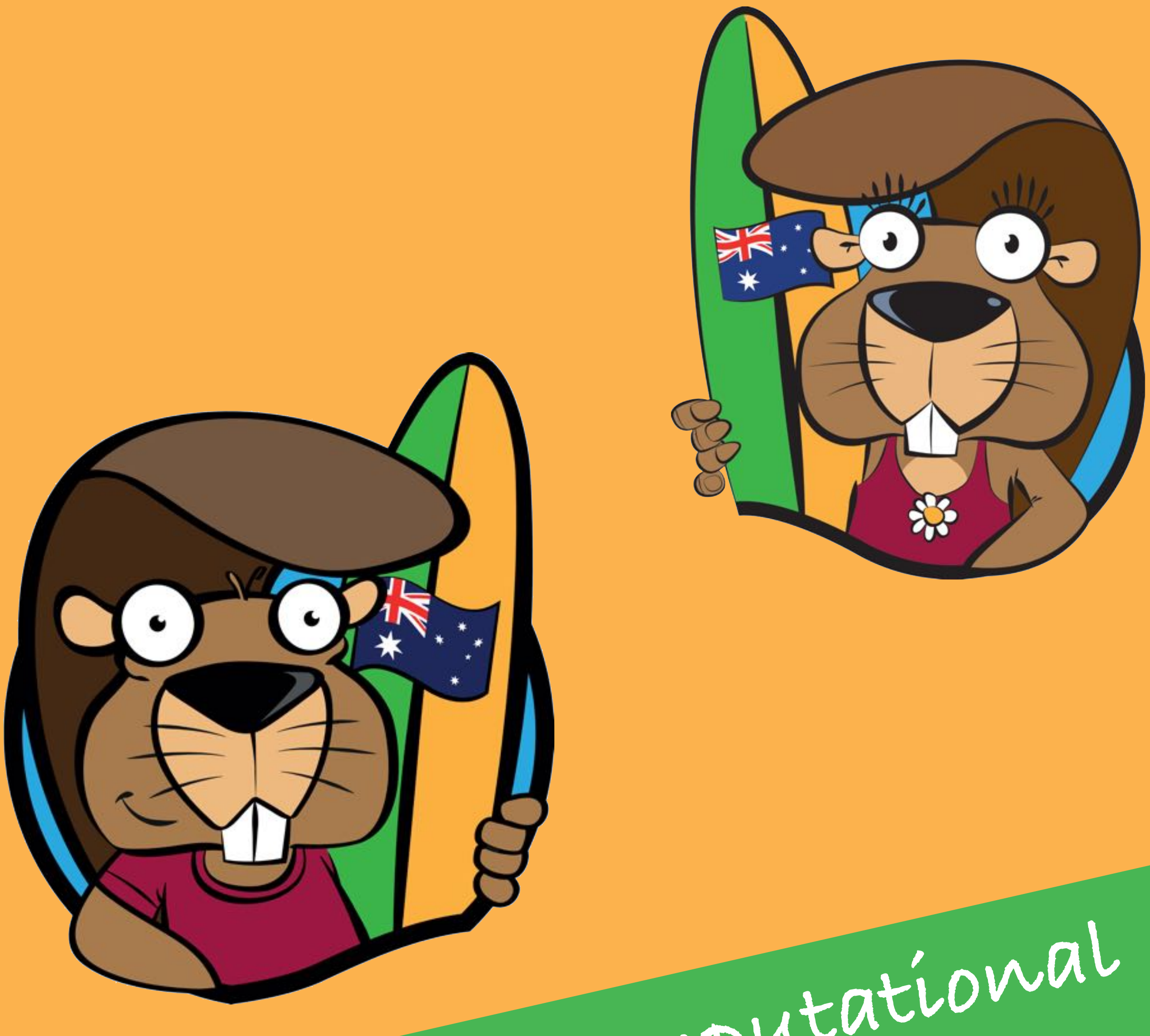
digitalcareers

bebras.edu.au



Bebras Australia Computational Thinking Challenge

Tasks and Solutions 2017

Editor:

Sarah Hobson

Thank You!

We would like to thank the international Bebras community for allowing us to use the tasks that have been developed for the new round along with some tasks from previous years. Bebras is a collective effort of many countries and we are privileged to be part of the Bebras community. This year, Bebras Australia runs with a similar set of questions as the UK and we would like to thank Bebras UK for their generous support.

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

Special thanks go to Eljakim Schrijvers and Daphne Blochuis 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, Sue Sentence and Sarah Hobson.

Acknowledgements

We would like to thank the Australian Government for funding the Digital Careers Program which manages Bebras Australia.

We would also like to thank the Australian Curriculum, Assessment and Reporting Authority (ACARA) for providing advice regarding Bebras tasks.

Introduction

About Bebras Australia
The Bebras Australia Computational Thinking Challenge was established in 2014 to enable Australian primary and secondary school students to engage in activities specifically designed to test their problem solving skills. Computational Thinking skills underpin skills development outlined in the Digital Technologies.

Everyone can do it
The challenges consist of a set of short questions called tasks that are delivered online. The tasks are developed as comprehension style questions rather than a math or science focus and can be answered without prior knowledge about computational thinking.

To solve the tasks, students are required to think about information, how information is organised, how it is processed, and make decisions based on that information. In so doing students demonstrate their ability with an aspect of computational thinking.

Work alone or in teams
The Challenge is broken in to age categories and within each category there are 15 tasks to be solved. The tasks have three levels of difficulty – Easy, Medium and Hard (each consisting of five questions). The questions get progressively more difficult as students' progress through the age categories.

Bebras is aligned with and supports the Australian Curriculum: Digital Technologies. Computational Thinking is one of the key principles that underpin the Digital Technologies Curriculum. Bebras is a great way to practice those skills in a fun and safe environment.

Bebras: International Contest on Informatics and Computer Fluency

UK Beaver

Bebras is an international initiative whose goal is to promote computational thinking for teachers and students (ages 8-18; Years 3-12).

The international Bebras Community jointly develops the tasks for the annual Bebras Week. Australia, France, Japan, Canada, The Netherlands, Austria, Switzerland and Germany use a common online system, which is being developed and managed by the Dutch company Eljakim IT.

Further information about Bebras International and its member countries is available at www.bebras.org

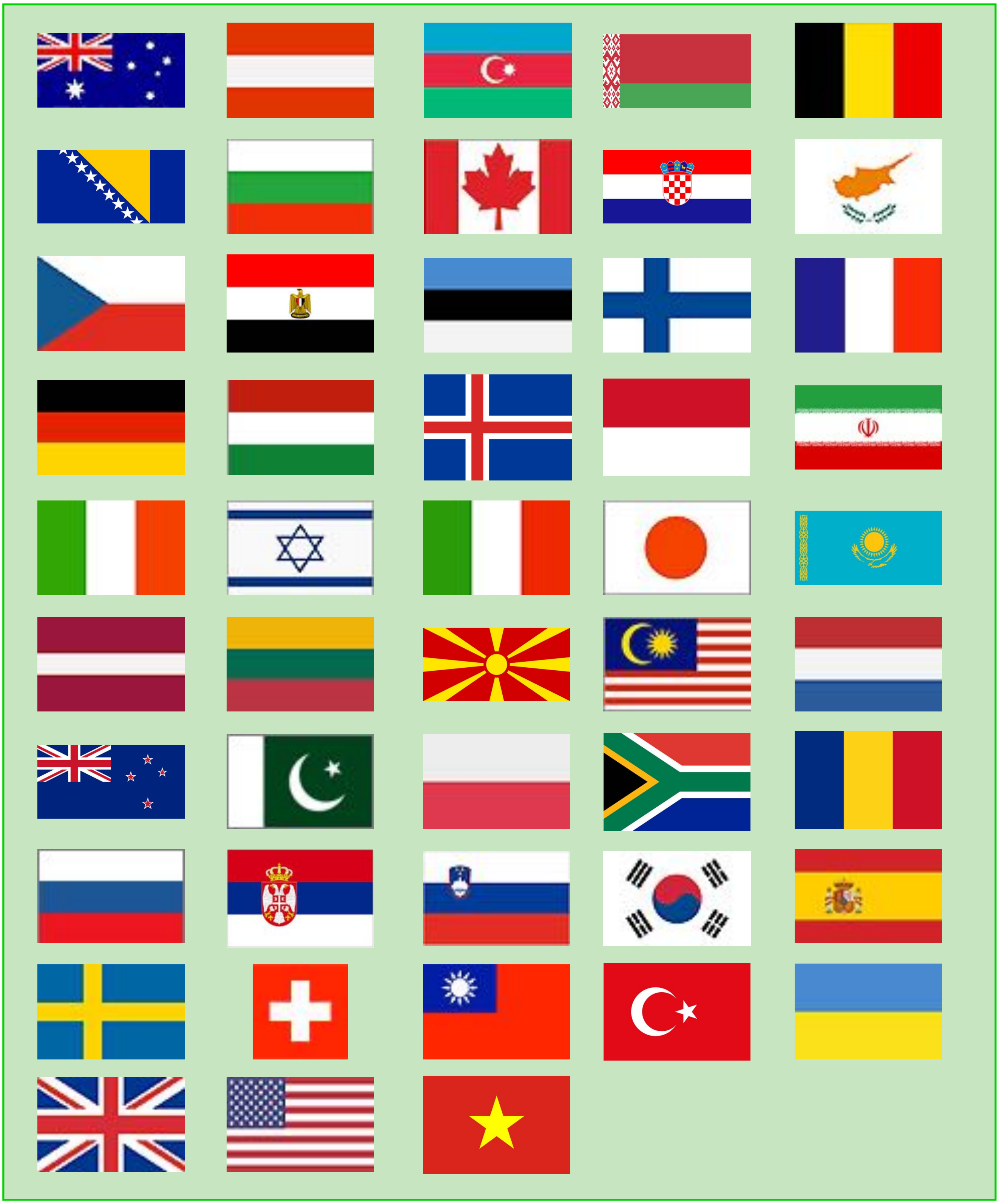
Japanese Beaver

Lithuania

Spain

Participating Countries

Each task in this booklet has a flag indicating the country of origin. However, many people were involved in the further editing, translating and providing additional material.
Digital Careers are indebted to the generosity of spirit and community of Computer Scientists around the world!



Welcome Australian Students

Across Australia, in all States and Territories, teachers from schools and home schools alike have registered as Bebras coordinators. We have plotted their origins on this map as a celebration of the diversity of the many places our coordinators and their students come from.

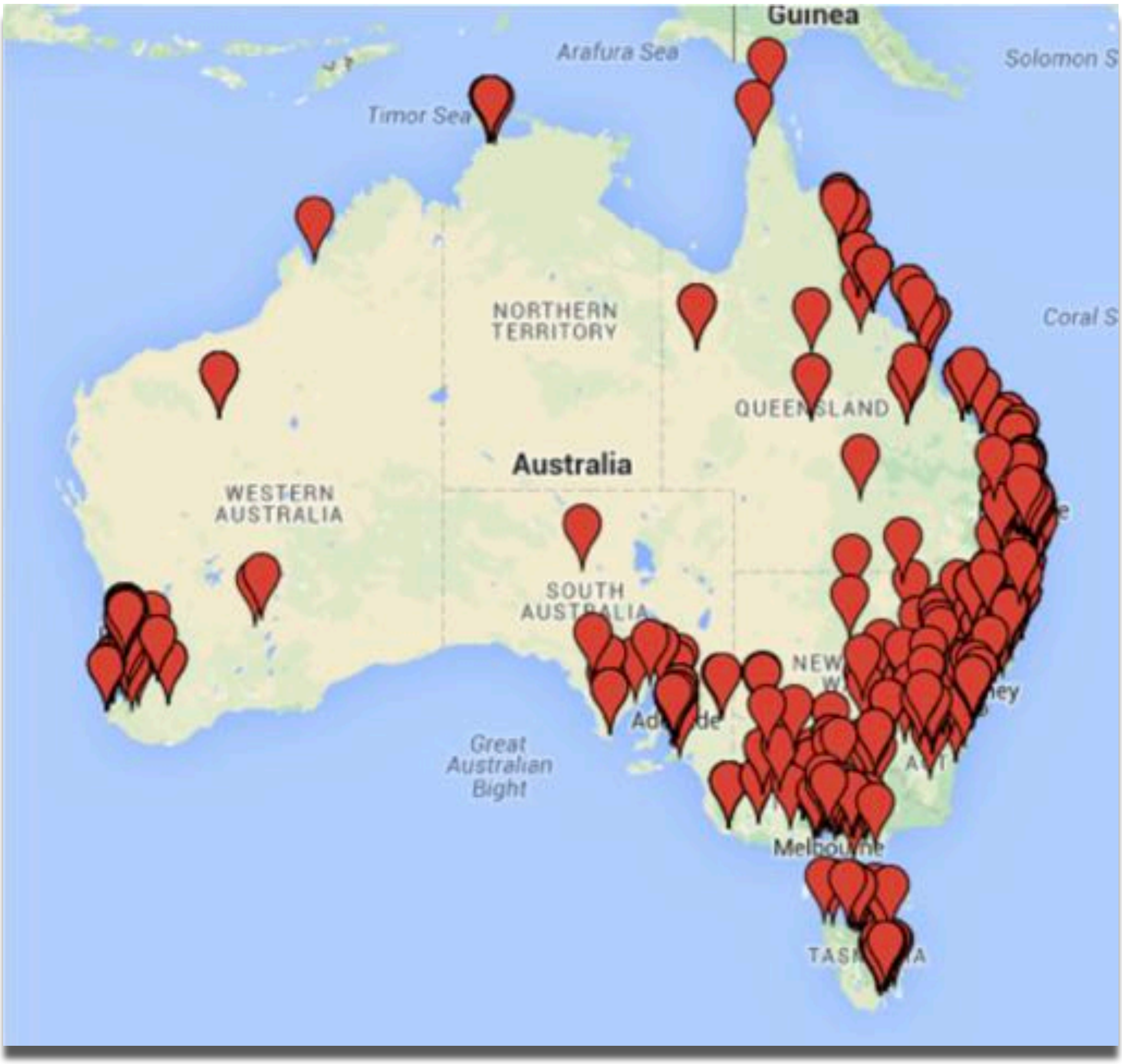


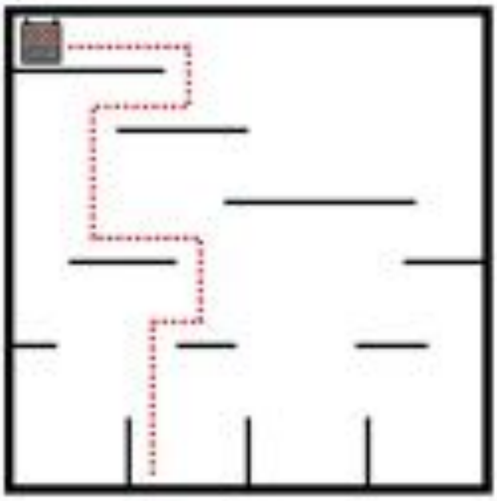
Table of tasks

Falling Robot

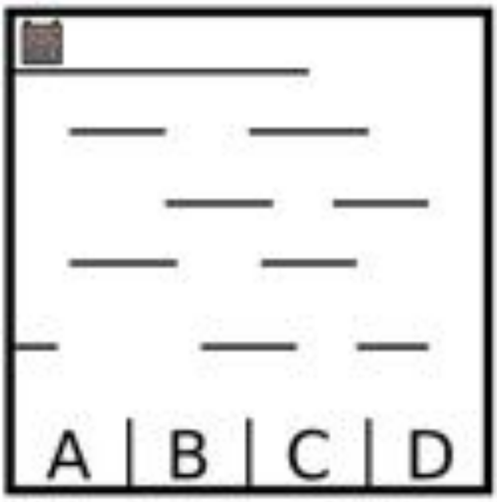
Years 3+4 A Years 7+8
Years 5+ 6 Years 9+10
Years 11+12

A robot moves through a vertical maze. The maze consists of various platforms. The robot begins in the upper left corner and then moves to the right.
When it reaches the end of a platform, it falls down onto the platform below.
As soon as the robot lands it changes direction. Eventually the robot reaches one of the buckets at the bottom of the maze.

The following image gives an example of how the robot will move down.

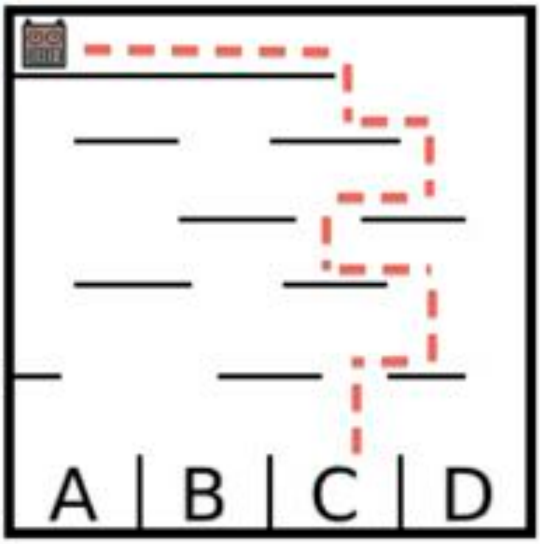


Which bucket will the robot reach in the maze below?



Answers:

A	Bucket A
B	Bucket B
C	Bucket C
D	Bucket D



Answer:

C

It's Computational Thinking:

Data interpretation - patterns
Algorithms - following

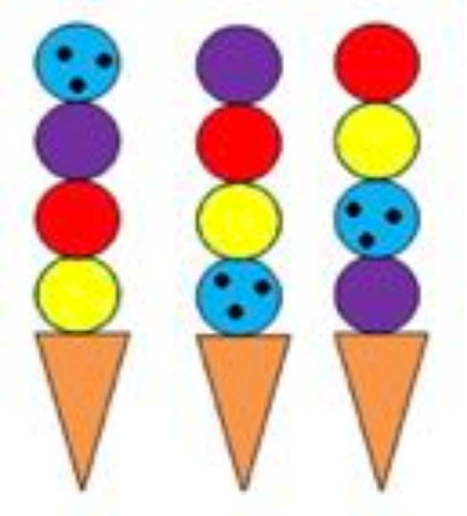
The robot follows a very basic rule, or algorithm. The student needs to understand that algorithm, either through its description, or by looking at how it performs on the left maze, then simulate that algorithm on the right maze.

