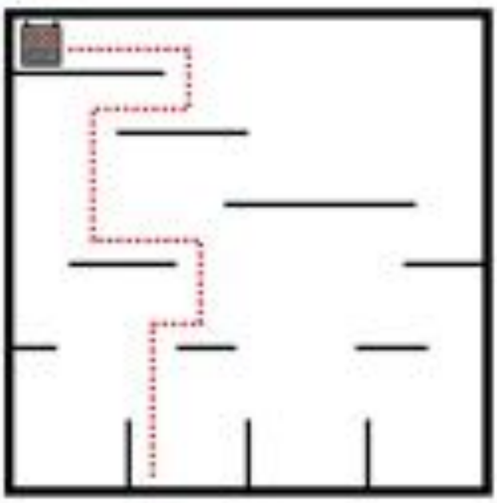
Falling Robot

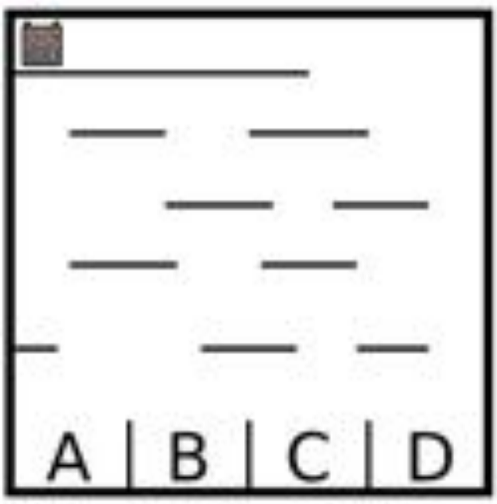
Years 3+4 A Years 7+8
Years 5+ 6 Years 9+10
Years 11+12

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.

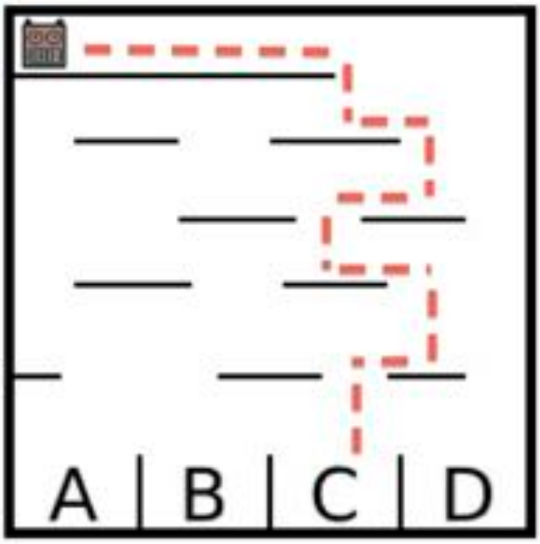


Which bucket will the robot reach in the maze below?



Answers:

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



Answer:

C

It's Computational Thinking:

Data interpretation - patterns
Algorithms - following

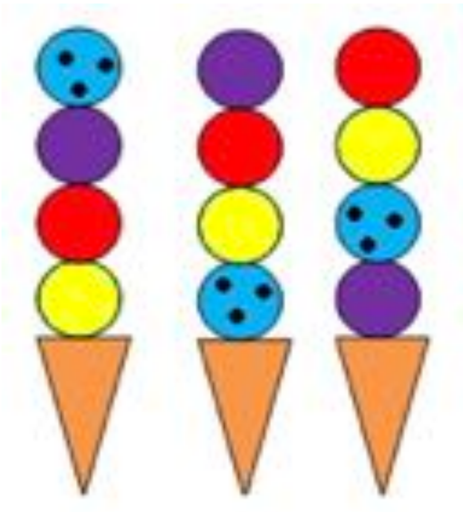
The robot follows a very basic rule, or algorithm. The student needs to understand that algorithm, either through its description, or by looking at how it performs on the left maze, then simulate that algorithm on the right maze.

