Bebras Australia Computational Thinking Challenge

2023 Solutions Guide
Round 1

Primary School
Grades 3–6

bebras.edu.au

Bebras Australia Computational Thinking Challenge

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

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

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

The Bebras international community has now grown to 60 countries with over 3.29 million students participating worldwide!

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

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

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

Engaging young minds for Australia's digital future

518
Australian schools participated in Round 1 2023

31,837
Australian students participated in Round 1 2023

3.29 million
Students participate worldwide

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

csiro.au/digital-careers

CSIRO digital careers

What is a Solutions Guide?

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

Within this Solutions Guide you will find all of the questions and tasks from Round 1 of the Bebras Australia Computational Thinking Challenge 2023. On each page above the question you will find the age group, level of difficulty, country of origin and key Computational Thinking skills.

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

Contents

What is Computational Thinking?

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

DECOMPOSITION
Breaking down problems into smaller, easier parts.

PATTERN RECOGNITION
Using patterns in information to solve problems.

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

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

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

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

More Computational Thinking resources

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

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

Computational Thinking skills alignment

2023 Round 1 Questions	Grade level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Years 3+4							
Shirts in Drawers	Easy	■			■	■	■
Most Popular Book	Easy	■	■	■			
Pearls	Easy	■	■		■	■	■
Lolly Container	Easy	■	■		■	■	■
Robot Colours	Easy	■	■			■	■
Pick-up-sticks	Medium	■	■		■	■	■
Apples Bananas Broccoli Carrots	Medium	■	■	■	■	■	■
Sticker Collection	Medium	■			■	■	■
Lila's Guessing Game	Medium	■	■		■	■	■
Building Instruction	Medium	■	■			■	■
Burger Recipe	Hard	■	■		■	■	■
Remembering Faces	Hard	■	■		■	■	■
Birthday Party	Hard	■		■	■	■	■
Robot Cleaning up Trash	Hard	■		■	■	■	■
Tug of War	Hard	■			■	■	■
Years 5+6							
Tortoise and Hare	Easy	■	■		■	■	■
Beaver World Cup Ball	Easy	■	■	■		■	■
Heart Graphics	Easy	■	■	■	■	■	■
Filling Green	Easy	■	■		■	■	■
Recommendation System	Easy	■	■	■	■	■	■
Dominoes	Medium	■	■	■	■	■	■
Sailor Necklace	Medium	■	■		■	■	■
Beehive	Medium	■	■		■	■	■
Sewing Machine	Medium	■	■			■	■
Grid Paper Pictures	Medium	■	■	■		■	■
Lights On	Hard	■	■		■	■	■
Turning Images into Numbers	Hard	■	■		■	■	■
Bombom's Message	Hard	■	■			■	■
Tangled Loop Kolam	Hard	■	■		■	■	■
Greedy Trolls	Hard	■	■	■	■	■	■

Australian Digital Technologies curriculum key concepts

Abstraction

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

Data Collection

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

Data Representation

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

Data Interpretation

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

Specification

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

Algorithms

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

Implementation

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

Digital Systems

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

Interactions

Human-Human Interactions: How users use digital systems to communicate and collaborate.
Human-Computer Interactions: How users experience and interface with digital systems.

Impact

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

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

Digital Technologies key concepts alignment

2023 Round 1 Questions	Grade level	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Years 3+4											
Shirts in Drawers	Easy			■	■	■	■		■		
Most Popular Book	Easy	■		■	■						
Pearls	Easy			■	■	■	■				
Lolly Container	Easy			■	■	■	■				
Robot Colours	Easy			■	■	■	■				
Pick-up-sticks	Medium			■	■	■	■				
Apples Bananas Broccoli Carrots	Medium	■	■	■	■	■	■		■		
Sticker Collection	Medium			■	■	■	■		■		■
Lila's Guessing Game	Medium					■	■				
Building Instruction	Medium					■	■				
Burger Recipe	Hard			■	■	■	■				
Remembering Faces	Hard			■	■	■	■	■			
Birthday Party	Hard	■		■		■	■				■
Robot Cleaning up Trash	Hard	■		■	■	■	■		■		
Tug of War	Hard		■	■		■	■				■
Years 5+6											
Tortoise and Hare	Easy			■	■	■	■				
Beaver World Cup Ball	Easy	■		■	■	■	■				
Heart Graphics	Easy			■	■	■	■				
Filling Green	Easy			■	■	■	■				
Recommendation System	Easy	■		■	■	■	■	■	■	■	■
Dominoes	Medium	■		■	■	■	■				
Sailor Necklace	Medium			■	■	■	■				
Beehive	Medium			■	■	■	■	■			
Sewing Machine	Medium			■	■	■	■				
Grid Paper Pictures	Medium	■	■	■	■	■	■		■		
Lights On	Hard			■	■	■	■	■	■	■	
Turning Images into Numbers	Hard		■	■	■	■	■	■			
Bombom's Message	Hard			■	■	■	■				
Tangled Loop Kolam	Hard			■	■	■	■				
Greedy Trolls	Hard	■	■	■	■	■	■	■	■		