Ice Cream Machine

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

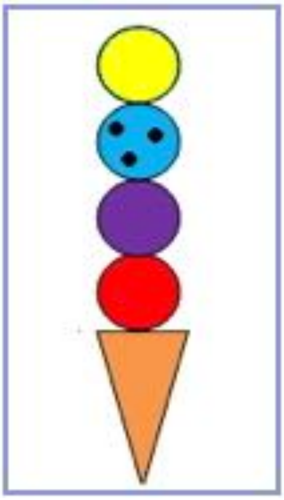
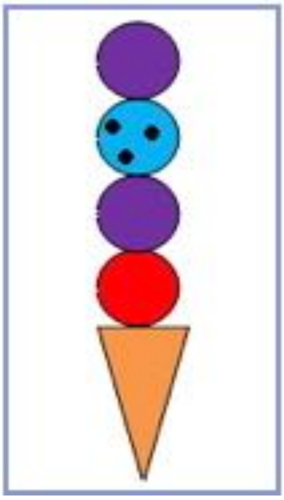
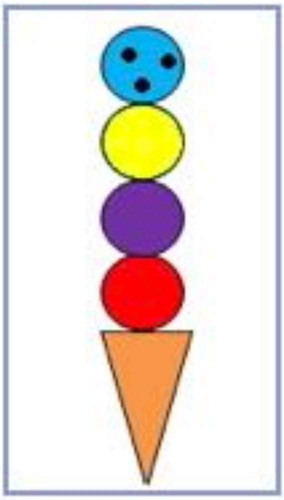
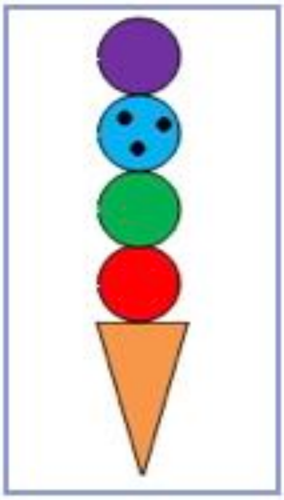
This special ice cream machine creates cones with 4 scoops of ice cream.

It does so in an ordered way. Here you see, from left to right, the last 3 ice creams that the machine has made.



Which ice cream will the machine produce next?

Answers:

A	B	C	D
			

Ice Cream Machine

Answer:

A

Explanation:

The order always is yellow – blue – purple – red – yellow – blue – purple – red ... In the picture you can see that red is followed by green, green is followed by blue, ...

It’s Computational Thinking:

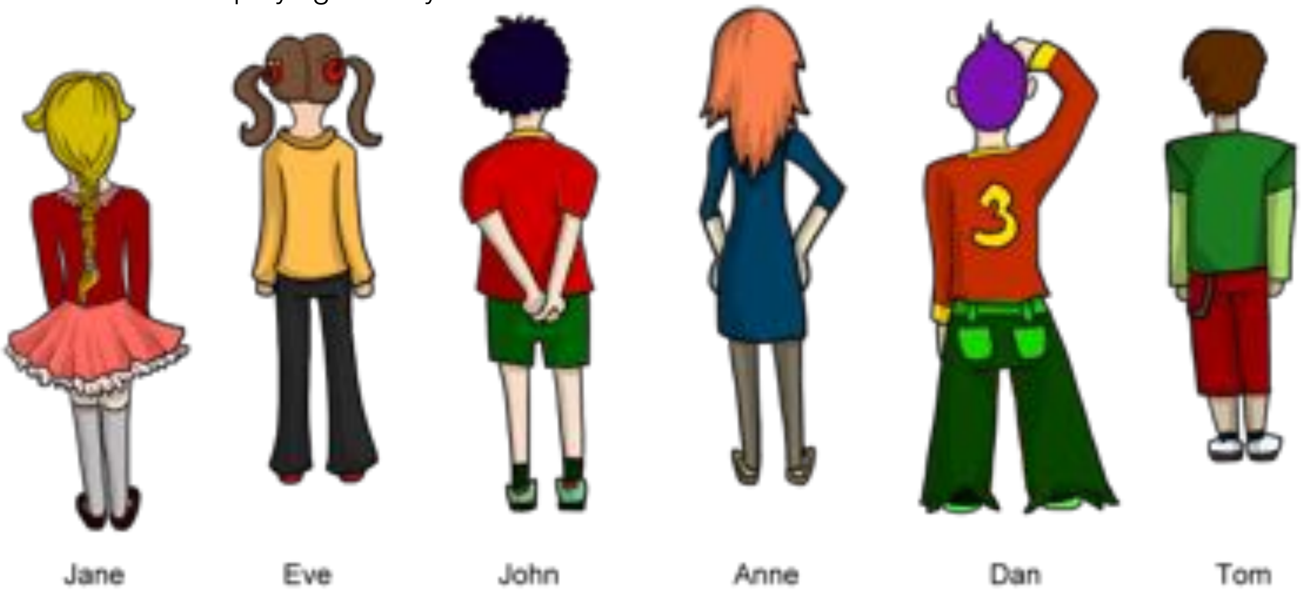
Data interpretation - patterns
Algorithms - following

Detecting the operation of an algorithm is sometimes required in informatics. In many fields of information technology, computer scientists observe traces of computer activity and are checking the semantics (meaning) of programs.

Broken Window

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

Six children were playing in the yard.



One of them threw a ball and broke Mr. Beaver's window.
Mr. Beaver only saw the back of the child running away.
The child had a red shirt and short dark hair.

Question:

Who broke the window?

Answer:

John

Explanation:

Only three of the children wear a red shirt: Jane, John and Dan. But Jane has long blonde hair and Dan's hair is violet. So, it had to be John.

It's Computational Thinking:

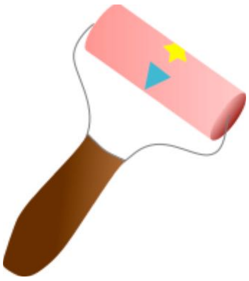
Abstraction
Data collection – properties
Data interpretation – patterns

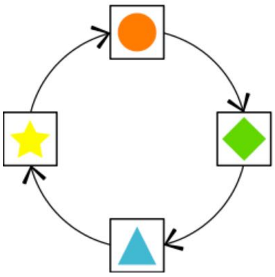
This task involves categorising information. One way that information is stored on a computer is called a database. A database stores many pieces of information which share common attributes. For example, in this problem, every child has a hair colour, a shirt colour, a length of hair. But each individual child has specific values for each of these attributes: For instance, Jane and Eve both have the "hair colour" attribute, but the values differ (blonde and brown). Determining which attributes are important and being able to select data according to some criteria is an important feature of a database.

Bebras Painting

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

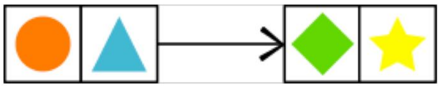
The Beaver children have found a magic roller.
The roller replaces a shape in a painting with the next shape shown by the arrows below.





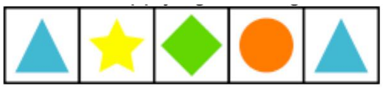
Example:

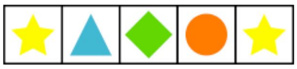
When Ben uses the magic roller to paint over the painting on the left, he gets the painting on the right.



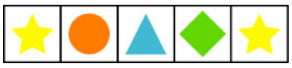
Question:

What will the painting below look like after using the magic roller?

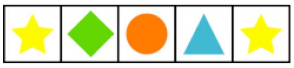




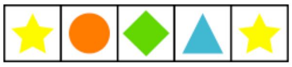
A



B

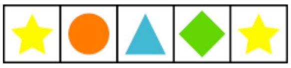


C



D

Answer:



Explanation:

B results after applying the algorithm described above.

The other answers are not correct. A and C both have an incorrect second symbol, which should be a circle. D is incorrect as only the first two symbols are correct.

It's Computational Thinking:

Breaking problems down into parts (Decomposition)
Data representation - symbolism
Algorithms – following

This task involves following an algorithm. Algorithms are sequences of instructions that need to be followed in a specific order. This algorithm is a simplified version of a computer display algorithm, where pixels are replaced based on their values (e.g. to change the intensity of an image, or to apply a filter, or another image transformation).

Shelf Sort

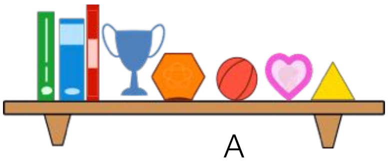
Years 3+4 A Years 7+8
Years 5+ 6 Years 9+10
Years 11+12

Beatrix is trying to rearrange her shelf. She has two rules:

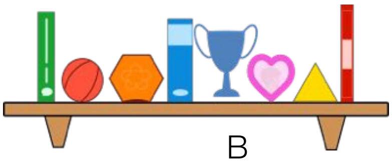
1. Rectangular items must not be next to each other.
2. Circular items must not be next to rectangular items.

Question:

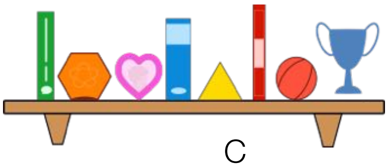
Which one of these shelves has followed her rules correctly?



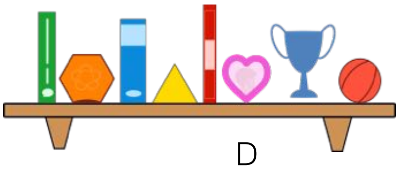
A



B



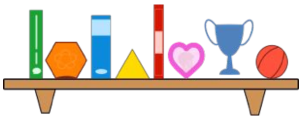
C



D

Answer:

D



Explanation:

A can't be right as there are rectangular objects together.
B can't be right as there is a circular object next to a rectangular object.
C can't be right as there is a circular object next to a rectangular object.
The only correct answer can be D as it follows the rules stated.

It's Computational Thinking:

Breaking problems down into parts (Decomposition)
Data representation – symbolism
Data interpretation – patterns
Algorithms – following

You can describe the rules that Beatrix set for herself as an algorithm: A sequence of instructions or a set of rules to get something done. In this task, the algorithm is written for a human, rather than for a computer to understand.

Data can take many forms such as pictures, text or numbers each with different attributes and values. When we look at data, we are looking for patterns in those attributes and values. By identifying the patterns, we can make predictions, create rules and solve more general problems.

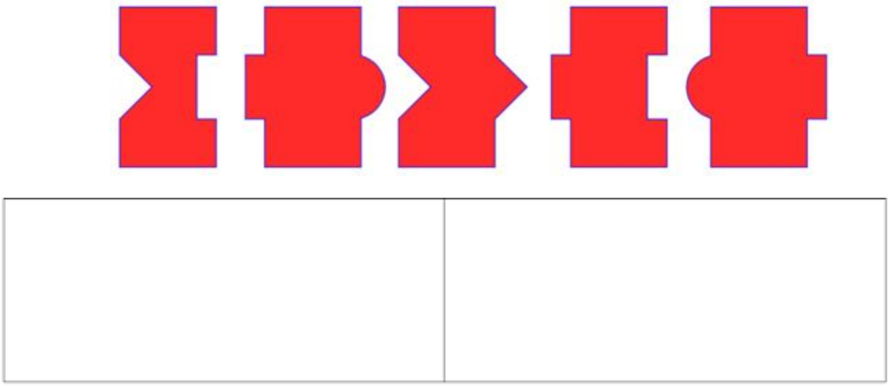
Puzzle

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

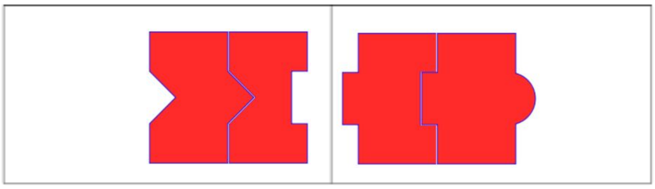
The beavers have a new puzzle. They want to make pairs!

Drag two pieces that snap together into the space below to make a pair.
Then try and make another.

Question:
Try and make two pairs then click on 'Save'.



Answer:



Explanation:

You might be tempted to form a pair from the two pieces initially on the left, but then the other two pieces cannot be rearranged to form a pair. You have to look a little further than the obvious to solve this task.

It's Computational Thinking:

Abstraction
Data representation - symbolism
Data interpretation – patterns
Digital Systems

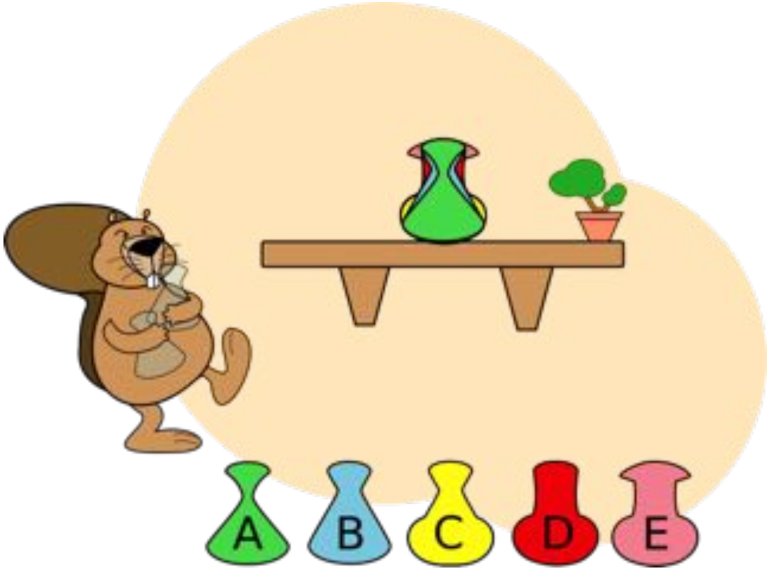
People love to optimise things: find how to run as fast as possible or the shortest route, make as much money as possible, or form as many puzzle piece pairs as possible. If the optimisation problems are particularly big or involve a lot of data, it is good idea to let the computer do the optimising.

In Computer Science, there are many ways to optimise. One of them is the 'Greedy' method, where you just choose the next step based upon the best improvement to the solution that can be achieved in the current step. Here, the greedy step is to form a pair of the two pieces on the left. A nice fact about computing is that, in most cases, it does not help to be greedy. Most of the time the first best step will prevent you from finding the best solution.

Bottles

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

A Beaver puts five bottles on a table.
He places them so that every bottle has a bit showing.
He places the first bottle at the back of the table and puts each new bottle in front of those already placed.



Question:
In what order are the bottles placed when they appear as shown in the picture?

E D C B A
D B C A E
E C D A B
D C E B A

Answer:

E D C B A

Explanation:

You can try to solve this in different ways. If you figure out that the thin bottle should be at the front otherwise it will disappear behind one of the other bottles, you already know that A has to be in front. You can try that with each bottle in turn until you solve the task. You can also check which bottle is large at the top or middle, since in those places the bottles differ most. Small bottles need to be in front.

It's Computational Thinking:

Abstraction
Data representation - symbolism
Data interpretation – patterns

This is basically a sorting problem. You are asked to sort the bottles in a specific way. Here, shapes and sizes are important. One has to decide the ordering according to these properties.

Party Guests

Years 3+4	C	Years 7+8	A
Years 5+ 6	B	Years 9+10	
		Years 11+12	

To arrange a dinner party Sara the beaver needs to talk to five friends:

Alicia, Beat, Caroline, David and Emil.

Sara can talk to Emil right away. However, to talk to her other friends, there are a few points to consider:

1- Before she talks to David, she must first talk to Alicia.
2- Before she talks to Beat, she must first talk to Emil.
3- Before she talks to Caroline, she must first talk to Beat and David.
4- Before she talks to Alicia, she must first talk to Beat and Emil.

Question:

In what order should Sara talk to all of her friends if she wants to talk to all of them?

Drag the names into the right order.

Alicia **Beat** **Caroline** **David** **Emil**

⇒ ⇒ ⇒ ⇒

Answer:

Emil ⇒ Beat ⇒ Alicia ⇒ David ⇒ Caroline

Explanation:

This task has to do with dependencies.

- Emil is the only friend with no dependencies so he must go first.
- Beat only depends on Emil so he can go second.
- Alicia depends on Beat and Emil, so she can go third.
- David depends on Alicia so he can go next,
- Finally Caroline depends on Alicia and David so she can go last.

No other sequence can fulfil the dependencies.

It's Computational Thinking:

Breaking down problems into parts (Decomposition)
Algorithms – following and describing
Data interpretation – modelling

Satisfying dependencies is a computational problem that happens frequently in real life.

This problem of dependency and ordering can be modelled with a graph. A graph is composed of vertices (here the friends) and arcs from some vertices to other ones (here there is an arc from X to Y if X needs to appear after Y. Our graph here is special: since there exists an ordering, it has no cycles. A cycle would be a path that starts at one vertex, follows some arcs and returns to the starting vertex. The graph in this problem is a directed acyclic* graph (DAG). The problem is then called a topological sort: we sort the vertices according to the dependency relation.

One way to do so is to select all the vertices where no arcs exit (hence those tasks can be done first), and put them at the beginning of our answer. Then we can remove these vertices from the graph, and continue with the new graph.