Ice Cream Machine

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

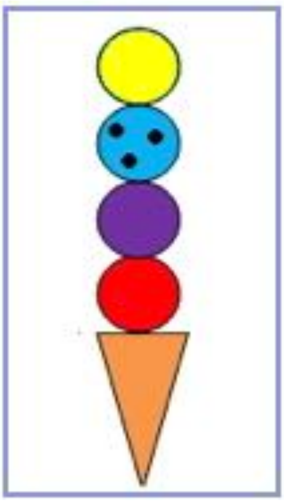
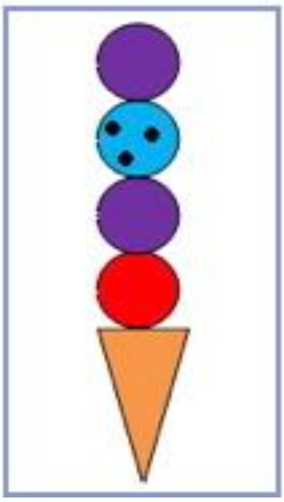
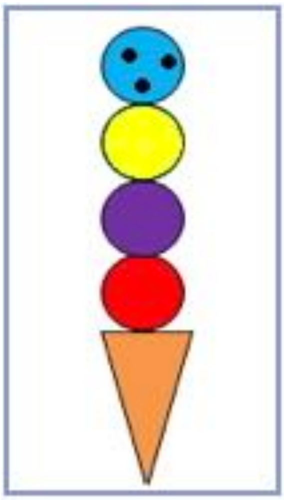
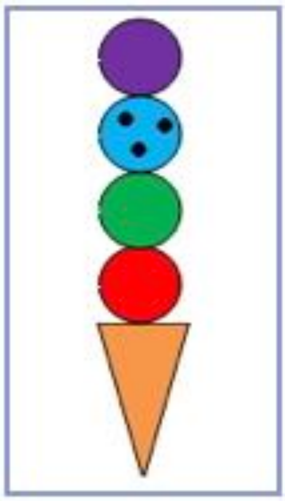
This special ice cream machine creates cones with 4 scoops of ice cream.

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
			

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:

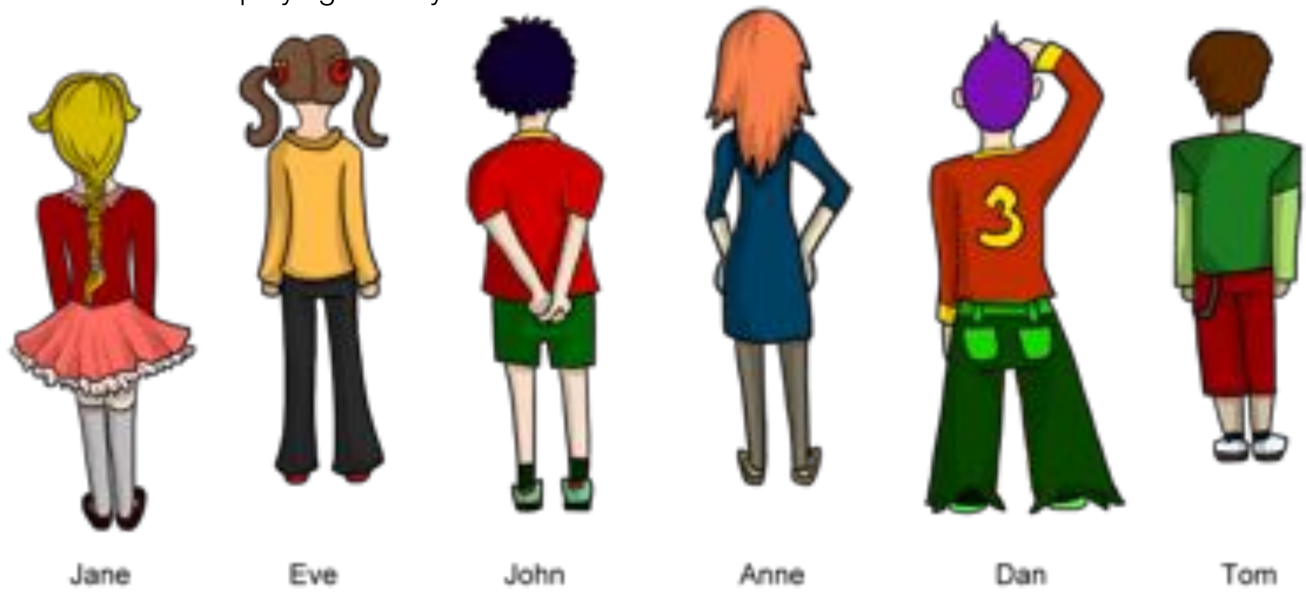
Data interpretation - patterns
Algorithms - following

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.

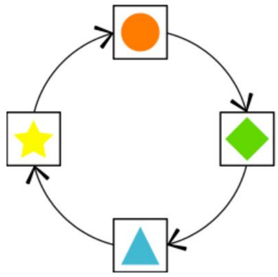
It’s Computational Thinking:

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.

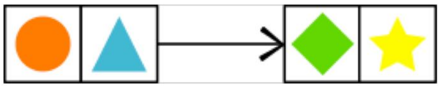
Bebras Painting

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.



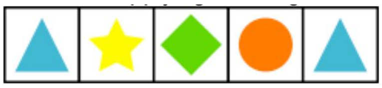
Example:

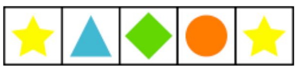
When Ben uses the magic roller to paint over the painting on the left, he gets the painting on the right.