Bebras Challenge 2023 Round 1

Years 3+4

This question comes from Pakistan

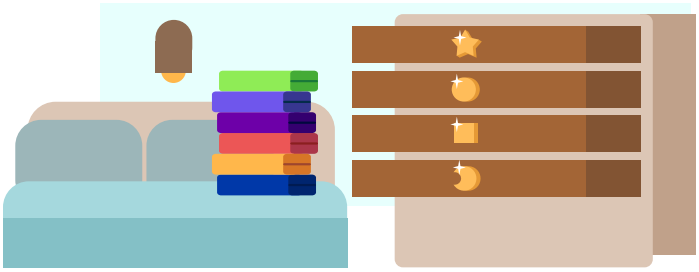
Shirts in drawers

Buffy has a pile of six clean shirts on her bed. She wants to put them into her four drawers.

She takes one shirt at a time from the pile and puts it into a drawer. Buffy starts with the top drawer, then uses the second from the top, and works her way down. When she has put a shirt into the bottom drawer, she starts again from the top.

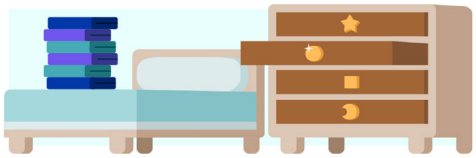
Question

Into which drawer will she put the last shirt?



EXPLANATION

Answer



Explanation

Buffy puts the first shirt in the ★ drawer.

The second shirt gets put into the ● drawer.

Then the third and fourth shirts get put into the ■ and ◗ drawers.

Since the ◗ drawer is the bottom drawer, the fifth shirt gets put into the ★ drawer again.

Finally the sixth shirt gets put into the ● drawer.

BACKGROUND INFORMATION

6 divided by 4 is 1 with a remainder of 2. This 2 represents the second drawer of this task. So in this task Buffy has to find the remainder of a division. In computer science, this calculation is called the modulo operation.

The modulo operation is very useful in computer science. Secure communication for instance between your web browser and a website server uses this operator. Also, in large networks where several lines connect the same two computers, the data being sent is distributed equally across these lines by the help of the modulo operation. So trying to equally distribute shirts into drawers is actually a very informatical process!

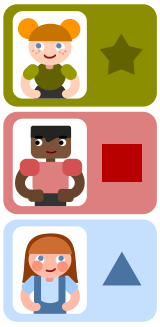
Most popular book

Three children often borrow books from the library.

The librarian wants to know which book was borrowed the most.

She made the *borrowing table* shown on the right.

The *borrowing table* is between the 3 childrens' library cards and the 4 books.



Question

Which book was borrowed the most?

?	■	★	★	▲	■	▲	★
?	1	1	2	1	1	3	2

EXPLANATION

Answer

Explanation

The table above shows the number of times each book was borrowed.

The crocodile book was borrowed by two children.

The cat book was borrowed by three children.

Only one child borrowed the chicken book and only one child borrowed the beaver book.

That means, the cat book was borrowed the most.

BACKGROUND INFORMATION

Data tables are a way of storing information.

Data tables are useful as not only are they a format that can be understood by computers, but they can help to visually show relationships between the data.

Pearls

Monica and Michael brought necklaces back from their holidays.

Monica's necklace:

Michael's necklace:

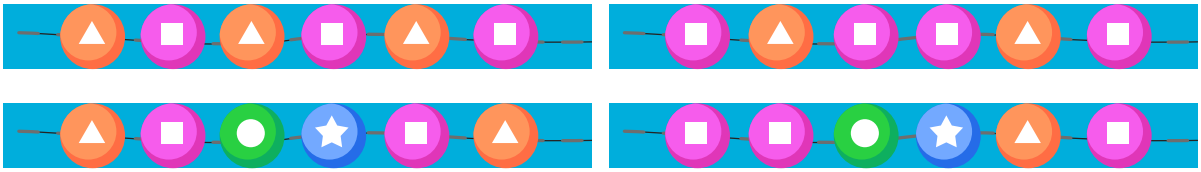
They each took several pearls from their own necklace to make a new necklace for their friend Anastasia. Now their necklaces look like this:

Monica's necklace:

Michael's necklace:

Question

Which is Anastasia's necklace?



EXPLANATION

Answer

Explanation

Monica's necklace is missing three violet pearls and Michael's is missing three red pearls, and that is exactly what is on this necklace.

 is not correct because it contains four violet pearls. Monica's necklace started with five violet pearls and she now has two left, so the largest number of violet pearls that can be on Anastasia's necklace is three.

 is not correct because it contains blue and green pearls. All the green and blue pearls remain on Monica's and Michael's necklaces, so Anastasia's necklace cannot have any green or blue pearls.

For the same reason, answer  is not correct.

BACKGROUND INFORMATION

In some high-level computer programming languages such as Python, a *set* is an abstract data type that allows binary operations such as subtraction. This task illustrates an algorithm for *set subtraction*.

Without needing to know anything about the mathematics of sets, the total collection of pearls must be the same before and after six pearls are used for Anastasia's necklace. As such, the multistep algorithm someone solving this problem might use involves counting repeats of various colours, populating a list of numbers (or iterating over the different colours), and keeping track of intermediate comparisons.