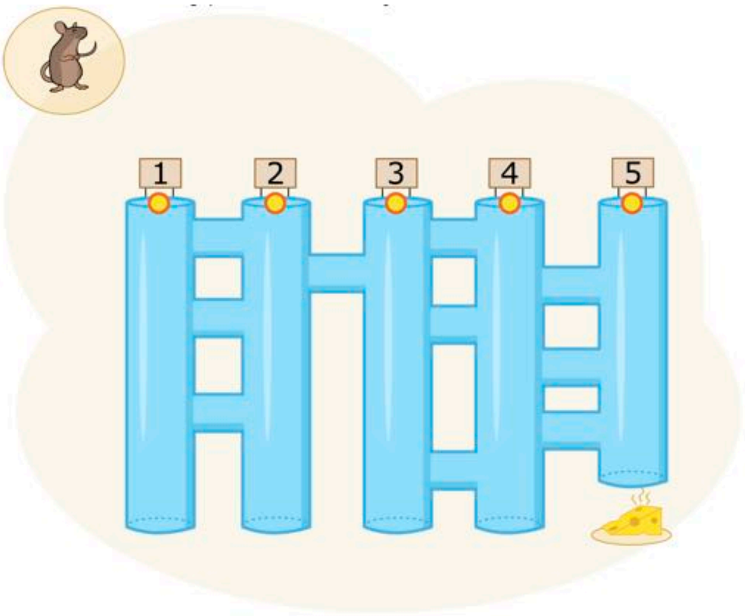
The existence of a vertex where no arcs exit is guaranteed by the fact that the graph is acyclic* (*having no graph cycles).

Tube System

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

A mouse is at the entrance of a tube system. It wants to reach the cheese at the end of tube 5. The mouse always follows these commands:

1. Go downwards until a crossing
2. At the crossing, move through to the next vertical tube
3. Go to command 1



Question:

In which tube should the mouse start so that it reaches the cheese?

1 2 3 4 or 5

Answer:

3

Explanation:

From tube 1 the mouse always reaches tube 3.
From tube 2 it reaches tube 1.
From tube 4 it reaches tube 2.
From tube 5 the mouse gets to tube 4.

It’s Computational Thinking:

Algorithms – following and describing

Many robots are programmed so that they have to follow exact commands. This mouse does the same thing: it follows the commands ‘go downwards’ and ‘change directions at the next crossing’ over and over again. These kinds of commands depend on the choice of the tube entrance as to which way the mouse runs in the tube system. Most computer programs are deterministic: if you always input the same data, the program performs the same calculations and delivers the same output.

Beaver Code

Years 3+4	C	Years 7+8	A
Years 5+ 6	B	Years 9+10	
		Years 11+12	

Barbara has been given two stamps.

With one she can produce a little flower, with the other a little sun.

Being a clever girl, she thinks of a way to write her own name by using the code below:

Letter	B	A	R	E	Y
Code	✿	☀ ☀	☀ ✿ ☀	☀ ✿ ✿ ✿	☀ ✿ ✿ ☀

So Barbara” becomes:

✿ ☀ ☀ ☀ ✿ ☀ ✿ ☀ ☀ ☀ ✿ ☀ ☀ ☀

She then writes the names of her friends. Unfortunately, they all got mixed up.

Question:

Drag the sun-flower-codes to the names of her four friends.

Abby	
Arya	
Barry	
Ray	

☀ ☀ ☀ ✿ ☀ ☀ ✿ ✿ ☀ ☀ ☀

✿ ☀ ☀ ☀ ✿ ☀ ☀ ✿ ☀ ☀ ✿ ✿ ☀

☀ ☀ ✿ ✿ ☀ ✿ ✿ ☀

☀ ✿ ☀ ☀ ☀ ☀ ✿ ✿ ☀

Answer:

3

Explanation:

This problem is most easily solved by noting that Abby starts with an A and a B and so we look for a code with two suns and a flower at the start. There is only one of these so this is assigned. Next it is noted that Arya's code begins with three suns and a flower. Again, there is only one of these so this is assigned. By continuing in this way, all the codes are quickly assigned to the correct names.

It’s Computational Thinking:

Breaking down problems into parts (Decomposition)
Data representation - symbolism
Data interpretation - patterns
Algorithms – following and describing
Digital systems

Often in Computational Science, instead of storing data in a simple and straightforward way, we can devise a scheme to store it more efficiently, using less space.

For instance, computers store information about characters that can be typed on the keyboard in what is called an ASCII encoding. Each letter corresponds to a different sequence of 8 bits (0's and 1's). In ASCII, every character takes the same amount of space.

However, letters have different frequencies in texts (for example, the letter “E” is the most common letter in English words), and we can use these frequencies to improve our encoding.

Specifically, we encode frequent letters with smaller codes: in this question, B should be frequent and takes one symbol, A two, and the other letters more. There is a famous and widely used algorithm to do this for texts, named the Huffman coding. You cannot, however, use any encoding you wish: you have to make sure the code is unambiguous. For example, suppose the code was the following: Letter B was one flower, letter A was two flowers.

What do two flowers mean? It could be BB or A, but we have no way to know which one for sure. One way to achieve unambiguity is to make sure the code is prefix-free; that is if we take the code of a letter, it is not the beginning of any other code.

Also, since the Huffman code used depends on the text itself (it depends on the frequencies of letters), it is necessary to store the correspondence between the code and the actual letters. This takes a bit of space, but is negligible for long texts.

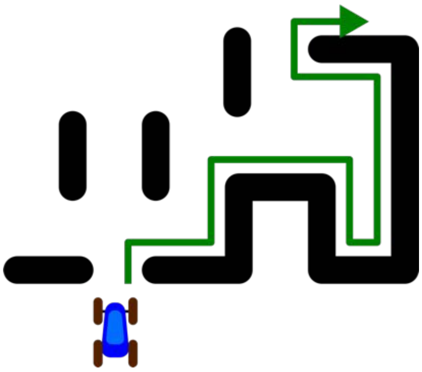
Mazes

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

A robotic car uses a simple rule to drive through a maze:

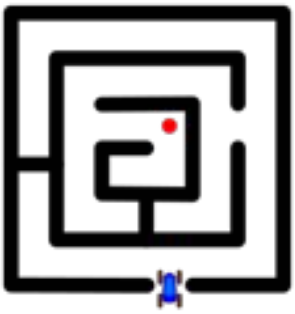
Turn right whenever possible.

The picture on the right gives an example of how the robot would drive through a maze.

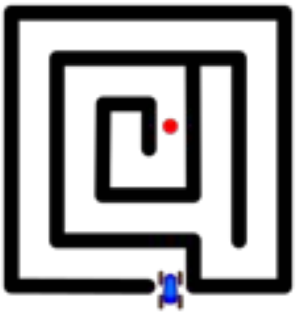


Question:

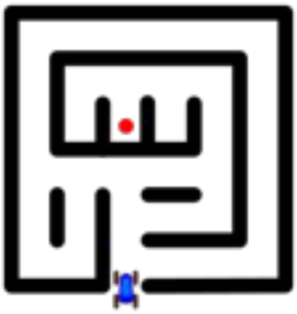
In how many of the following mazes will the car reach the red dot if it uses this system?



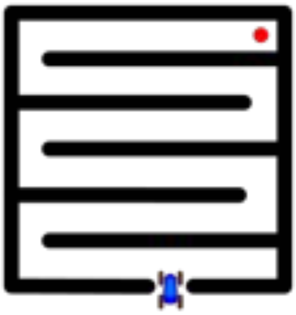
Maze A



Maze B



Maze C



Maze D

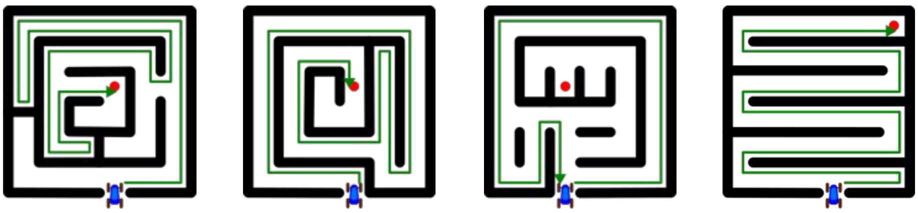
Choose from: 0 1 2 3 or 4

Answer:

3

Explanation:

In the pictures below, the green arrow indicates the path taken by the car. In Maze C the whole centre part of the maze is not visited and the red dot is not reached. In the 3 other cases the red dot is reached.



It's Computational Thinking:

Algorithms - following

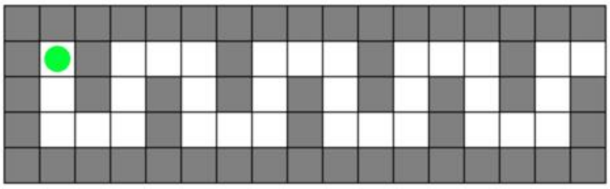
The method which is used by our car, is called the wall follower. It is one of the simpler techniques (algorithms) to solve a maze for which you do not know the layout in advance. By following this rule, you will never get lost: you will always return to the starting point eventually. However, as can be seen by our example, it does not guarantee that you will visit the entire maze.

Robot Exit

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

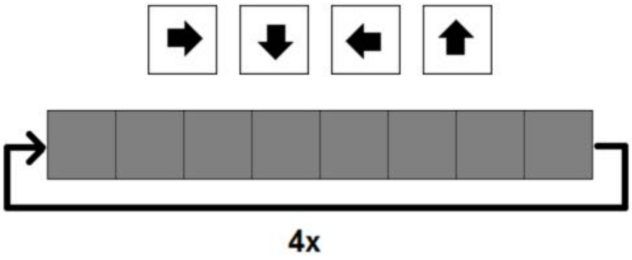
Help the green robot to exit the maze.

The robot will repeat your instructions 4 times.

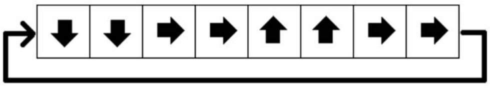


Question:

Drag the arrows to form a set of instructions.



Answer:



↓ ↓ → → ↑ ↑ → →

Explanation:

In mobile robotics, maze problem solving is one of the most common problems. To solve this problem, an autonomous robot is used. Mazes can be of different kinds; having loops, without any loops, grid systems or without a grid system. In this short loop maze algorithm, the robot is instructed to follow a preference of directions.

It's Computational Thinking:

Algorithms – following
Algorithms – describing

In computer programming, a loop is a sequence of instruction's that is continually repeated until a certain condition is reached.

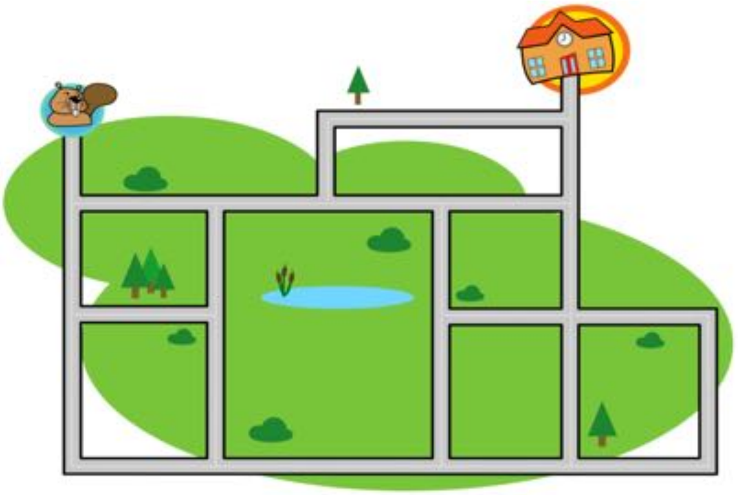
Typically, a certain process is done, such as getting an item of data and changing it, and then some condition is checked such as whether a counter has reached a prescribed number.

If it hasn't, the next instruction in the sequence is an instruction to return to the first instruction in the sequence and repeat the sequence. If the condition has been reached, the next instruction "falls through" to the next sequential instruction or branches outside the loop.

A loop is a fundamental programming idea that is commonly used in writing programs. It is sometimes referred to as 'repetition' or 'iteration'.

Car Trip

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



A self-driving car needs to take a student to school.

The car is programmed so that it only uses these 3 instructions:

Left: turn 90° left

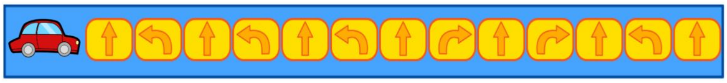
Right: turn 90° right

Forward: go forward until you cannot go forward anymore

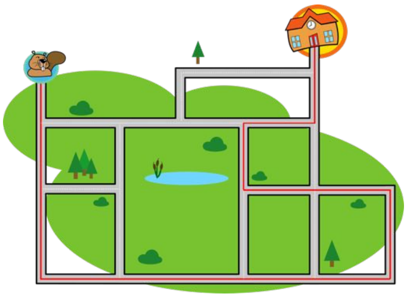
Question:

Write a set of instructions (a program) that will get the beaver to his school. You can do this by dragging the three instruction blocks next to the car.

Answer:



↑ ↰ ↑ ↰ ↑ ↰ ↑ ↱ ↑ ↱ ↑ ↰ ↑



Explanation:

The important thing for participants to remember is that there is no forward movement when turning 90 degrees, so the 'straight' command has to be entered between every turn.

It’s Computational Thinking:

Breaking down problems into parts (Decomposition)

Algorithms – following

Algorithms – describing

The task is similar to giving instructions to a robot: writing a computer program requires step by step execution. Programs are essential to our use of computers: they tell computers what sequences of operations they must do. Computers and robots are good at computing fast, doing repetitive things, but they cannot think just by themselves, and require instructions to perform tasks. As shown in this task, the order of the operations is very important: the right set of instructions in the wrong order will not give the expected result.

Party Banner

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

Beaver Bert has a long strip of coloured paper for a party.
The strip has three different colours (yellow, red, blue) in a regularly repeating pattern.
Bert's friend, James, has cut out a section of the paper, as shown in the diagram below.

Y	R	R	B	Y	R	R	B	Y	R	R	. . .	R	B	Y	R	R	B

James says that he will give back the missing piece of paper if Bert can correctly guess the size of the piece cut out.

Question:

How many coloured squares does the missing piece of paper have?

31 32 33 or 34

Answer:

31

Explanation:

We know the pattern ended with YRR, meaning that the James has cut out at least one B. After that, he cuts out some number of sequences of 4 (i.e., YRRB). After that, the right side of his piece of paper must have YR, since the second piece begins with RB. So, the length of his piece of paper is 1 (for B) + 4*X (where X is the number of repeated patterns YRRB) + 2 (for the YR). So, the length of her paper is 4X+3.

Looking at the possible answers, we see that 31/4 has remainder 3: that is, 31 = 4*7 + 3. So, our equation is solved when X=7. None of the other answers can be written as 4X+3.

It's Computational Thinking:

Abstraction
Breaking problems into parts (decomposition)
Data representation - symbolism
Data interpretation – patterns and contexts
Algorithms – following and describing

Finding a pattern in information is important for a variety of problems. For instance, sequences of DNA are composed of patterns, and finding repetitions or substrings that satisfy a certain property is an important research area in genetics and medicine. To solve these sorts of problems, we use text processing algorithms and pattern-matching programs to help determine whether certain strings appear in a sequence of text.

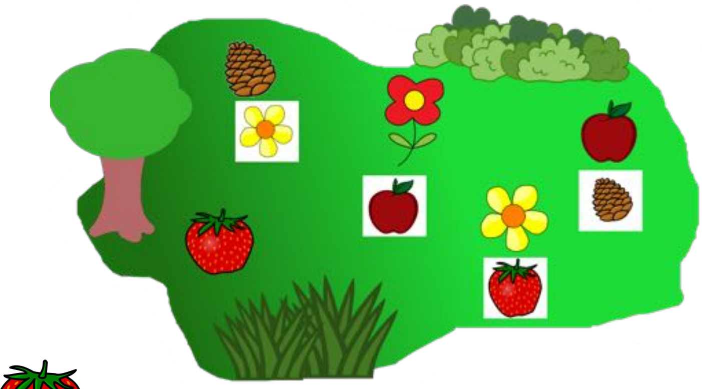
This problem also involves some abstraction, data interpretation and algorithm decomposition: We take a sequence of information and represent it as a formula or equation which we can solve. In order for computer scientists to solve problems, they need to take an explanation and convert it into something more concrete, formalised and mathematical in order to write a program to solve it.

Secret Recipe

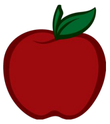
Years 3+4	C	Years 7+8
Years 5+ 6	B	Years 9+10
		Years 11+12

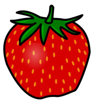
Esther has asked Ivan to cook a special cake made of five ingredients.
She has put labels next to the ingredients in the garden. One ingredient has no label.
The labels tell Ivan in which ingredient must be added next in the sequence.

The garden looks like this:

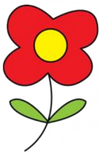


Question:
Which ingredient should be added first?





Answer:



Explanation:

If Esther starts with the flower, she can add all five ingredients in the right order. The first added ingredient must be the one with no referring image.

Choosing the strawberry, she could not have continued to the next as there is no paper with it. The apple is not correct because if she had started with the apple, she would have skipped the red flower. The pine cone is not correct because if she had started with the cone, she would have skipped the red flower and the apple.

It's Computational Thinking:

Data representation - symbolism
Breaking down problems into parts (Decomposition)
Data interpretation – patterns
Algorithms – following

The data structure used here is called a linked list in which there can be an arbitrary number of items. A linked list is a linear collection of data elements that consist of an item and a reference point (pointer) showing the next item. The first item of the linked list is very important as the list starts from there and it is the only point that refers to the whole list.

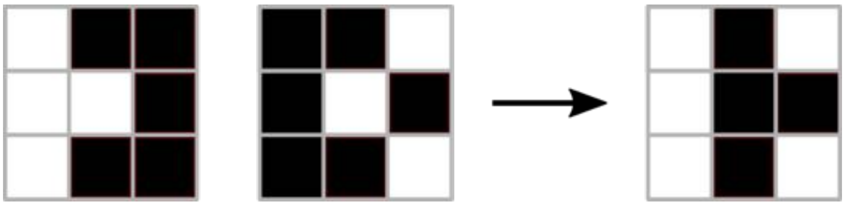
The recipe here is a linked list. The ingredients are the items and each label is the pointer to the next item in line. In other words, the plants are the data and the labels are the pointers. The first component is that ingredient which is not referred by any label, but accompanied by a label.