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

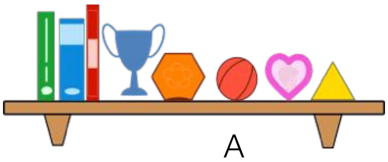
Years 3+4 A Years 7+8
Years 5+ 6 Years 9+10
Years 11+12

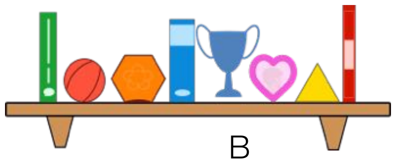
Beatrix is trying to rearrange her shelf. She has two rules:

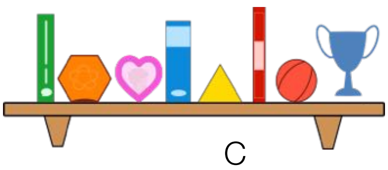
1. Rectangular items must not be next to each other.
2. Circular items must not be next to rectangular items.

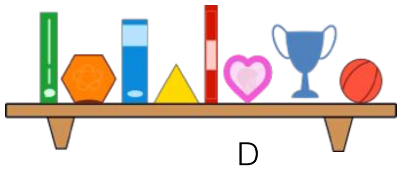
Question:

Which one of these shelves has followed her rules correctly?


A

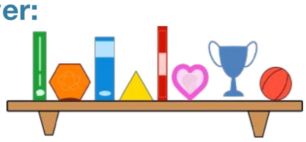

B


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

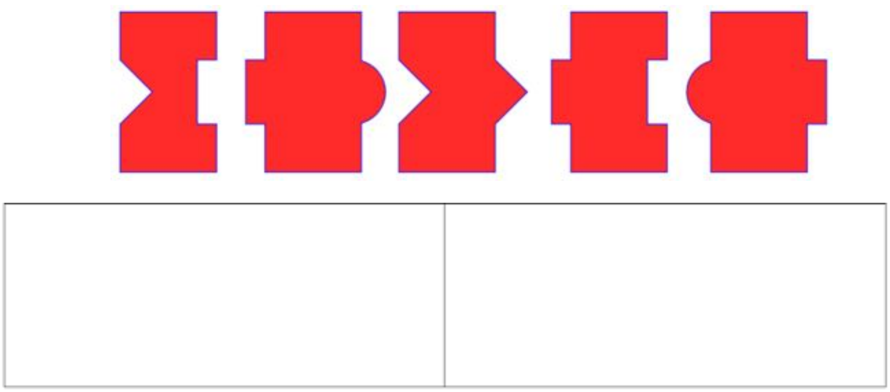
Years 3+4	C	Years 7+8
Years 5+ 6	A	Years 9+10
		Years 11+12

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

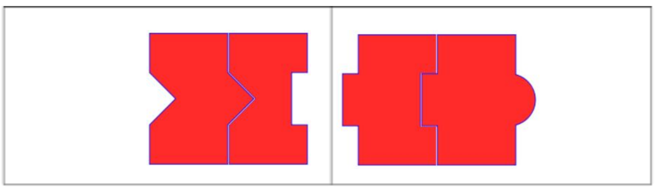
Drag two pieces that snap together into the space below to make a pair.
Then try and make another.

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

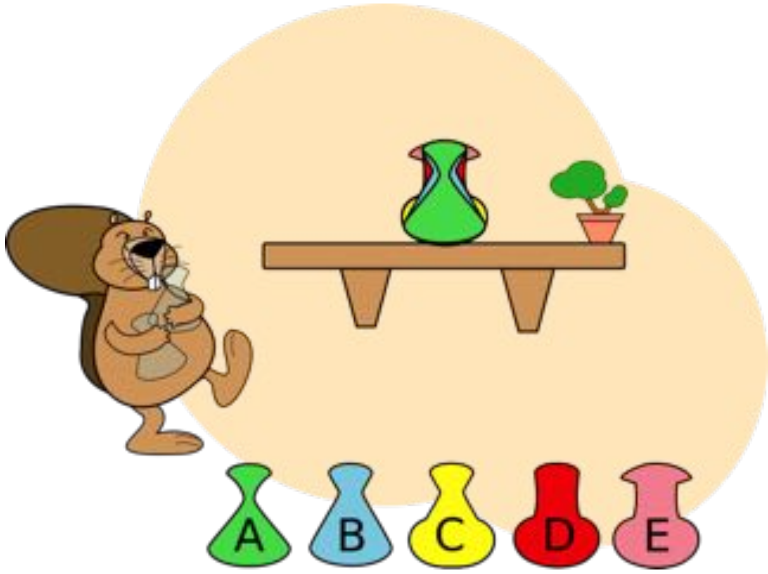
People love to optimise things: find how to run as fast as possible or the shortest route, make as much money as possible, or form as many puzzle piece pairs as possible. If the optimisation problems are particularly big or involve a lot of data, it is good idea to let the computer do the optimising.

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).