These operations are all that are needed for a huge range of useful algorithms in computer science.

This question comes from the United Kingdom

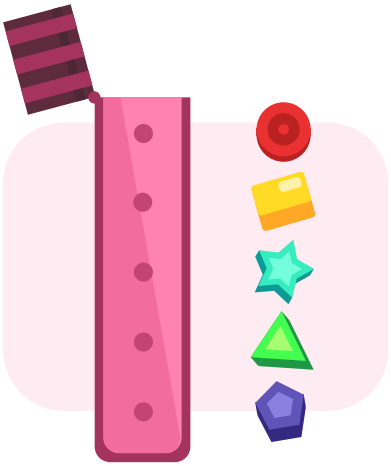
Lolly container

Brian's favourite type of lolly comes in five flavours. Brian puts one of each flavour into a container to take to school. During the day, Brian eats the lollies in the order they come out of the top of the container.

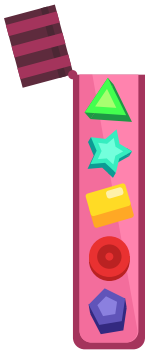
Today he wants to eat them in this order:

Question

Pack Brian's container for him so that he gets the lollies in the preferred order.



Answer



EXPLANATION

Explanation

For the sweets to come out of the tube in the preferred order, it is important to understand that the first one in will be the last one out. This means that the tube of sweets needs to be filled like the image above.

BACKGROUND INFORMATION

Making sure that things are ordered is an important concept in computer science. This applies to everything, from instructions in an algorithm to storing data in a table.

The actual order used in this task is called a stack order. There are many ways to take items out of a stack. In this task, the stack adds and subtracts items based on *Last In First Out* (LIFO), where the latest item put into the stack is the first one that is taken out. This differs from *First in First Out* (FIFO), where the oldest item placed in the stack is the first taken out.

This question comes from
South Africa

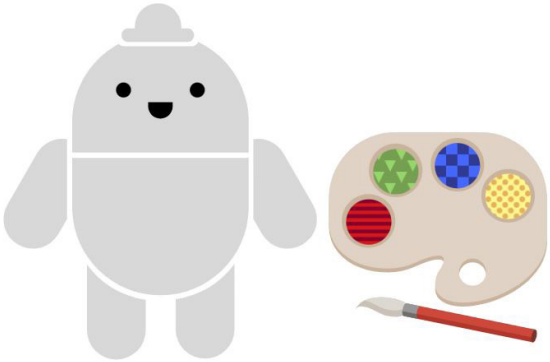
Robot colours

Taweeq has four robot friends. Each robot is painted using four colours: red, blue, green and yellow.

Each robot has a unique colour pattern. This means that there are never two robots with the same part painted in the same colour.

Question

Based on the first three robots, what will the fourth robot look like? Colour the robot.



Answer

EXPLANATION

Explanation:

Let's start from the hat. The first three robots have green, red and blue hats respectively. The only remaining colour that can be used to paint a hat is yellow. Similarly, to be different to all the other robots, the arms must be red, the face green, the body yellow and the legs blue.

BACKGROUND INFORMATION

Programmers create sets of instructions for computers to follow, called *algorithms*. These rules give the computer a way to work out what to do in a given situation. In this example, we have created a set of rules that tell us how the robots must be painted.

Data can take many forms, for example, pictures, text or numbers. When we look at data in this question, we are looking for a sequence of images that will assist in solving the problem. By identifying these images we can make predictions, create rules and solve more general problems.

This question comes from Brazil

Pick up sticks

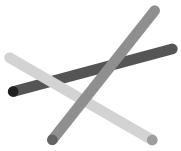
Ruan is playing the game 'pick-up sticks'.

He drops some sticks on a table and then picks them all up following these rules:

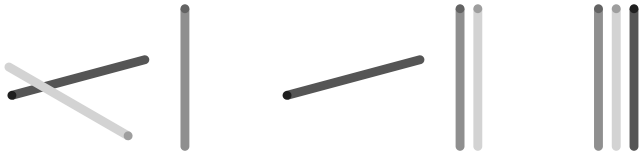
- Pick up one stick at a time

- Only pick up a stick if no other stick is covering it

Ruan plays a game with 3 sticks that look like this when dropped.



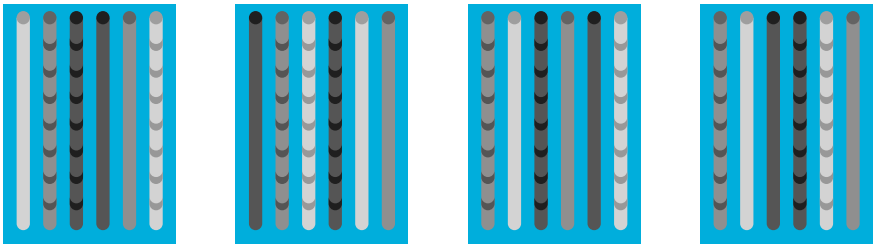
Ruan picks them up in order, and places them from left to right:



Question

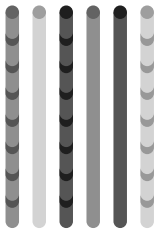
Ruan now drops 6 sticks like this:

In which order should he pick them up?