The benefit of the linked list is that items of different types and sizes can be stored together, just like fruits and flowers in this question.

Paint it Black

Years 3+4		Years 7+8	A
Years 5+ 6	B	Years 9+10	B
		Years 11+12	C

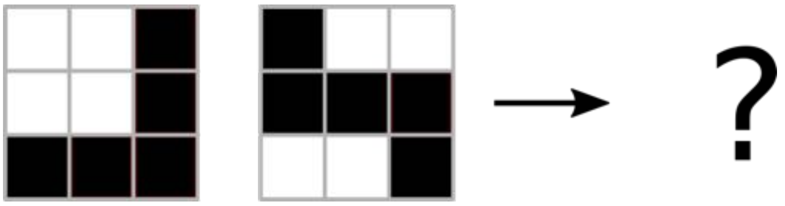
Combining Card A and Card B, you get Card C:



Card A Card B Card C

Question:

How many black cells will Card F have after combining Card D and Card E?

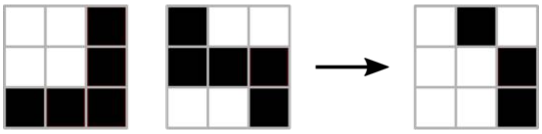


Answer:

3

Explanation:

Combining the cards obeys the following rule. When the colour of the corresponding cells is the same the resulting colour is black. Otherwise the resulting colour is white.



It's Computational Thinking:

Abstraction
Data interpretation – patterns
Algorithms – following
Digital Systems

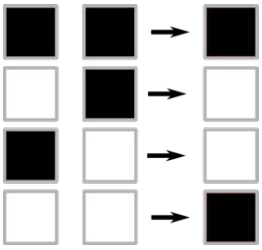
A Boolean circuit is an example of a mathematical computation model. An equivalence is one of the basic Boolean operations. If the white cell is interpreted as 0 or FALSE and the black cell as 1 or TRUE, this operation could be described this way:

$1 \Leftrightarrow 1 \rightarrow 1$

$0 \Leftrightarrow 1 \rightarrow 0$

$1 \Leftrightarrow 0 \rightarrow 0$

$0 \Leftrightarrow 0 \rightarrow 1$



Blossom

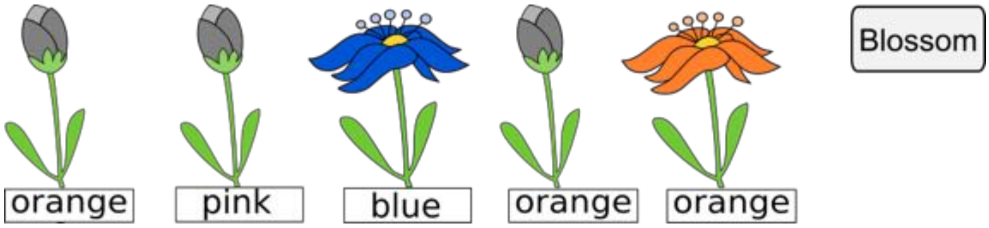
Years 3+4 B Years 7+8
Years 5+ 6 Years 9+10
Years 11+12

Jane is playing a computer game.

First the computer secretly chooses colours for five buds. The available colours for each flower are **blue**, **orange**, and **pink**. Jane has to guess which flower has which colour. She makes her first five guesses and presses the Blossom button.

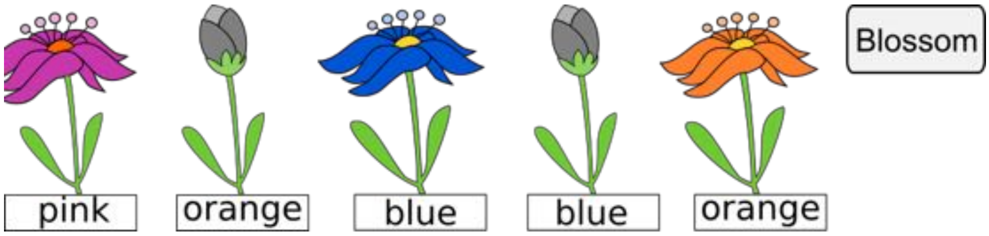
The buds, whose colours she guessed correctly, break into flowers. The others remain as buds.

Jane's first go:



Jane then has another go at guessing and presses the Blossom button again.

Jane's second go:



Question:
What colours did the computer choose for the flowers?

A. blue pink blue orange orange
B. pink blue blue blue orange
C. pink blue blue pink orange
D. pink pink blue pink orange

Answer:

C. pink blue blue pink orange

Explanation:

After two guesses, there are three blossomed flowers. So, we can already see the colour chosen by the computer for the first, third and fifth flower. The colour of the first flower is pink, so answer A) cannot be correct.

For the second flower Jane guessed pink in the first guess and it did not blossom, then she guessed orange and it did not blossom either. As there are only three colours available, the second flower must be blue. This rules out answer D).

Similarly, Jane chose orange and blue for the fourth flower and it still has not blossomed, so it must be pink. And this rules out answer B)

Answer C) must therefore be correct.

It's Computational Thinking:

Abstraction
Data representation - symbolism
Data interpretation – patterns
Algorithms - following

Drawing consequences from events that happened or did not happen is an important ability for solving many kinds of problems. The task is a simplified version of the Mastermind board game. It is simplified because after each guess the player gets complete information about all the flowers. If in each guess the player chooses a different colour for the not-yet-blossomed flowers, then in the third guess he/she can always correctly pick the colours of all the flowers.

Magic Potions

Years 3+4		Years 7+8	B
Years 5+ 6	C	Years 9+10	A
		Years 11+12	A

Betaro Beaver has discovered five new magic potions:

one makes ears longer
another makes teeth longer
another makes whiskers curly
another turns the nose white
the last one turns eyes white.

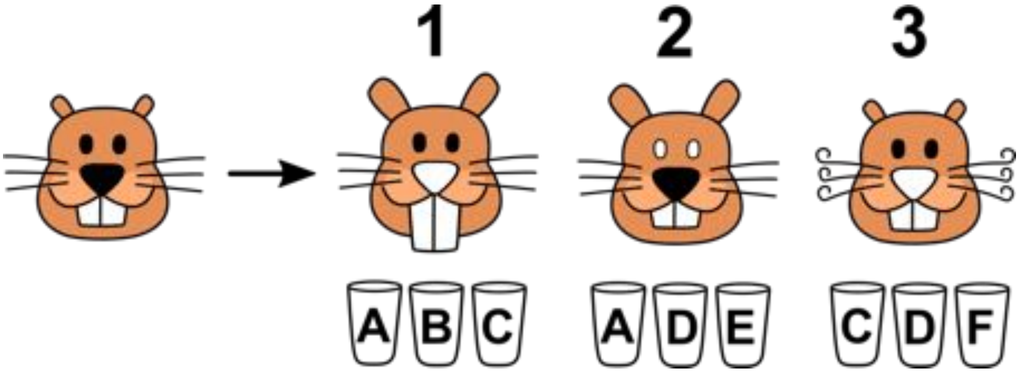
Betaro put each magic potion into a separate beaker. He put pure water into another beaker, so there are six beakers in total. The beakers are labeled A to F. The problem is, he forgot to record which beaker contains which magic potion!

To find out which potion is in each beaker, Betaro set up the following experiments:

Expt 1: A beaver drinks from beakers A, B and C together - the effects are shown in Figure 1.

Expt 2: A beaver drinks from beakers A, D and E together - the effects are shown in Figure 2.

Expt 3: A beaver drinks from beakers C, D and F together - the effects are shown in Figure 3.



Question:
Which beaker contains pure water?

Answer:

Beaker D

Explanation:

Solution 1:
By Experiment 1, none of A, B and C is pure water, since there are three changes that happen to the beaver.
By Experiment 2, either D or E is pure water or the magic potion making his nose white since A is not pure water, from Experiment 1.
By Experiment 3, D and F are pure water or the magic potion making his whiskers curly, since C is not pure water, again from Experiment 1.
Therefore, D is pure water.

Solution 2:
Experiment 1 has three effects, Experiment 2 and 3 both have two effects. Therefore, there is no pure water in Experiment 1 and there is exactly one water beaker in Experiment 2 and Experiment 3. The only common beaker between experiments 2 and 3 is beaker D. Thus, D is pure water.

It's Computational Thinking:

Algorithms – following and describing
Digital systems

In this problem, we have a collection of facts that we need to find new information from. This can be done using logical reasoning. Logic plays an important role in Computer Science. The smallest unit a computer works with is a bit, which has a value of 1 (true) or 0 (false). All other information in a computer is stored using a specific combination of bits. The computer uses logic to figure out what decisions it should follow, and each of these decisions is based on whether certain bits are set to true (1) or false (0).

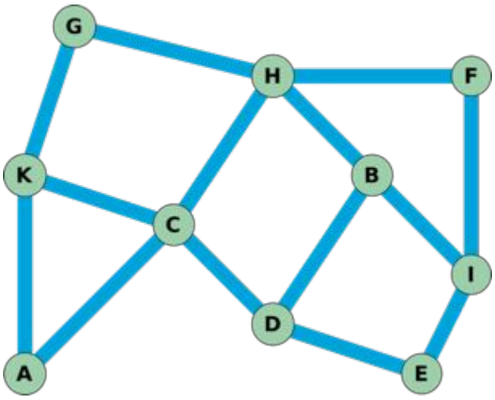
This problem also explores basic set theory. We are looking for an element in the set which is not in the set used in Experiment 1, which means it is the complement of A, B, C. We then look at the intersection of Experiments 2 and 3 in order to determine the common element in both.

Primary Health Care

Years 3+4		Years 7+8	B
Years 5+ 6	C	Years 9+10	A
		Years 11+12	

Doctor Hamid wants to build three hospitals for the beavers.
The hospitals can only be built on the places shown on the map below.
To get to a hospital, the beavers should not have to swim through more than one stream from any of these places.

Question:
Choose three places to build the hospitals for Doctor Hamid.



Answer:

There are several correct solutions, one for instance uses the places E, H and K:

- For the places D, E and I the beavers can swim to E.
- For the places B, C, F, G and H the beavers can swim to H.
- For the places A, C, G and K the beavers can swim to K.

The other solutions are: A E H, C G I, C H I, C I K, D F K, B I K and C E H.

Explanation:

The solutions can be found by placing a station at a random position and marking all stations that are reachable within one step. Then you can position the next station and so on. Once all three stations are placed there are two possibilities: either it's a solution or there are one or more places that are not marked. If it's not a solution, you can remove the last station you've placed and place it in another place and check again. If you are still not lucky to find a solution with 3 stations you have to "backtrack" and place the last station on another place. By doing this systematically one can find all possible solutions.

It's Computational Thinking:

Abstraction
Data interpretation – patterns and models
Impacts - sustainablity

In Computing this channel system is generalised to the concept of a graph consisting of vertices (intersections) and edges (water canals). The more general problem is to find a so-called "vertex cover" in a graph. This is a subset of the vertices that cover the whole graph. Whenever all the neighbour vertices to this subset are added together, they will cover all vertices of the graph. Usually a minimal number of such vertices is the most cost efficient. In more complex graphs it is very hard to find these cost-effective subsets of the vertices. It needs a computer algorithm to find a solution.

The method of placing the stations described in the explanation section is called backtracking. You try one solution and if it is not correct you take back the last step you've made and try another step, ideally systematically until you have exhausted all possible last steps. Then you take back the pre-last step, try all solutions and so on until you have found a correct solution. This method is not very efficient, but for this kind of problems it works reliably.

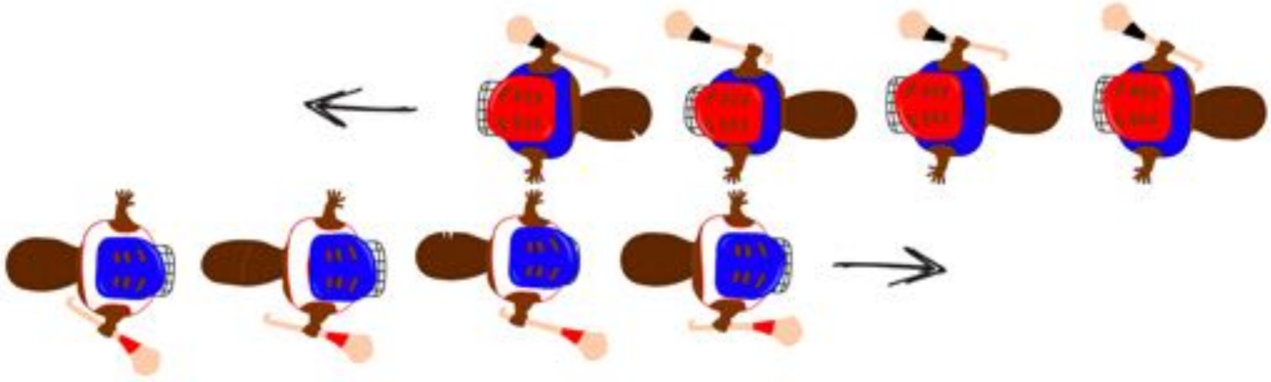
Hurlers Shake Hands

Years 3+4		Years 7+8	B
Years 5+ 6	C	Years 9+10	A
		Years 11+12	

Beavers enjoy playing hurling, a popular game similar to hockey.

After the game ends, the beavers in each of the two teams line up in a row and walk past the other team. As they pass each other, they shake hands. At the beginning, only the first player on each team shakes hands. Next, the first two players shake hands (see picture below). This continues until each player has shaken hands with every player on the other team.

There are 15 players on each team.



Question:

If each player takes one second to shake hands and move to the next player, how many seconds of shaking hands will there be?

Answer:

29

Explanation:

The amount of handshaking is exactly the length of one line plus the length of the other line, minus one.

Let us imagine that there is only 1 player on each team. After 1 second, all handshaking has finished. Let us imagine that there are only 2 players on each team. During the first second, the first player on each team shakes hands. During the second second, the first player on each team is shaking hands with the second player on the other team, and during the third second, the second two players are shaking hands with each other. So, that's three seconds.

With 15 players in each team, the number of seconds required is 15 + 15 – 1 = 29.

It's Computational Thinking:

Algorithms – following and describing
Digital systems
Specification – techniques
Interactions – data and processes
Impacts - sustainability

This task can be viewed as an illustration of a parallel processing paradigm called pipeline processing. Pipeline processing is a very efficient way to get many computers working together to solve problems quickly, but it can take a relatively long time to reach that efficiency, just like our players at the back of each line had to wait quite a while before their first handshake.

Analysing the running time of an algorithm is a sophisticated part of computer science called computational complexity analysis. In this Task, we know the team size is fixed at 15, and can deduce that the "running time" of the hand shaking algorithm is 29 seconds. However, in computational complexity we would be asked to measure the running time independent of a specific team size. We would conclude that the hand shaking algorithm takes 2N-1 seconds, for any team size N, where N is 1, or any larger natural number.

Concurrent Directions

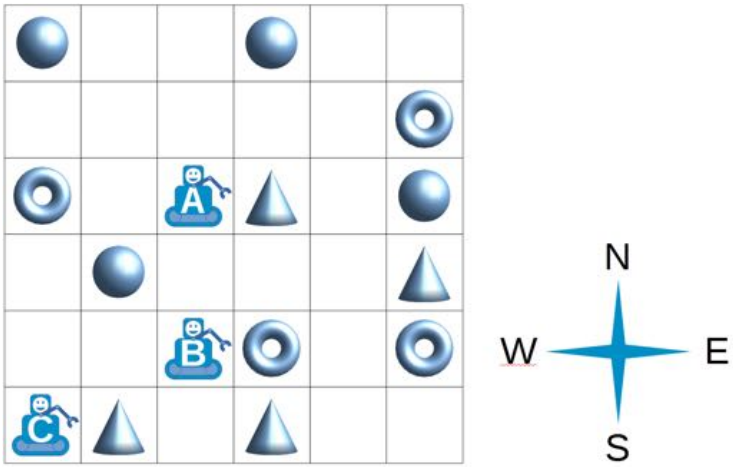
Years 3+4		Years 7+8	B
Years 5+ 6	C	Years 9+10	A
		Years 11+12	

In a warehouse, three robots always work as a team.

When the team gets a direction instruction (N, S, E, W), all robots in the grid will move one square in that direction at the same time.

After following a list of instructions, the robots all pick up the object found in their final square.

For example, if we give the list N, N, S, S, E to the team, then robot A will pick up a cone, robot B will pick up a ring, and robot C will pick up a cone.



Question:
Which list of instructions can be sent to the robots so that the team picks up exactly a sphere, a cone, and a ring?

A. N, E, E, E
B. N, E, E, S, E
C. N, N, S, E, N
D. N, E, E, S, W

Answer:

B. N, E, E, S, E

Explanation:

A. If the team's list is N, E, E, E then robot A will pick up a ring, robot B will pick up a cone, and robot C will pick up a ring. No sphere is picked up, so this is an incorrect answer.

B. If the team's list is N, E, E, S, E then robot A will pick up a sphere, robot B will pick up a ring, and robot C will pick up a cone. There is one of each type of object, so this is the correct answer.

C. If the team's list is N, N, S, E, N then robot A will pick up a sphere, robot B will pick up a cone, and robot C will pick up a sphere. No ring is picked up, so this is an incorrect answer.

D. If the team's list is N, E, E, S, W then robot A will pick up a cone, robot B will pick up a ring, and robot C will pick up a cone. No sphere is picked up, so this is an incorrect answer.

It's Computational Thinking:

Algorithms – following
Digital systems
Specification – techniques
Interactions – data and processes
Impacts – sustainability

When robots or computers work together at the same time, we say that they are working in parallel.

If we have a small number of robots or computers working in parallel, we could give each of them a different list of instructions. However, if we have thousands or millions of computers working together then it is not practical to write separate instructions for each. We have to give large groups of them the exact same instructions. For example, the Tianhe-2 supercomputer has 3 million separate computing cores that can be used to work together on solving a single complicated problem.