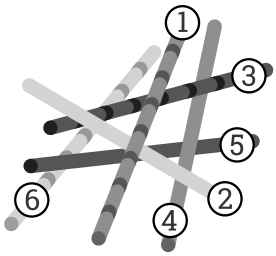
EXPLANATION

Answer



Explanation:

The sticks above are arranged in order, from left to right, from top to bottom in the pile of sticks. Every time the top stick is removed, there is then only one stick that is not covered by another. The order the sticks are picked up is shown on the right:



BACKGROUND INFORMATION

Colour and pattern sequences are a common way of representing information in a diagram, a type of representative data structure. Diagrams are useful tools to simplify abstract information in computer science, and allow for a solution to be found.

In this question, using a diagram allows for a stepwise pattern to be found that allows for the sticks to be picked up while still following the game rules. This is an example of an *algorithm*.

Furthermore, the act of removing a stick can be seen as an operation in an abstract structure, such as a graph, that can be done only if certain requirements are met. After a sequence of valid *transformations* (the act of removing sticks from the stack), the result can be seen as a simplified or reduced version of the original structure (stacks of sticks getting smaller and smaller).

Apples, bananas, broccoli and carrots

Some fruit (apples and bananas), and some vegetables (broccoli and carrots) are placed on four plates:

Then the following actions are performed, in order:

1. One banana is added to each plate.
2. Each plate with less than four items, in total, is removed.
3. All the fruit is removed from the remaining plates.
4. Any plates that have only carrots on them are removed.

Question

How many plates remain after all the actions are performed?

1 2 3 4

EXPLANATION

Answer

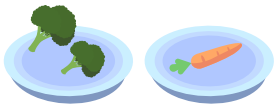
The correct answer is 1.

Explanation

After the first action, the plates will look like this:

After the second action, two plates will remain:

After the third action, the plates will look like this:



After the fourth action, one plate will remain:

The only plates remaining after all the actions have been taken, will be those that start with at least three items and don't hold only fruit and carrots. Exactly one plate fits these criteria.

BACKGROUND INFORMATION

To answer this task, we can carefully follow the steps of an *algorithm*. Each step consists of an action where the *input* and *output* of the action is a list of plates.

Two of the actions involve changing which fruits and vegetables are on each plate. Specifically, the first action adds a banana to each plate and the third action removes all the fruit from each plate. These two actions are a type of *mapping* which involves performing the same operation on a collection of items.

The other two actions involve removing (and keeping) some of the plates. These are examples of *filtering* where an operation is performed on each item in a collection which determines which items remain in the collection and which items are removed from the collection.

When computer scientists implement algorithms using a programming language, they often make use of common patterns such as mapping and filtering. Different programming languages emphasise different patterns. For example, Python is a popular modern language with features built into the core language to support filtering and mapping.

Sticker collection

Four students collect stickers of things they like. Each student likes only one thing. They want to share four new stickers so that each student gets a sticker that fits their collection.

Ajay does not like cars but loves snacks.
Divya does not like flowers.
Ram exchanged his car sticker with Divya.
Seema is an animal lover.

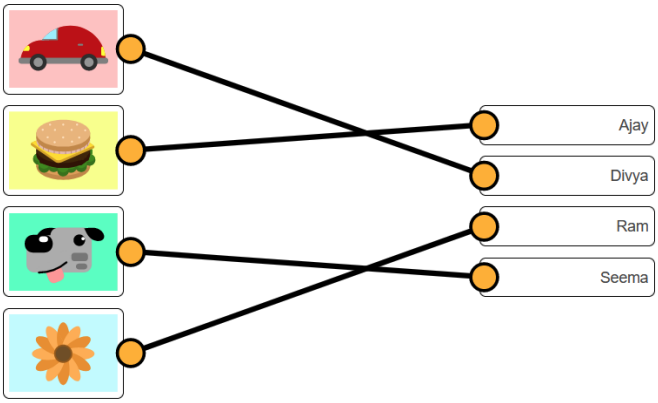
Question

Which will be the final stickers that Ajay, Divya, Ram and Seema get?

Ajay
Divya
Ram
Seema

EXPLANATION

Answer



Explanation

We can obtain this answer by recalling the conditions mentioned in the task.
Ajay likes snacks, so gets the burger sticker.
Seema is an animal lover, so gets the dog sticker.
This leaves the flower and car stickers remaining.
Divya does not like flowers, and Ram exchanged his car sticker with Divya.
This means that Divya will get the car sticker, and Ram will get the flower sticker.

BACKGROUND INFORMATION

In this task we make use of logical thinking to satisfy the interests of Ajay, Divya, Ram and Seema. Each have a particular interest and there was only one way to distribute the four stickers in order to have everybody satisfied.

Logical thinking is very important when it comes to programming. A computer programmer must make use of *logical operators* (such as “and”, “or”, and “not”) in order to write a program which is able to satisfy certain conditions.

In this question, a computer scientist might break down the task into the following steps:

- Identify the result that they want to achieve;
- Identify the facts that are available;
- Break the problem statement into smaller parts and filter out irrelevant data;
- Always keep in mind the rules or conditions mentioned in the task;
- Keep track of the actions execution (i.e., Divya and Ram exchanging stickers);
- Test the effectiveness of the result by working backwards and be open to making revisions.

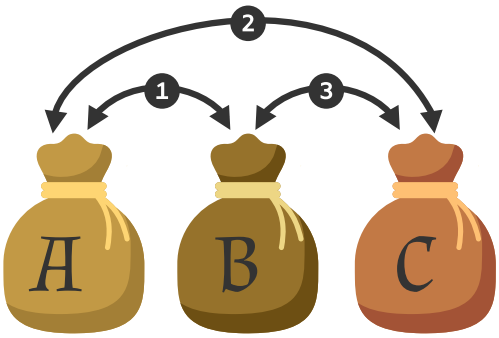
Lila's guessing game

Lila and her friends are playing a guessing game.

To start the game, Lila puts a marble in bag A, a gem in bag B, and a piece of paper in bag C.

Then, while her friends' eyes are closed, Lila moves the items in the bags.

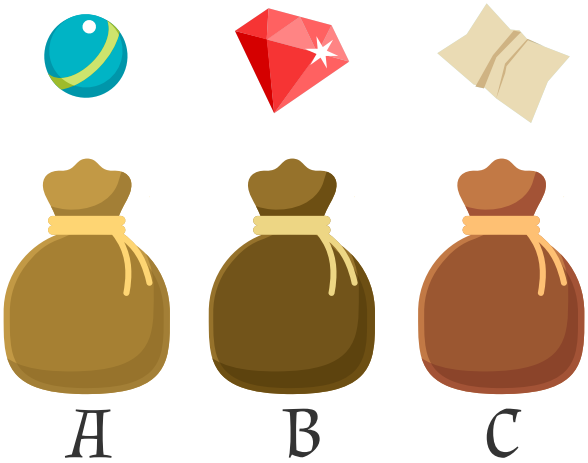
First, she swaps the items in bags A and B,
Then, she swaps the items in bags A and C,
Lastly, she swaps the items in bags B and C.



Question

Where are Lila's items now?

Place each item into the correct bag.



Continued on next page

Lila's guessing game - continued

EXPLANATION

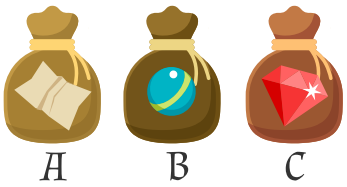
Answer

The paper is in bag A, the gem is in bag B and the marble is in bag C.

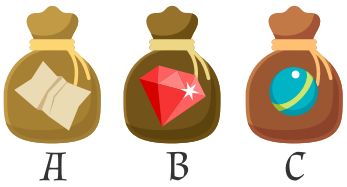
Explanation

Lila switches the items three times. After the first switch, the bags look like this:

After the second switch, the bags look like this:



After the third and final switch, the bags look like this:



BACKGROUND INFORMATION

A *permutation* is an arrangement of objects in a particular order. Arranging the objects in a different order creates a different permutation. So the same group of objects can have many permutations. At the start of this task, Lila's items are in the permutation: marble-gem-paper. At the end of the task, the same items are in a different permutation: paper-gem-marble.

Permutations are related to *sorting*. A sorted list is simply one of many possible permutations of that list. Sorting is a common task in computer science. For instance, the files in a computer folder are often sorted by name or by date.

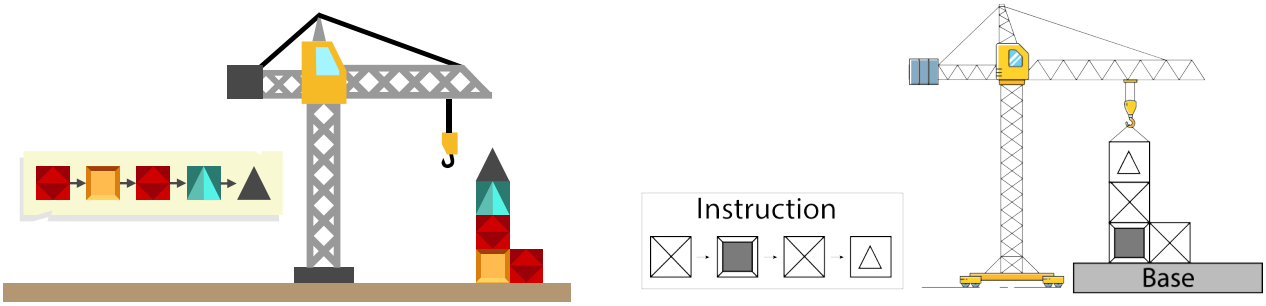
Many different sorting algorithms or sorting techniques have been developed. All sorting algorithms begin with the same permutation (the unsorted list) and they all end with the same permutation (the sorted list). The difference is what happens during the sorting process. The list will pass through many other permutations, but exactly which permutations those are depends on which sorting algorithm has been used.