In this task, we have the potential for an additional complication. The robots are working in the same space, and their instructions must be prepared more carefully to ensure that they do not crash into each other or otherwise block one another. Preparing parallel instructions in such a situation is an extremely challenging aspect of computer science called concurrent programming.

Bus Stop

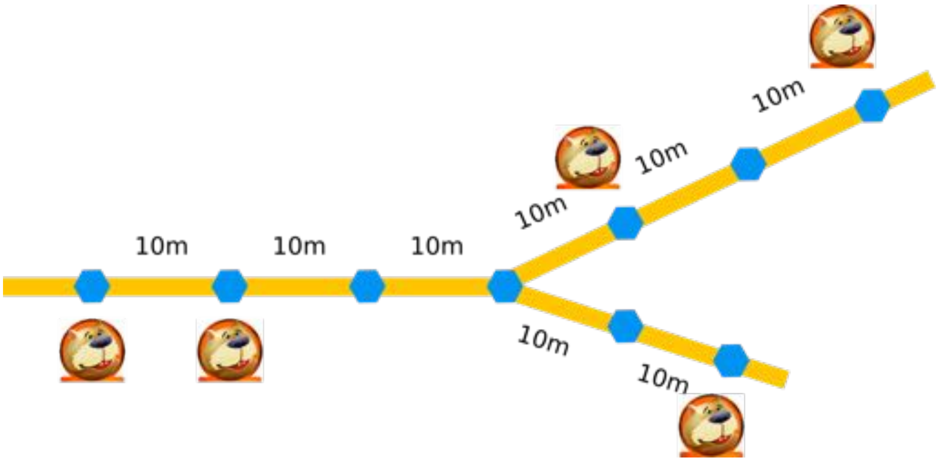
Years 3+4		Years 7+8	A
Years 5+ 6	C	Years 9+10	
		Years 11+12	

The lodges of five beavers are shown on the map below.

The Beavers want to put a bus stop in one of the places marked by blue hexagons.

All the hexagons are 10m apart.

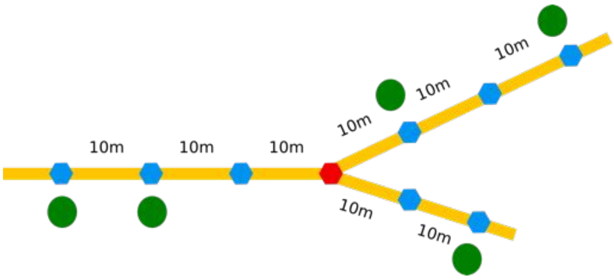
The beavers decide that the sum of the distances from their lodges to the bus stop must be as small as possible.



Question:

Click on the best place for the bus stop.

Answer:



Explanation:

One naive way to see this answer is to try all 9 possible locations for the bus stop and calculate the sum of the distances. This requires a lot of calculation. To reduce the amount of calculation, we can create a model or formula for our solution.

Suppose we located the bus stop to be at the branching of the roads (where all three roads intersect). The sum of all the distances from the lodges to the bus stop:

$30 + 20 + 10 + 30 + 20 = 110$

When the bus stop is moved x meters to the left, the first two distances will be reduced by x metres, and the distance from the last three lodges to the bus stop the will increase by x metres:

$(30-x) + (20-x) + (x + 10) + (30 + x) + (x + 20) = 110 +x$

We will get the same result if we choose a location for the bus stop at a distance of x meters to the right on the upper right route:

$(30 + x) + (20 + x) + (x +10) + (30 -x) + (-x + 20) = 110 +x$

If you choose the location for the bus stop at a distance of x meters to the right on the lower right route, the result will be:

$(30 + x) + (x + 20) + (10 + x) + (x + 30) + (20 - x) = 110 + 3x$

It's Computational Thinking:

Data representation - symbolism
Breaking down problems into parts (Decomposition)
Data interpretation – patterns and models
Algorithms – following and describing
Impacts – sustainability

This is a classic problem of tree centre finding. It is also an optimisation problem. In computer science, many problems involve finding the best, or least cost, or minimal or maximal value to some function in order to save the most money, use the least amount of resources, take the least amount of time, etc. This problem is a simple case of a real-world problem involving transportation networks: City planners need to make these sorts of decisions for subway, bus or train locations in order to optimise the use of those systems. In computer science, we apply various algorithms to help find the optimal answer. For this problem, we can use one of three possible algorithms:

1) An exhaustive search of all options (for a small number of vertices). For larger networks (if there were thousands of locations) this is not feasible.

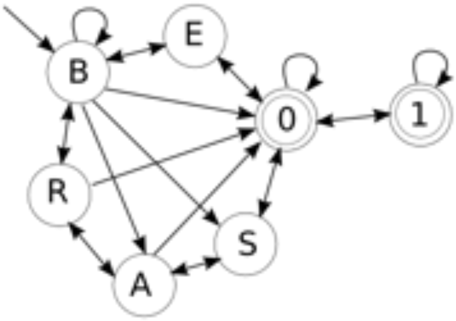
2) If the locations formed one row, we can simply compute the median location, and it will be optimal.

3) The optimal node has the property that, if we looked down each path from each of its neighbours, there is less than the half of the beavers living there. To optimise this criterion, we can use a dynamic-programming solution to find this answer in an efficient time.

Rafting

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

Beavers build rafts. For river traffic control, all rafts should be registered.
This means that each raft should have a license plate with unique text.
The text is made up of letters and digits as shown in the diagram below.
The licence must start with the letter B and end with the digit 0 or 1.



Question:

Which two of these license plates cannot be registered?

BB0001 BBB100 BBB011 BB0100 BR00A0 BSA001 BE0S01

Answer:

BBB100 and BR00A0

Explanation:

The best way to solve this is simply follow the diagram and check the solutions one by one. BBB100 is incorrect, because the digit part starts with 1 (you can't get from the B to the 1) and BR00A0 is incorrect because you can't get from 0 to A as it is a one-way arrow.

It's Computational Thinking:

Data representation - symbolism
Breaking down problems into parts (Decomposition)
Data interpretation – patterns and models
Digital systems

Finite automata are an important part of theoretical computer science. Computers often read the sequence of characters and words in a document or a computer program with the help of finite automata. Finite automation can look at the sequence of instructions, recognise the patterns and accept or reject input accordingly. If the characters present are an allowed combination, a known word for example, it will accept them. Otherwise it will not.

Segway

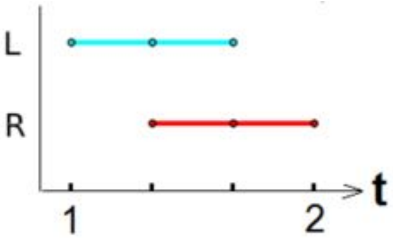
Years 3+4	Years 7+8	C
Years 5+ 6	Years 9+10	B
	Years 11+12	

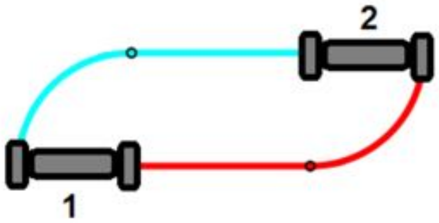
Jan has a special vehicle that looks like a Segway. He moves it by pressing two buttons: a blue (light) button on the left, and a red (dark) button on the right.

When he presses a button, the wheel on that side of the vehicle rotates. If both buttons are pushed at the same time, both wheels rotate and the vehicle moves forward.

If he pushes a single button, only one wheel rotates and the vehicle turns.

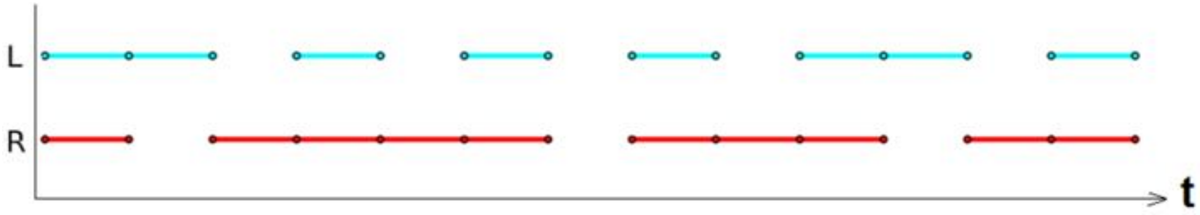
Example:
The follow tables shows which button was pushed when, and how the vehicle moved from location 1 to location 2.





First, the blue button was pressed and the vehicle turned to the right. Then both buttons were pressed, and the vehicle moved forward. Finally, the red button was pressed, and the vehicle turned left. The orientation of the vehicle is now the same as in the beginning: facing towards the upper wall.

Question:
Here is a record of the button presses from a different journey:



The vehicle kept going until it hit one of the walls. At the start the vehicle was facing towards the upper wall.

Towards which wall was the vehicle facing in the end?

upper lower left or right

Answer:

lower

Explanation:

The left button was pressed 8 times during the ride, while the right button was pressed 10 times. That means the right button was pressed two times more and the vehicle turned left twice, so will face the opposite direction from where it started - it must hit the lower wall.

It's Computational Thinking:

Abstraction
Algorithms - following
Data interpretation – patterns and models
Digital systems

Such vehicles (e.g. robotic removers of unexploded devices) can be remotely controlled by two signals for two motors. The computer can generate the signals for automatic crossing of a troubled terrain according to a program and knowledge of the terrain.

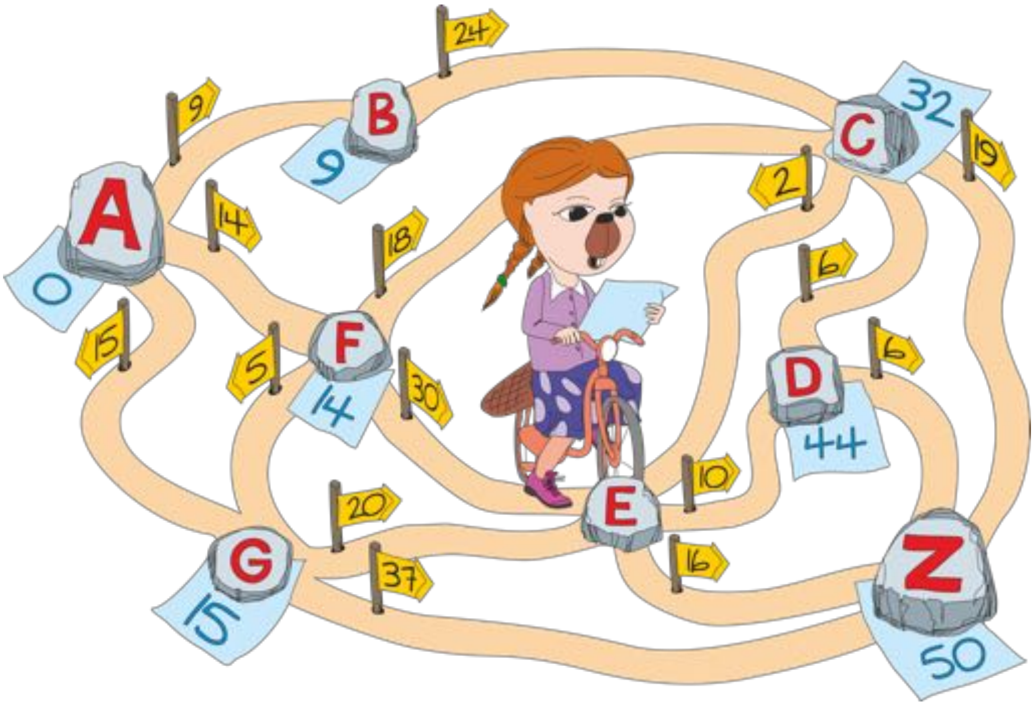
This vehicle can be considered as a finite-state machine or automaton that can be in one of a finite number of states. The machine is in only one state at a time.

A graphical representation of a situation is often clearer than its written description, and can help and be less error prone when one has to devise the operations to be performed.

Bike Paths

Years 3+4		Years 7+8	C
Years 5+ 6		Years 9+10	B
		Years 11+12	A

Cleveria is a beaver biker. She explores the one-way paths that pass through the villages in her district. Each village has a village stone labeled with a single letter. All the paths have a distance and a direction. The distance and direction are given by the yellow flags.



Over the course of many different trips Cleveria leaves blue notes with a number on under a stone in each village. The notes are about the distance from village A to the village stone with the note under.

Question:

What is the meaning of the numbers she has left under the stones?

A. The shortest distance going through the least number of villages

B. The shortest distance to this village

C. The shortest distance to this village by taking a left turn at crossings if possible

D. The shortest distance to this village by taking a right turn at crossings if possible

Answer:

B. The shortest distance to this village

Explanation:

In order to find the correct answer, the distances for each village according to the different specifications A through D have to be computed:
A) is wrong because otherwise D = 45, Z = 52;
C) is wrong because otherwise C = 33, D = 45, Z = 52;
D) is wrong because otherwise C = 51, D = 45, Z = 52.

So the blue number shows the length of the shortest route from A to a particular village (B).

To find the shortest route we can use the Dijkstra's algorithm:

1. Assign a tentative distance value to every crossing by setting it to zero for A (our initial crossing) and to infinity for all other crossings.
2. Set A as the current crossing.
3. For the current crossing, consider all of its unvisited neighbours and calculate their tentative distances. For each neighbour, compare the newly calculated tentative distance with the current assigned value and assign to it the smaller one.
4. When we are done considering all of the neighbours of the current crossing, mark the current crossing as visited. A visited crossing will never be checked again.
5. Select the unvisited crossing that is marked with the smallest tentative distance, set it as the new "current crossing", and go back to step 3.
6. If the destination crossing (Z) has been marked visited, then stop. The algorithm has finished.

It's Computational Thinking:

Data interpretation – patterns and models
Algorithms – following
Interactions – data and processes
Impacts – sustainability

The problem of finding the shortest route is called "shortest path problem" and it is one of the fundamental computing tasks in everyday applications. Shortest path algorithms are applied to automatically find directions between locations, such as driving directions on websites like MapQuest or Google Maps. Dijkstra's algorithm is one of the most popular algorithms used to find the shortest route. Dijkstra's algorithm will assign some initial distance values and will try to improve them step by step. The structure used to represent places and routes connecting them is called a graph, a very important data structure in Computer Science.

Cave Game

Years 3+4	Years 7+8	C
Years 5+ 6	Years 9+10	B
	Years 11+12	A

Hale and Serge are playing a game:

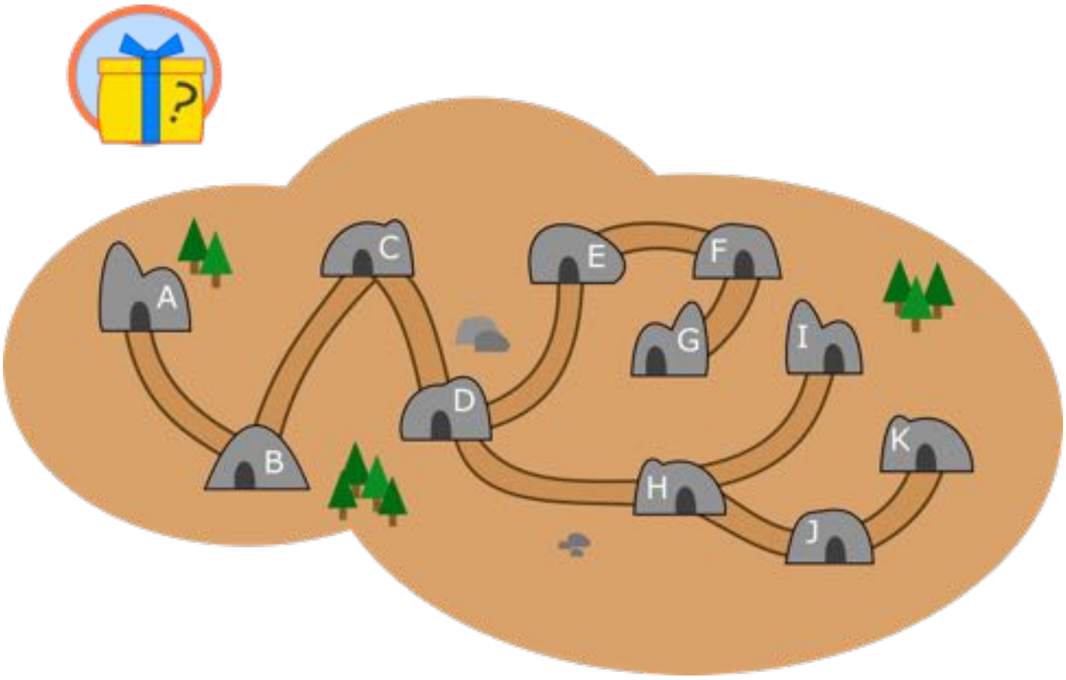
Hale hides a present in one of several caves.
Serge has to find which cave it is in.

To do so, Serge has the map shown below and is only allowed to ask questions like:
“Is the toy in cave X?”

If Serge guesses correctly, Hale will answer “yes”.

Otherwise, she will tell Serge which of the neighbouring caves leads to the hidden toy.

When Serge knows for sure where the toy is, the game is over and he will walk to the cave.



Question

Serge wants to ask as few questions as possible to find the present.
In the worst case, how many questions does he have to ask to be sure to have found the present?

Answer:

3

Explanation:

We first show that wherever the toy is, Serge can win in three questions. Here is an example of the questions:

First question: Is it d?
There are then 4 possible answers, and for each of them we can conclude in at most 2 steps:

Yes Then he knows where the toy is: total of 1 question
c Then he asks b and he knows: total of 2 questions
e Then he asks f and he knows: total of 2 questions
h Now he can ask h. Either it is the right cave or the answer is i: total of 2 questions, or the answer he gets is j, and by asking j he finds the toy: total of 3 questions.