In Beaverland, block houses are built by following instructions.

A crane takes each block in the order shown in the instructions. It then puts it on the base or on top of another block.

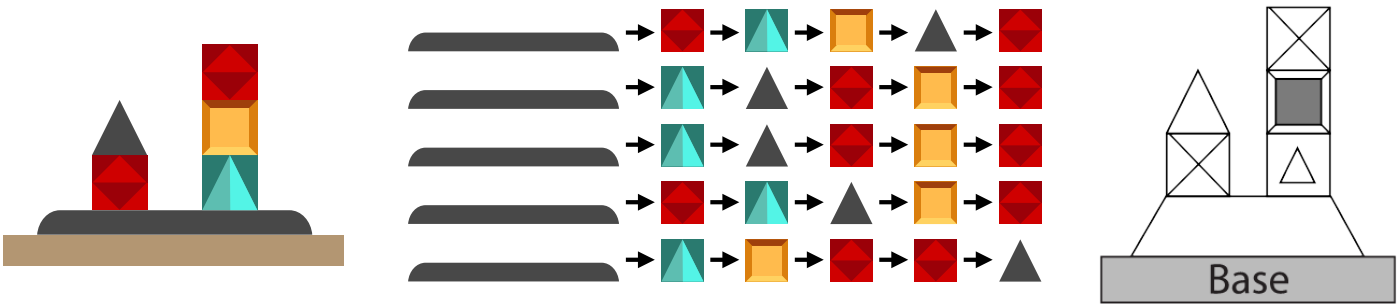
Building instruction

For example, the house shown in this image could be built following the instructions below:



Question

Which of the building instructions below *cannot* be used to build the house below?



EXPLANATION

Answer

Explanation

In the house shown, the triangle block is placed on top of a red block, so at least one the red blocks must be placed before the triangle block. This is not the case with the instructions above. All other instructions can be followed to build the house shown.

BACKGROUND INFORMATION

Computers operate by following a list of instructions, called a *program*, that has been written to carry out a particular task. Programs are written in languages that have been specially designed to tell computers what to do with a limited set of instructions. Some languages are more suitable for some purposes than others.

Regardless of what language they use, programmers must become adept at specifying exactly what they want the computer to do. Unlike human beings, a computer will carry out instructions to the letter even if they are patently ridiculous!

Burger recipe

Mei is making burgers according to the rules below.

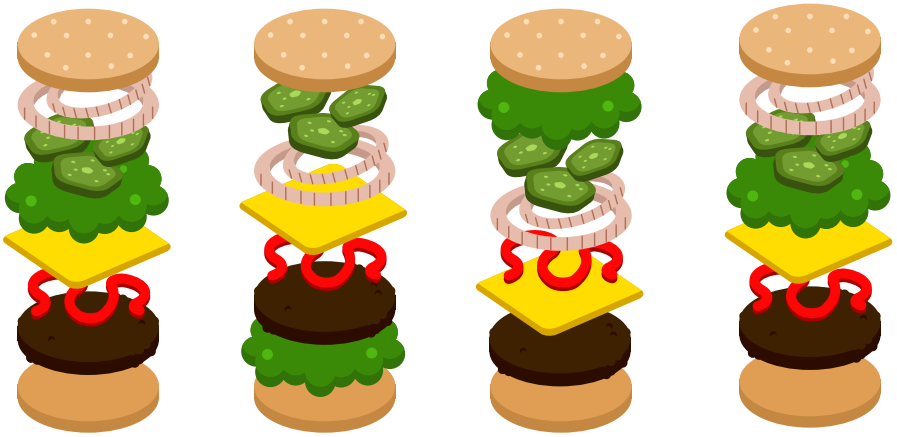
Rules:

1. The sauce should be right above the meat.
2. Meat and cheese should be below the pickles, lettuce and onions.
3. Onions should not be in contact with the buns.

Buns	Meat	Sauce	Pickles	Lettuce	Onions	Cheese
						

Question

Which burger is correctly made according to the rules?



EXPLANATION

Answer

Continued on next page

Burger recipe - continued

Explanation

This hamburger follows rules #1 and #2, but the onion touches the top bun, so it doesn't follow rule #3.

This hamburger follows rule #1, but the lettuce is below the meat and the cheese, so rule #2 was not followed.

The cheese is between the meat and the sauce, so it doesn't follow rule #1.

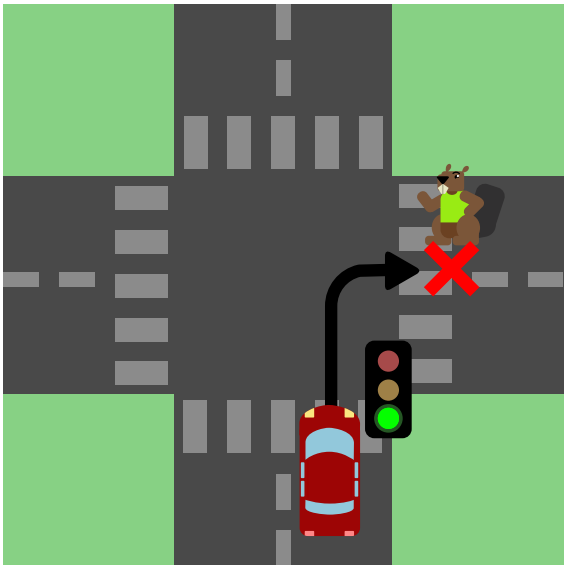
This hamburger satisfies all the rules, so this is the correct answer!

BACKGROUND INFORMATION

In computer science, finding out whether a solution obeys all the given rules is called *constraints checking*.

In the figure below, a self-driving car has to decide whether it can turn right.

It uses two rules:



- The light must be green.
- There should be no pedestrians crossing.

Checking whether a given solution satisfies the constraints is one thing, finding such a solution is another. This is called the *constraint satisfaction problem*.

Most of the time, finding a solution is much, much harder than just checking constraints, even for a computer.

This question comes from Australia

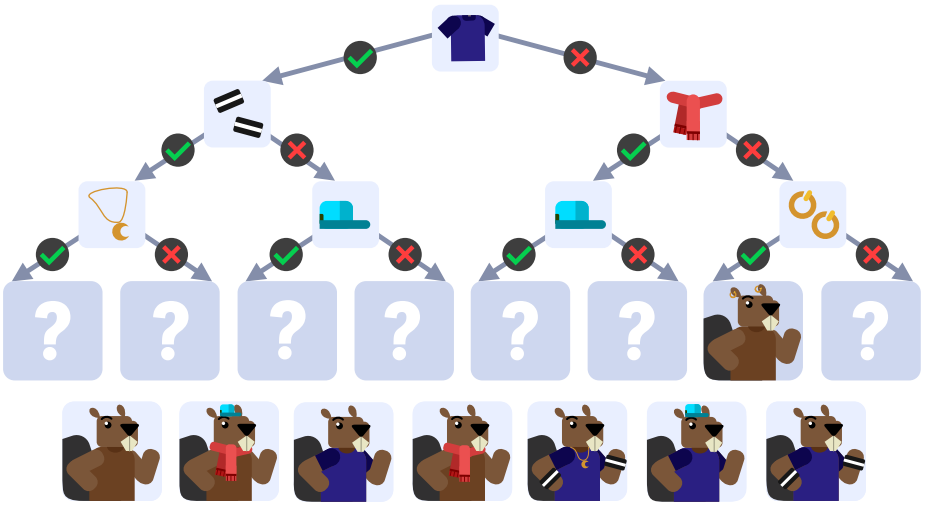
Remembering faces

A group of 8 friends want to make sure that each one of them is wearing a different outfit to school. The friends have created unique outfits using a decision tree of different clothing and accessories.

The first friend has arrived as shown below. According to the decision tree, they did not wear a shirt or scarf, but did wear earrings.

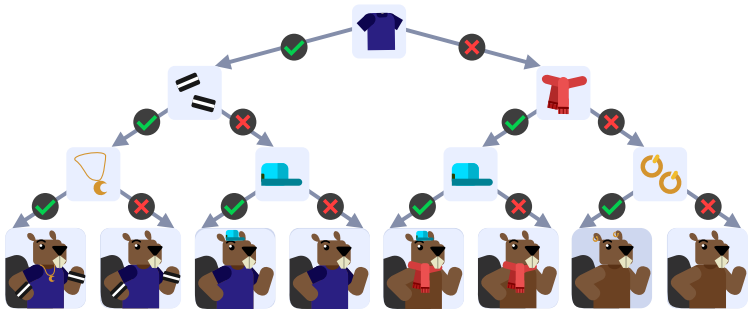
Question

Put the remaining pictures of the different outfits in the right place by dragging them to the question marks.



EXPLANATION

Answer



Explanation

There are multiple ways to come to this solution. One of the most straightforward is to pick an image of a beaver, and follow the arrows from top to bottom. We can answer each question in turn ('yes' or 'no'), to identify where that beaver should be placed.

BACKGROUND INFORMATION

In this task we can put pictures in the right place by answering questions. At each question we have to decide whether it is true or false by saying 'yes' or 'no', and moving down left or right. Data with only two categories, such as yes/no in this task, is called *binary*.

In computer science we call this type of picture a *decision tree*. With decision trees we can sort things into categories.

We also use this concept in an everyday life. For example, we can play a game of guessing a word or number by answering questions with just 'yes' and 'no'.

Birthday party

Maxie the beaver is planning a birthday party. He makes a to-do list with tasks to be done before the party, shown below:

To-do list					
	Making sure how many beavers are coming	Buying snacks	Picking a date	Estimating the cost	Choosing a venue
Tasks to be done beforehand			None		

Maxie realises that some tasks need to be completed before other tasks. For example, picking a date before confirming how many beavers are coming.

Question

What is a correct order to complete the tasks?