Now we show that with fewer than 3 questions, there are cases where we cannot be sure of the cave hiding the toy. Of course, Serge cannot guess with no questions, or with just one. Now suppose Serge only asks two questions. Asking d first is optimal: if the answer is a, b, c, d, e, f or g then in a total of two questions he knows. But if Hale answers h to this question, then the one remaining question is not enough to know if the toy is in h, i, j, or k with certainty. Picking any other cave other than d on the first guess will also lead to at least 3 questions being required.

It's Computational Thinking:

Data interpretation – patterns and models
Algorithms – following and describing
Interactions – data and processes
Impacts – sustainability

This is a question about searching in a tree. A lot of information is stored in tree-like structures. The design and use of proper algorithms to retrieve this information is an important subject in computer science.

In this problem, the caves are the nodes of the tree, and the nodes that have at least three paths connecting them are of particular importance: They decompose the tree into "branches", and if we know which branch to take to find what we look for, it enables us to search only that branch and not the rest of the tree. This is why the structure of the tree and the way data is put inside is very important: organising data makes it possible to choose one branch among all of the others, and having a lot of branching points means that only a small portion of the tree needs to be visited. Indeed, if there was only one branch (ie the tree is a path), then we would have to explore it entirely until finding what we want. On the other hand, if the network of the caves made a "star" (one cave at the middle), then one question would be enough in all cases: Ask for the central cave.

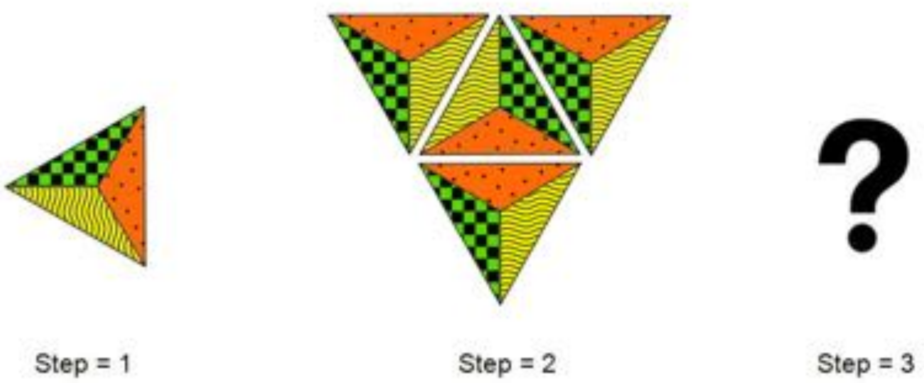
Triangles

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

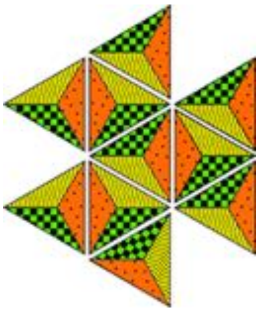
A beaver wants to create a mosaic with identical, triangle-shaped tiles.

He starts with one tile. He rotates it 90 degrees clockwise and then adds tiles on each side of the triangle-shaped tile, as shown in the picture below.

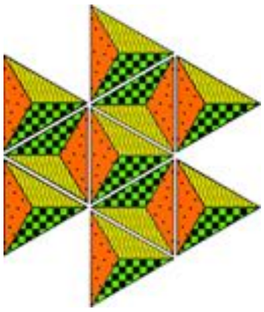
Then he rotates the whole shape 90 degrees clockwise again and adds tiles to the sides as before.



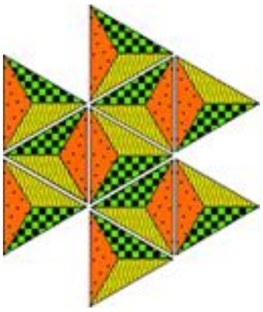
Question:
What will be the final shape of the triangles after step 3?



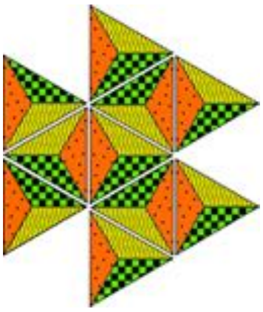
a



b

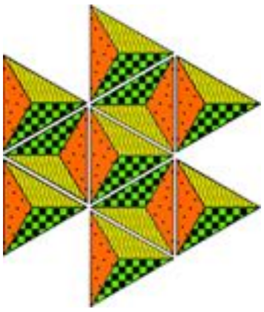


c



d

Answer:



Explanation:

Answer a is incorrect because the tiles are not rotated 90 degrees clockwise.
Answers c and d are incorrect because the tiles do not match on their adjacent sides.

It's Computational Thinking:

Data interpretation – patterns and models
Algorithms – following and describing

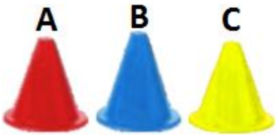
This task involves executing a sequence of instructions iteratively. Executing algorithms is a key practice in computer science. This task also demonstrates various computational thinking skills: algorithmic thinking is evident in the execution of the algorithm. Evaluation takes place as students consider the various solutions against the first two elements in the sequence. The task also gives an opportunity to demonstrate generalisation in terms of recognising patterns in the potential answers.

Cards and Cones

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

Inés has a pack of cards; each card has a number written on it from 1 to 9. The pack contains many of the same cards.

She places three coloured cones in front of her:



Inés intends to create stacks under the cones with the numbers facing up.
Each time she puts a new card on the stack it will cover the rest of the stack.

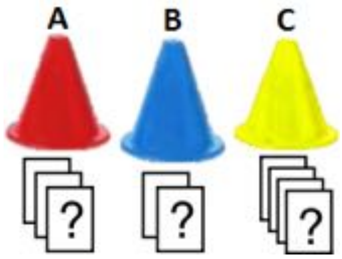
Her friend, Jules, takes notes as Inés puts cards, one at a time, under the cones:

Inés starts by placing a card with the number 5 on it under the red cone. Jules writes: **A <-- 5**
Next Inés places another card under the red cone, on top of the previous one. Jules writes: **A <-- 3**
Then Inés peeps under the red cone and finds a card from the pack that has the same number as she sees. She places it under the blue cone. Jules writes **B <-- A**

Jules' final notes look like this:

A <-- 5
A <-- 3
B <-- A
B <-- 3
A <-- B
B <-- 5
A <-- 6
C <-- B
A <-- B
B <-- 1

Question:
What cards are visible when the cones are lifted?
Write the correct numbers in the spaces below the question mark.



□ □ □

Answer:

5 1 5

Explanation:

Below is shown the top card in the stacks under the three cones after each of the instructions recorded by Jules.

Notes	A	B	C
A <-- 5	5		
A <-- 3	3		
B <-- A		3	
B <-- 3		3	
A <-- B	3		
B <-- 5		5	
A <-- 6	6		
C <-- B			5
A <-- B	5		
B <-- 1		1	

It's Computational Thinking:

Abstraction
Data interpretation – patterns and models
Algorithms – following and describing
Specifications - techniques
Interactions – data and processes

This question relates to how an actual program works. Think of the cones as representing a variable. The variable stores a value, in this case an integer. As the program progresses, values in the cones are replaced and exchanged. These cones also function like a stack. The values on the top of the pile under a cone are constantly being replaced and when we check what is the visible card under a cone, it is the last one placed. This means that the last card placed under the cone is actually the first value read making these cones an analogy of stacks in computer science.

Further, values can be copied from one 'cone' to another. In programming this is called passing by value. In fact, some languages support copying by reference, that is, instead of copying the integer, we put a reference inside a cone that tells the cone its value is the same as the one in the blue cone. Thus, if we change the value of the blue cone, we simultaneously change the value in the red cone!

Red & Blue Marbles

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

Beaver Emil is trying a new puzzle on his computer. He has to arrange a stack of marbles in a cylinder.

Rules:

The marbles must be either red or blue.

There must be at least three marbles in the cylinder at the start.

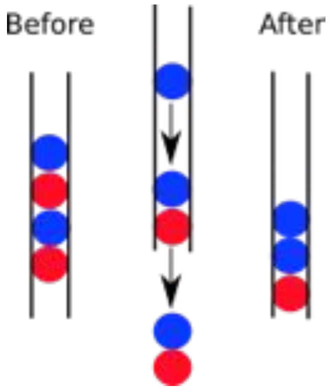
Aim:

To produce a stack that never has less than 3 marbles in the cylinder when the GO button is repeatedly pressed.

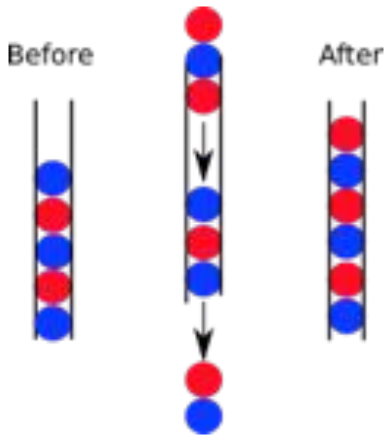
What happens when GO button is pressed once:

Each click of the GO button lets the two lowest marbles drop out.

Then one of two things happen depending on the colour of the first marble to drop out:



If the first marble that drops is blue: three new marbles drop on the top of the cylinder: one red, one blue, and one red.



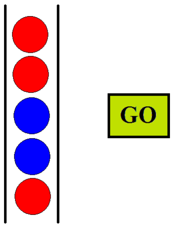
If at least three marbles remain in the cylinder after each press of the GO button, Emil will click the button again.

The game ends if two marbles or less remain in the cylinder.

Example:

The stack shown on the right produces a game that ends after five clicks.

At this point, only two blue marbles will remain in the cylinder.



Question?

Produce a starting stack that consists of only three marbles which will produce a never-ending game.

Answer:
Any stack of thee marbles where the bottom one is blue.

Explanation:

The game does not end if Emil begins from a stack of three marbles that has a blue marble at the bottom.
Let us note that, when the starting stack consists of three marbles and the bottom marble is red, one click will end the game; but when the starting stack consists of three marbles and the bottom marble is blue, after at most five clicks the cylinder will contain the stack RBRRBR (see the figure below). Indeed, listing the colours from the top to the bottom of the stack, you will get:

BBB → RBRB → RBRRB → RBRRBR;

BRB → RBRB → RBRRB → RBRRBR;

RBB → RBRR → BRB → RBRB → RBRRB → RBRRBR;

RRB → RBRR → BRB → RBRB → RBRRB → RBRRBR.

Thus, the game falls into a four-cycle (because after four clicks the same configuration RBRRBR will recur):

RBRRBR → BRRBR → BRRB → RBRRB → RBRRBR.

It's Computational Thinking:
Breaking down problems into parts (Decomposition)
Data interpretation – patterns and models
Algorithms – following and describing
Digital Systems

The puzzle represents a Post production system, a computational model using string rewriting, developed by Emil Leon Post in the 1920s but first published in 1943. Emil Leon Post (1897-1954) was a Polish-born American mathematician and logician who helped to increase our knowledge in computability theory.

Rewriting models comprise various systems of formulas that allow replacing one substring with another one. One rewriting model can be seen as a system of objects (finite), with the set (finite) of relations defining possible manipulations and transformations of those objects.

Theoretical Computer Science nowadays refers to these systems as context-free grammars. For example, a system of multiplications and additions can be defined using a simple context free grammar (CFG) with just a few rules. Another famous and useful example is the use of an appropriate context free grammar to precisely and completely define a programming language, both for educational purposes and for its implementations in computing devices.

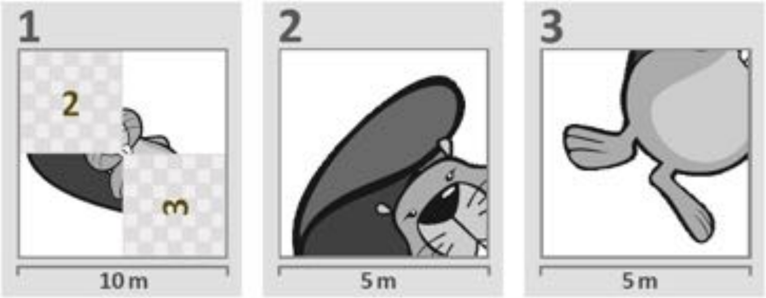
Years 3+4	Years 7+8		
Years 5+ 6	Years 9+10	C	
	Years 11+12	B	

Recursive Painting

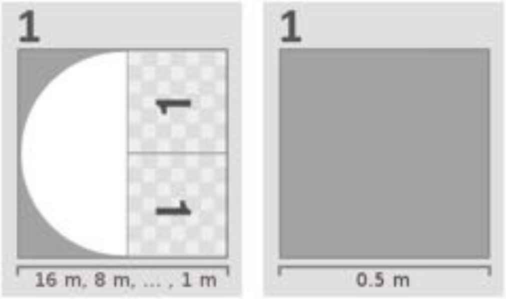
Ingrid and her friends have volunteered to help paint a floor that is 16m long by 16m wide.

The instructions are printed on numbered sheets that refer to the other sheets by their number. Each sheet has a scale printed at the bottom.

Here is an example floor plan from a previous project. It draws a beaver.



Ingrid is given the plan for a new project:

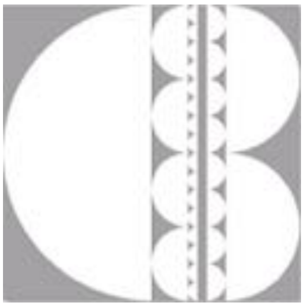


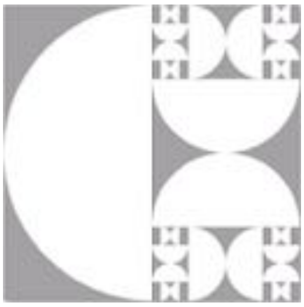
The planning sheet refers to itself and both sheets have the same number!

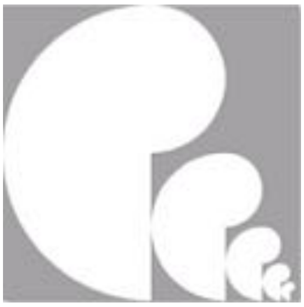
Ingrid's friend asks how this can be and she answers: "We can do it. The second sheet is important because it tells us when to stop."

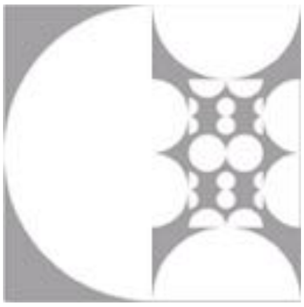
Question:

What does the painted result look like?

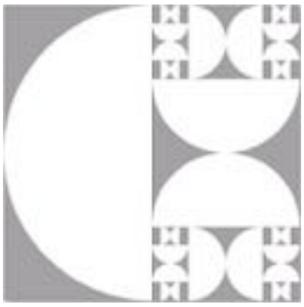








Answer:



Explanation:

Consider the left sheet of the plan. It describes how to paint the left part of the floor with a half circle with its round side facing to the left. For the right half of the floor, it describes how to use the same sheet again, twice, but the part that has to be painted has to be at least 1m long. Do note that the orientation of the two 1's is opposite. This indicates the sheet has to be rotated each time to the proper orientation.

Note that the two round sides of sheet "1" touch each other, this means that in the answer the round sides of the half circles will always touch.

It's Computational Thinking:

Breaking down problems into parts (Decomposition)
Data interpretation – patterns and models
Algorithms – following and describing
Digital Systems
Interactions – data and processes
Impacts – sustainability

Self referencing instructions like this one are called recursive. Recursion is an important concept in Computer Science. Recursive solutions are usually shorter and more compact than their alternatives but sometimes a little bit harder to understand. Recursive patterns are frequently found in nature. Terminating conditions like the one shown in the example play an important role in recursion to avoid endless loops.

Scanner Code

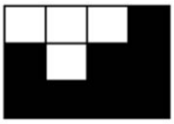
Years 3+4		Years 7+8	
Years 5+ 6		Years 9+10	B
		Years 11+12	B

Two scanners encode an image by translating its pixels into a special code. The code lists the number of all consecutive pixels of the same colour (black or white), followed by the number of all the consecutive pixels of the other colour, and so on. Both scanners start from the top left corner, and go from left to right, and row by row.

The two scanners use different methods at the end of a row:
Scanner A processes the pixels row by row and restarts the encoding on the next row.
Scanner B processes the pixels row by row but does not restart the encoding on the next row.

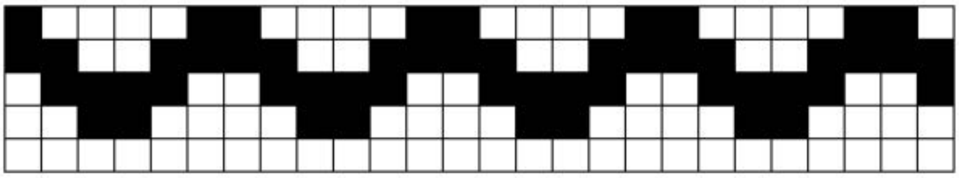
Example:
The image on the right would be represented by the following codes:

Scanner A: 3,1,1,1,2,4 (3 white, 1 black, 1 black; 1 white, 2 black, 4 black)
Scanner B: 3,2,1,6. (3 white, 2 black, 1 white, 6 black)

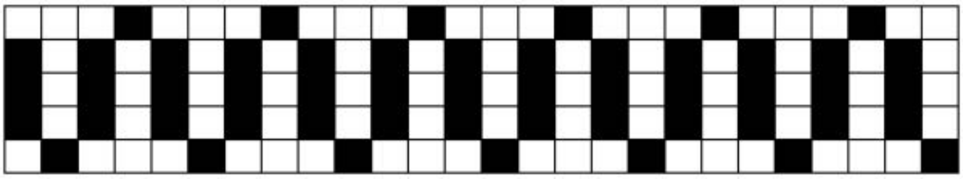


Question:
Which of the following pictures will have the same code no matter which scanner is used?

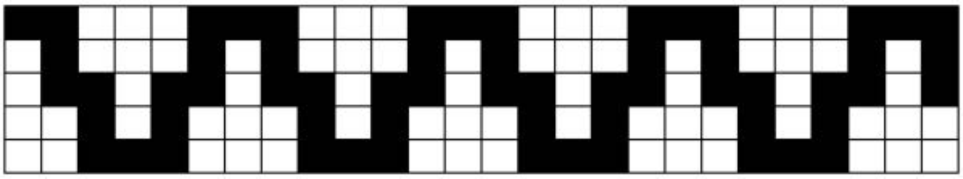
A



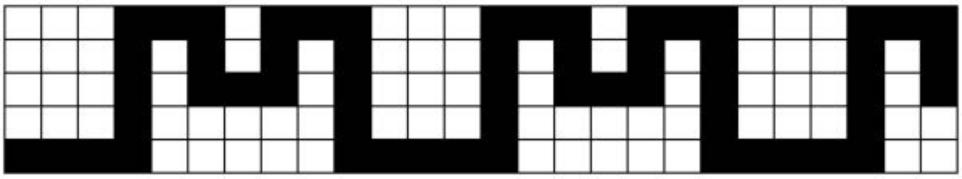
B



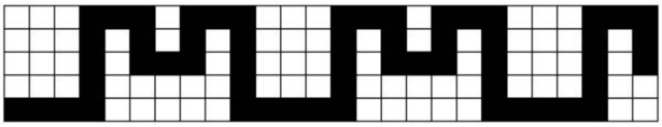
C



D

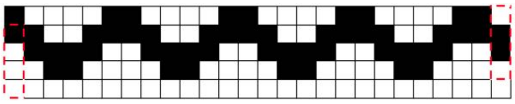


Answer:

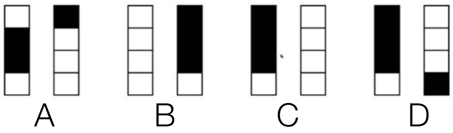


Explanation:

The difference in the 2 methods is in whether the last pixel(s) in a row and the first pixel(s) in the next row are combined or not. Scanner A does not combine them. Scanner B combines them only if they are of the same colour. If they are not the same colour, the resulting code from the two scanners will be the same. Therefore, we must find a picture in which the last pixel in a row and the first pixel in the next row are different, for all rows:



We only need to compare the 4 pixels at the end of each row and the corresponding 4 pixels starting each row, in each of the four supplied images, to make sure that they are of the opposite colours:



Of the 4 pictures, the only one that meets this requirement is picture D.

It's Computational Thinking:

Abstraction
Data representation - symbolism
Data interpretation – patterns and models
Digital systems

A scanner is a device that optically reads (or scans) an image and converts it into a digital image. When scanning, the colour and brightness of each tiny area (pixel) seen by a sensor is measured and recorded as a numeric value. This process is referred to as digitising the image.

A pixel is a computer word formed from PICture ELement, because a pixel is the smallest element of a digital image. Each pixel is a sample of an original image, and more samples will provide more accurate representations of the original image.

Scanner A uses line breaks to restart its encoding on subsequent rows whereas Scanner B reads the pixels as one long continuous image. Each can have its own set of advantages when being used in practice. For example, with a long image, you might use less numbers with Scanner B but would also need to encode the dimensions of your image. It might not be practical to do this with smaller images. These tradeoffs are very important decisions that must be made when working in computer science.

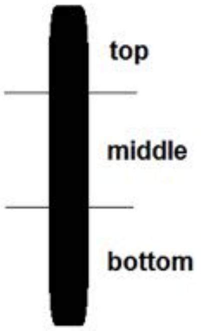
Kix Code

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

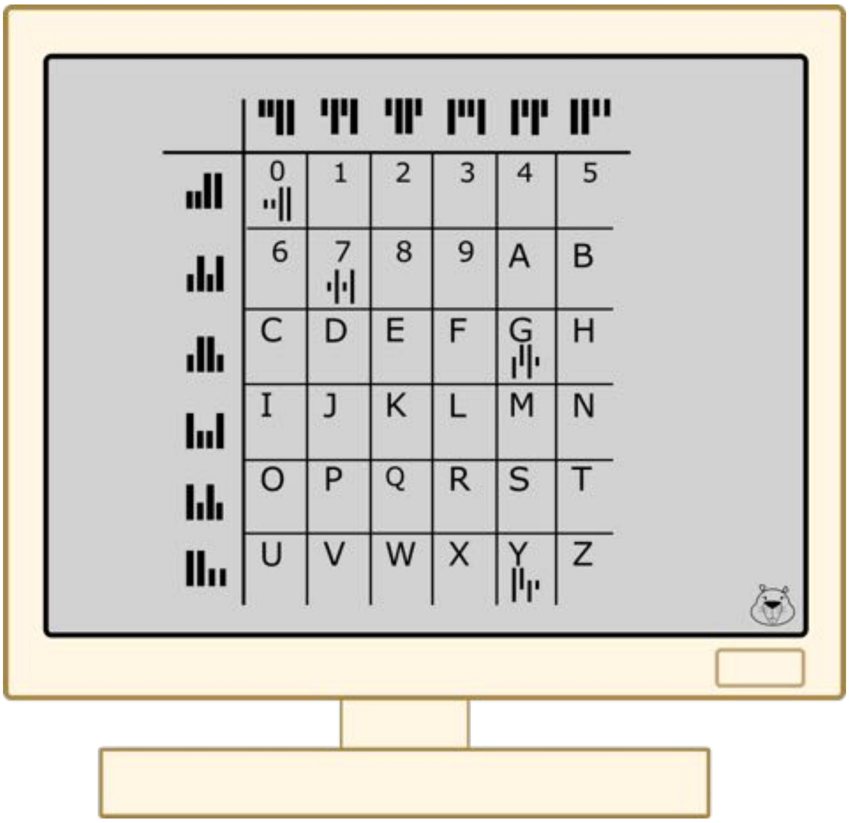
The Bebras Post Office uses postal codes that contain four characters.
To make the postal codes readable by machines, they convert the postal codes into Kix codes.

In a Kix code, each character is represented by 4 vertical bars.

A code has 2 sections: upper and lower.
The upper section contains only the middle and the top bars, while the lower sec
only the middle and the bottom bars.

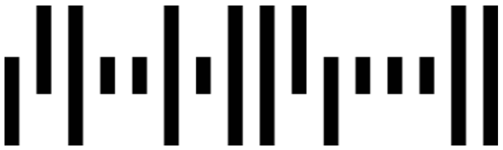


This table shows the codes for several characters:

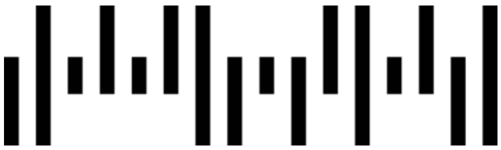


0	1	2	3	4	5
6	7	8	9	A	B
C	D	E	F	G	H
I	J	K	L	M	N
O	P	Q	R	S	T
U	V	W	X	Y	Z

Example:
The Kix code for "G7Y0" is



Question:
Another postal code has this Kix code.



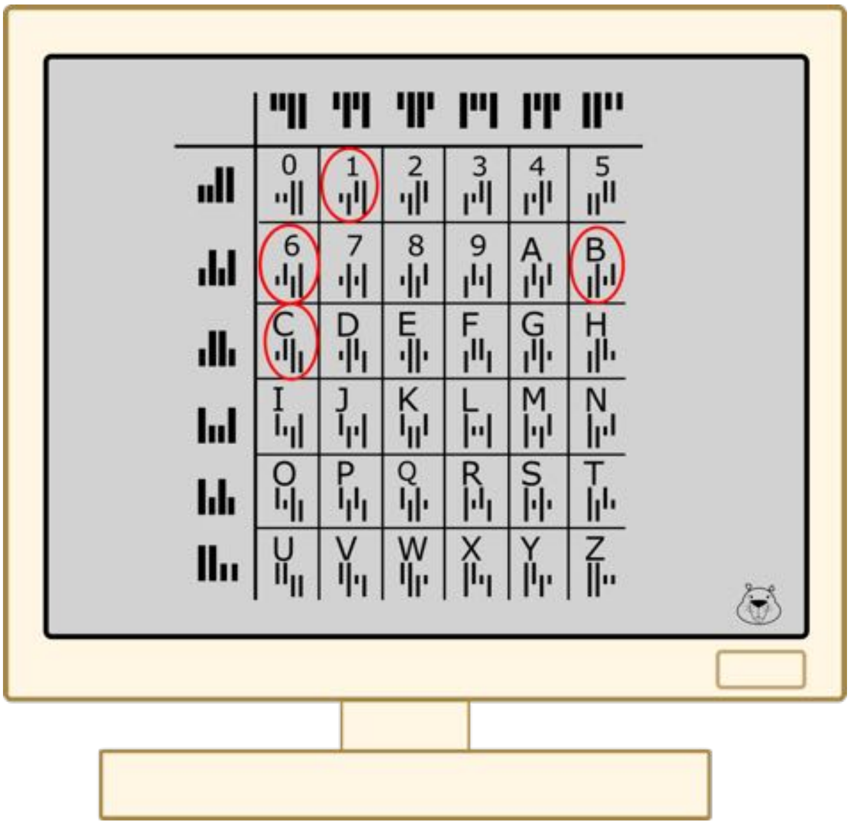
What is the postal code?

Answer:

BC16

Explanation:

The answer can be obtained by looking at the table and converting the Kix code into the corresponding characters.



0	1	2	3	4	5
6	7	8	9	A	B
C	D	E	F	G	H
I	J	K	L	M	N
O	P	Q	R	S	T
U	V	W	X	Y	Z

It's Computational Thinking:

Abstraction
Data representation - symbolism
Data interpretation – patterns and models
Digital systems

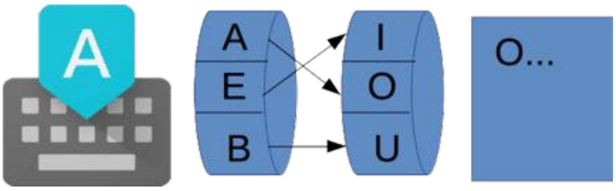
The Kix codes (actually used by the Dutch Post Company) are an example of bar codes. Bar codes are machine readable codes, which are useful in automating the selection and sorting of posts.

In this example, the information in the table is split up between the upper section and the lower section. This technique is commonly used in presenting information.

B-enigma

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

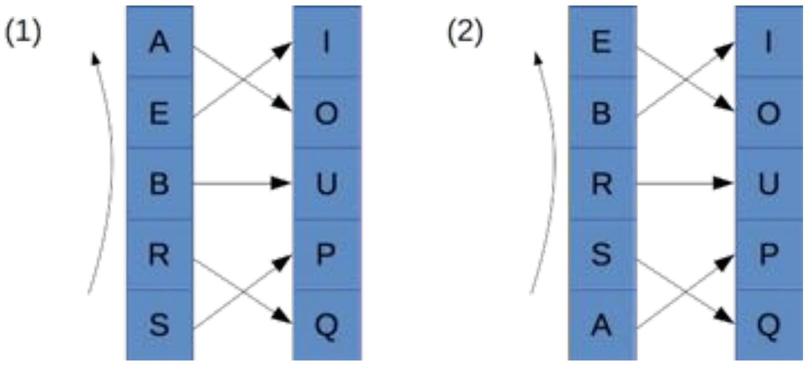
The Beavers need to communicate secretly. They decide to use a mechanism called the B-Enigma machine to hide (encrypt) their messages.



The B-Enigma works as shown above. Each time a letter is typed (e.g. “A”), the left rotor will find a letter on the right rotor according to the arrows (e.g. “O” for “A” in the first step). After typing a letter, the left rotor will move up one position.

This is shown in a different way in the diagram below. After rotating up one position the left rotor will then be in position (2). However, note that the rotor on the right never moves. The links between the two rotors (shown by the straight arrows) also remain the same.

In the diagram below, all the letters available are shown on both rotors.



Question?

The Beavers wish to send the message “BEBRAS”.

What will the encrypted message be if we start from position (1)?

A. UOSAEB

B. UOUQOP

C. UOOOIP

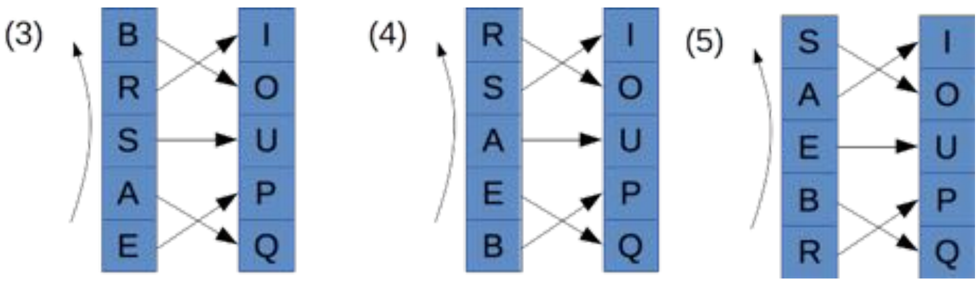
D. UOOUPQ

Answer:

C. UOOOIP

Explanation:

At position (1) letter B is encrypted to U and at position (2) E is encrypted to O as shown in the above image. The other transformations for the letters 3-5 are shown below. For the last letter we use position (1):



At position (3) letter B is encrypted to O, so answers A and B are incorrect.
At position (4) letter R is encrypted to O, so answer D is incorrect.

It's Computational Thinking:

Abstraction
Data representation - symbolism
Data interpretation – patterns and models
Digital systems
Interactions – data and processes
Interactions - people

The machine is a simplified version of the Enigma machine which was used during the Second World War by the German Army to encrypt communications. The Enigma code was first broken by the Polish in 1932. British Intelligence worked throughout the war to decipher the coded messages produced by this machine. Most undergraduate classes on cryptography start with a presentation of the Enigma. The work on breaking the Enigma code by Alan Turing lead to the development of the first computers.

The Game

Years 3+4
Years 5+ 6

Years 7+8
Years 9+10
Years 11+12 C

Beaver Big is playing a game with Beaver Small on the special game board shown.

They start from the leftmost box (Box 5). **Beaver Big goes first.**

She can choose to move Up or Down:
Up will move to Box 4.4; Down will move to Box 5.7.

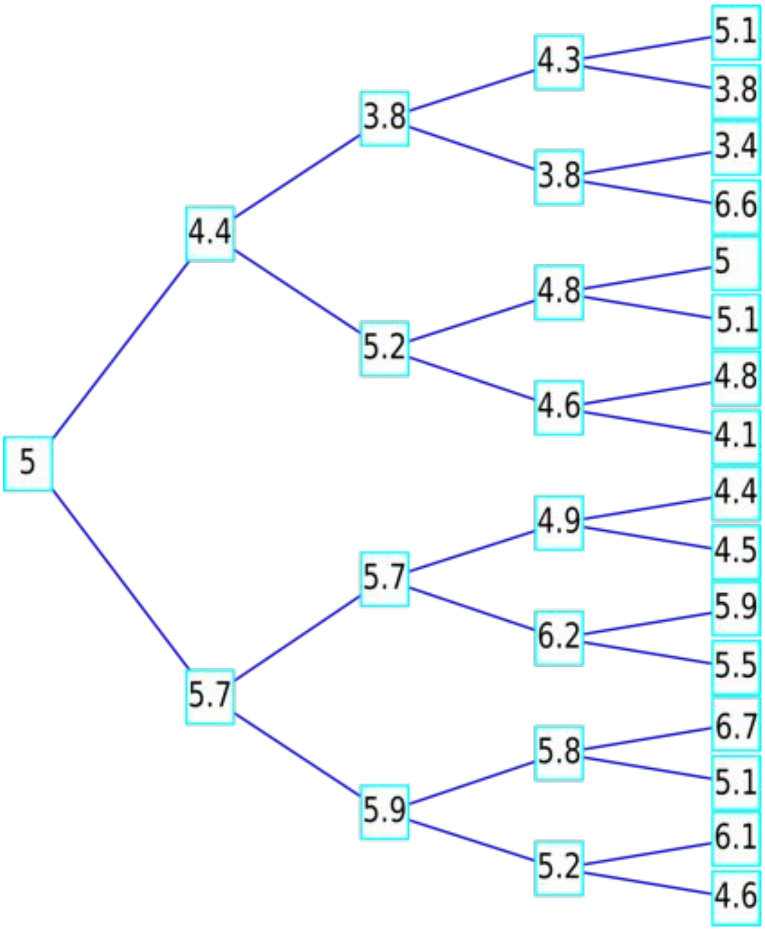
Then, it is Beaver Small's turn to choose Up or Down. From then on they take turns until, finally, Beaver Small chooses a box in the rightmost column.

Because both beavers can see all the numbers on the game board all the time, they are able to plan their moves accordingly.

Question:

Beaver Big plays so that the final box will have the biggest possible value and Beaver Small plays to get the smallest possible value.

If both always play as well as they possible can, what will the number in the final box be?



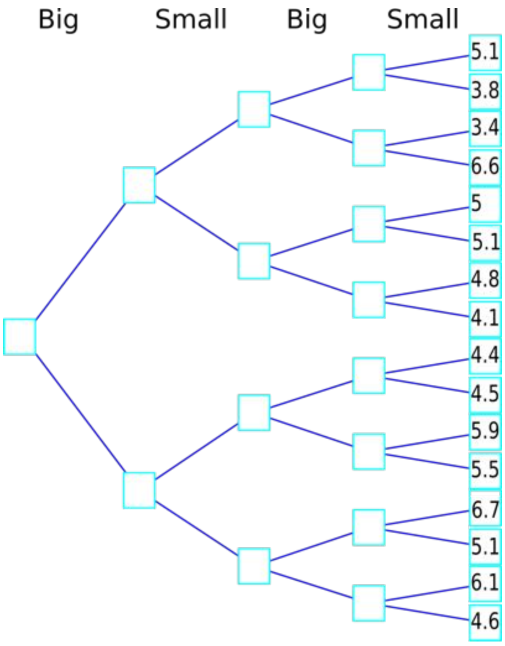
Answer:

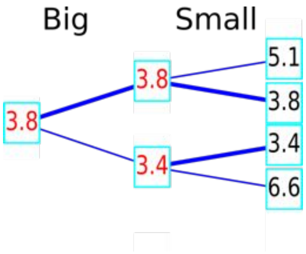
The final position will be the lowest 5.1 box on the right.

Explanation:

To illustrate the right answer, the names of the player at each step are added to the graph.

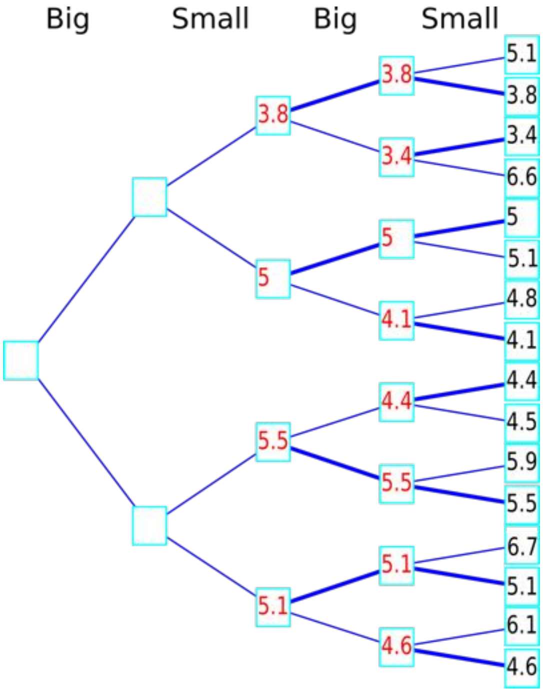
Because we are interested only in the numbers in the rightmost boxes, the other numbers are not needed – they are only placed in the question to confuse us. So all numbers except those laying on the rightmost boxes can be removed.



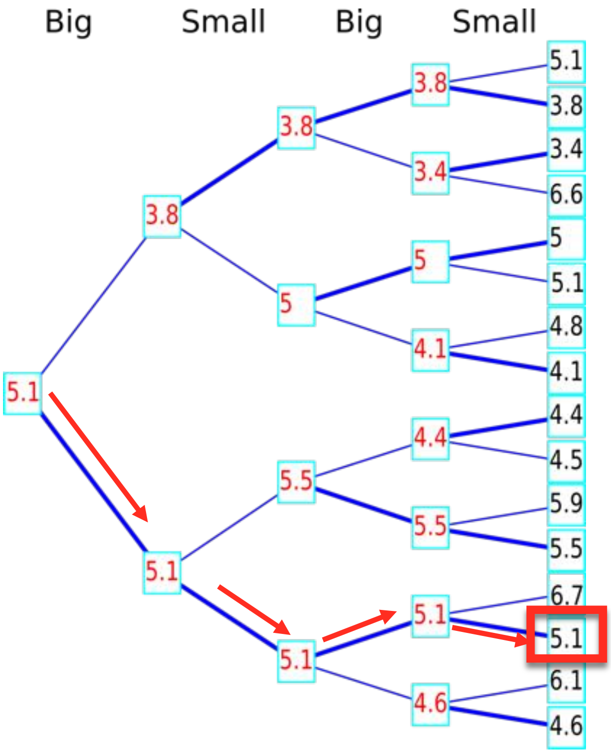


The smartest way to plan the strategy for this game is to start from the final move. In each empty box is written a red number that is chosen by the player as she plays as smart as possible. For a deeper explanation we can use only a part of the graph on the right. Red numbers are those chosen by players in each turn. We can show that there is only 1 number to which the players can arrive.

On the right we can see this process when a few more numbers have been added.



The final solution is found by completing all the boxes in the same way and then tracing the answer 5.1 back through to the correct box:



It's Computational Thinking:

Data interpretation – patterns and models
Algorithms – following and describing

Selecting elements of a binary tree is a very common process in Computer Science.

The graph used in this task is a flipped (right to left) version of the graph often used in sport tournaments, e.g. tennis. In such graphs, the numbers in the rightmost boxes represent 16 players who will compete to become the winner. Starting from the right, in each stage, 2 players compete with each other, and the winning player advances to the left.

Our graph, therefore, could describe a strange tournament in which different rounds have different rules to decide the winner. For example, the winner of the final is the one that has more points, but the winners of the semi-finals are the players with less points.

Selfish Squirrels

Years 3+4
Years 5+ 6

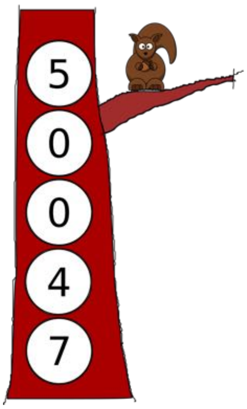
Years 7+8
Years 9+10
Years 11+12 C

16 squirrels live in a tree with five big holes one above the other.

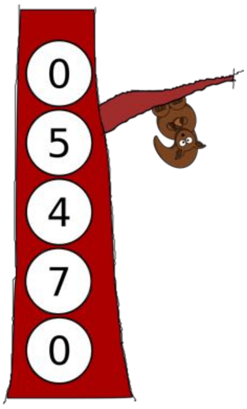
Each day all of the squirrels find out how many squirrels are in their hole and the neighbouring holes above and below it.

The next night, each squirrel secretly either stays where they are or moves to a hole above or below it, whichever currently has the lowest number of squirrels in. If they are the same, the squirrels prefer their current hole to the hole above. They also prefer a high hole to a low hole.

Example:
Today there are 5, 0, 0, 4 and 7 squirrels in the holes from top to bottom as shown below. Tomorrow all 5 squirrels in the top hole will move to the hole below (0 neighbours is better that 4). The 7 squirrels from the bottom hole will move up (4 neighbours is better than 6) and the 4 squirrels from the hole next to the bottom will go up (0 neighbours is better than 3)

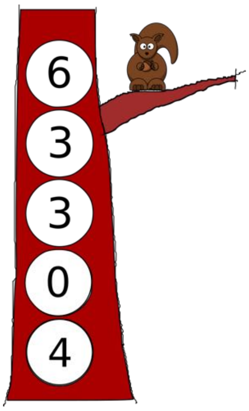


Today



Tomorrow

Question?
Here is a different situation that squirrels find themselves in:



After how many days will all the squirrels end up together in the same hole?

2 3 4 or never

Answer:

3

Explanation:

The correct answer is 3.
(6,3,3,0,4) -> (0,9,0,7,0) -> (9,0,7,0,0) -> (0,16,0,0,0)

It’s Computational Thinking:

Breaking down problems into parts (Decomposition)
Algorithms – following and describing
Digital systems
Interactions

This problem is about swarm intelligence. The idea of such algorithms is that one can solve problems with very simple devices if the number of devices is huge. For example, ants behaviour is based on simple rules and act independently of each other. But if there are many ants, they are able to do sophisticated things such as building anthills, searching for the optimal path in a graph, solving the salesman problem, or even cutting leaves.

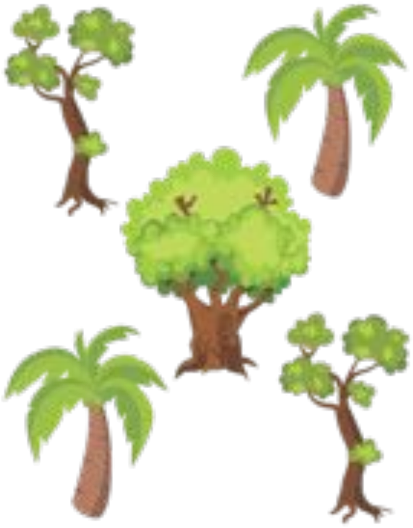
In the case of this problem, we also have a lot of devices, i.e. squirrels that behave based on simple rules. But it turns out that their collective behaviour is far from being smart. They want to live in a hole that is as roomy as possible but usually get into just one hole!

The moral of this task is, that one should carefully tune simple behaviour rules such as those of ants in ant algorithms, and it is sometimes better to cooperate than to be selfish.

Swinging Monkey

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

A leafy tree is surrounded by two bare trees and two palm trees .



Five types of bananas, say P, Q, R, S, T, are placed on the trees, a different type for each tree. A monkey swings from one tree to another tree to enjoy one banana, and then swings to another tree. It takes the monkey

- three seconds to swing from the leafy tree to any other tree, and to eat one banana (or going in the opposite direction),
- two seconds to swing from a bare tree to a palm tree or vice versa, and to eat one banana,
- seven seconds to swing between two bare trees or two palm trees while avoiding the leafy tree along the way, and to eat one banana.

The monkey swings and eats bananas of type P, Q, S, R, T, R, P.

Question:

What type of bananas can possibly be on the leafy tree if the total amount of time the monkey swings and eats is as small as possible?

P or Q or T
P or S or T
Q or S or T
Q or R or S

Answer:

P or S or T

Explanation:

The given sequence P, Q, S, R, T, R, P visits all the banana types, with six “swings”:

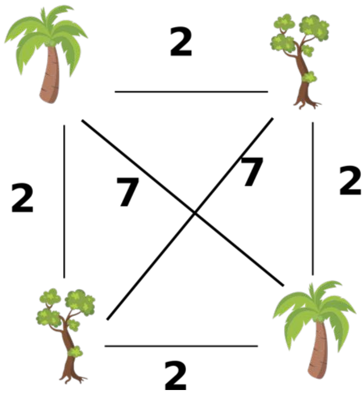
P-Q, Q-S, S-R, R-T, T-R, R-P.

To be minimal the six swings have to four 2-second swings, and two 3-second ones: thus, the sequence takes 14 seconds in total.

R is involved in four swings: S-R, R-T, T-R, R-P; R-T and T-R must take the same time, so to swing from or to R there are exactly three swings with potentially different timing. Consequently, R cannot be the leafy tree, otherwise the three swings would take 3x3 seconds, and the sequence would not be minimal. This reasoning excludes answer D, and in fact it can now been seen that T can now definitely be the leafy tree: in this case T-R and R-T take three seconds, all the other swings two.

Symmetry with respect to R should now suggest that we check what would happen if Q were the leafy tree. If Q were the leafy tree, P-Q and Q-S would take 2 seconds. Since the sequence is minimal, S-R, R-T, T-R, and R-P have to be all 2-seconds swings. However, this is not possible, as you can find out by trying to mark with S, R and T the trees in the picture on the right.

Thus, Q cannot be the leafy tree and so the only possibility left is that the leafy tree is one of P, S, or T.



It’s Computational Thinking:

Breaking down problems into parts (Decomposition)
Algorithms – following and describing
Digital systems
Interactions

This problem involves finding the best, or optimal, solution to a problem. Computers are often used to find the maximum or minimum value of some measurement. In this case, we might think of the trees as applications on a touch screen and the monkey swinging as the movement of a human finger from one application to another. A user interface designer might be interested in how to arrange the applications for a common sequence of uses so as to require as little time as possible.

L-game

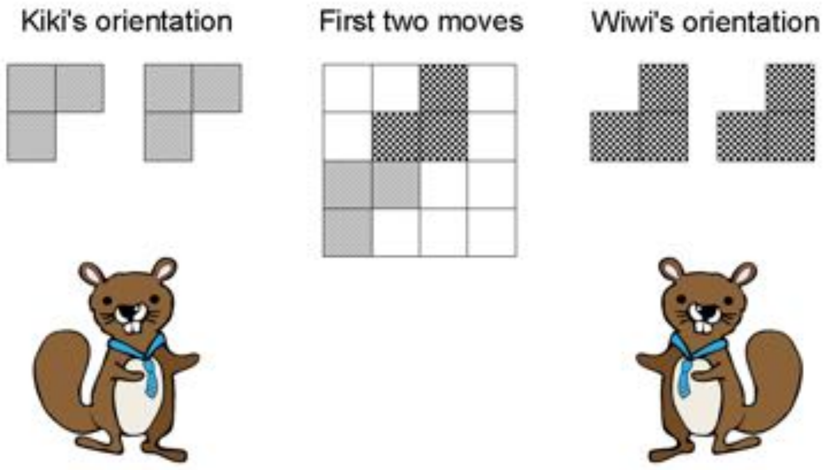
Years 3+4
Years 5+ 6
Years 7+8
Years 9+10
Years 11+12 C

Kiki and Wiwi are playing L-Game on a 4x4 board.
They take turns placing L-shaped pieces so that

- every piece placed by Kiki is oriented as shown below,
- every piece placed by Wiwi is oriented as shown below,
- every piece is placed entirely on the board, and
- no two pieces overlap.

Pieces cannot be moved after they are placed. A player loses the game when it is their turn but it is not possible to place a piece according to the rules above.

An example where Kiki goes first is shown below. In this example, Kiki can win the game by placing a piece in the bottom-right corner.



Question:
Kiki has nine possible first moves. In how many of them is she guaranteed to win no matter how pieces are placed in following turns?

0 1 2 or 3

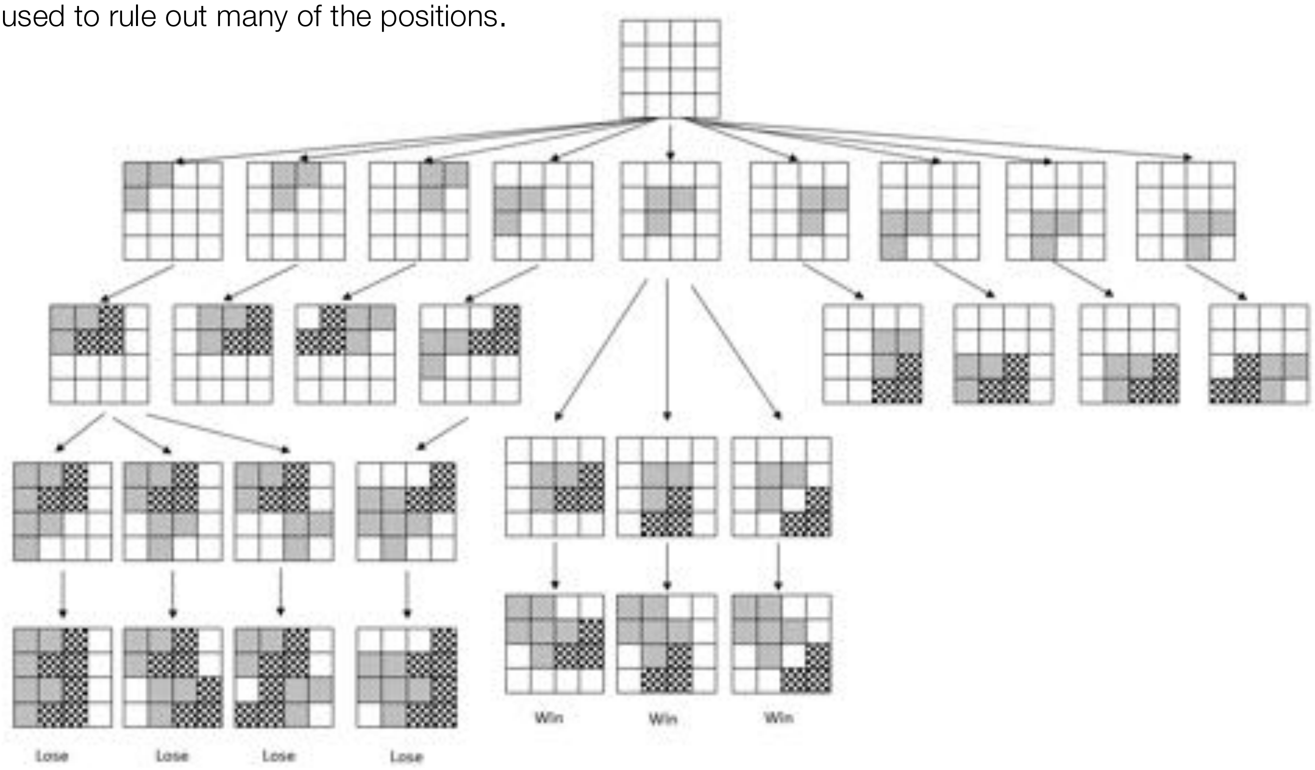
Answer:

1

Explanation:

The correct answer is B. By placing a piece in the "middle" position, Kiki is guaranteed to win the game. No matter how Wiwi places a plate on his or her first turn, Kiki can only place a piece in the top left corner. This means that, according to the rules, Wiwi cannot then place a plate.

If Kiki places a piece in any other position on her first turn, then there is always at least one way that she can lose the game. The following diagram details some of the possibilities and symmetry can be used to rule out many of the positions.



It's Computational Thinking:

Abstraction
Data interpretation – patterns and models
Algorithms – following and describing
Digital systems
Specifications

All the possibilities in a game can be represented by a diagram like the one in the explanation.

In this game tree, a root node corresponds to the initial state of the board. Then, for each possible move, arrows point to the resulting new state of the board. The full tree is built by continuing in this way. A game tree is a special type of directed graph.

A game tree can be built and searched in order to play or study the game. Depending on the type of problem, it is sometimes more useful to explore neighbour nodes first, before moving to neighbours on the next level (breadth-first search, or BFS), while sometime it Is more useful to explore as far as possible along each branch before backtracking (depth-first search, or DFS). These two search strategies have different properties and memory requirements.

Supporters

Australian Government

Queensland Government

ACT Government

Chief Minister, Treasury and Economic Development

ACCE

AUSTRALIAN COUNCIL FOR COMPUTERS IN EDUCATION

State Government Victoria

THE UNIVERSITY OF QUEENSLAND AUSTRALIA

QUT Queensland University of Technology

SAP

UNSW AUSTRALIA

Computer Science & Engineering

Faculty of Engineering

MONASH University

Information Technology

acs AUSTRALIAN COMPUTER SOCIETY

JAMES COOK UNIVERSITY AUSTRALIA

CSIRO

SWINBURNE

SWINBURNE UNIVERSITY OF TECHNOLOGY