Continued on next page

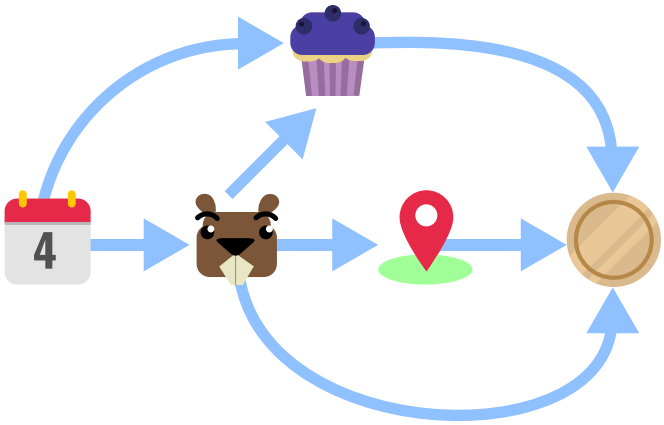
Birthday party – continued

EXPLANATION

Answer

Explanation

If we take the information from the table and use arrows to represent the order of carrying out tasks, we will get the following picture:



In this picture, each circle represents a task, and each arrow represents which task should be done before the other. Based on the picture, we can see that there are only two possible solutions for this problem. Picking the date always has to be the first task to be done. Counting people needs to be the second task. Either buying the snacks or choosing a venue can be placed next. The last task will always be estimating the cost.

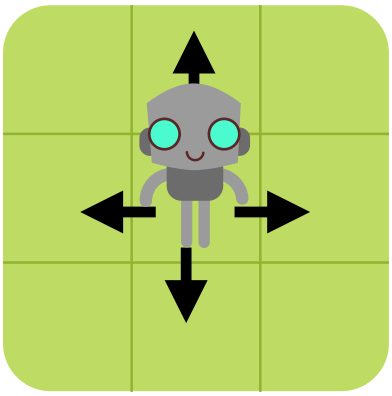
BACKGROUND INFORMATION

The algorithm for sorting these tasks is called a *topological sort*. Computers are equipped to sort items with numerical order. However, when items to be sorted do not possess any numerical characteristics, or when we want to sort items based on non-numerical characteristics, we have to let the computer know the sequence relationship between each item. We can often use a graph like this to represent the sequence.

In our daily life, topological sorts are applied in scheduling a series of jobs or tasks. For example, when playing online games, we have to pass a certain level before we can play the next level.

Robot cleaning up trash

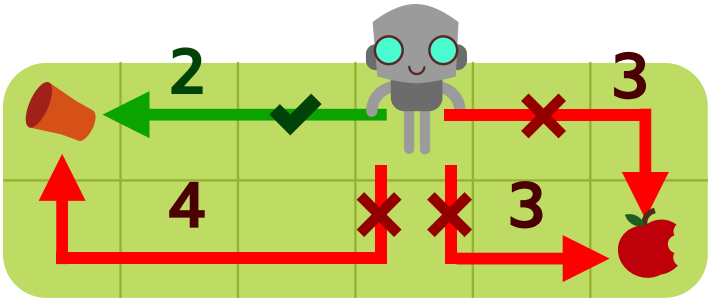
Rina cleans up after her summer party. Luckily TUX the robot helps her. TUX can only move up, down, left or right. Moving from one square to another counts as 1 move.



TUX works like this:

- TUX automatically detects the nearest piece of garbage.
- TUX moves to the nearest piece of garbage.
- TUX picks the garbage up.
- From there TUX again detects the nearest piece of garbage.
- TUX repeats this until all of the garbage is picked up.

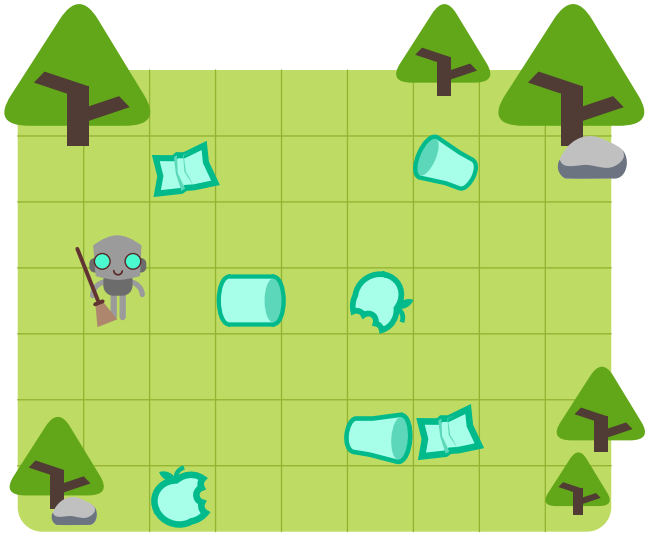
In this example, TUX would now move to the paper cup by taking 2 steps left. This is the nearest piece of garbage and the shortest way to get there. This is why TUX wouldn't move to the apple, or take 4 steps to get to the paper cup.



Rina starts TUX to pick up all of the garbage below.

Question

Which piece of garbage will TUX pick up last?



Continued on next page

This question comes from China

Robot cleaning – continued

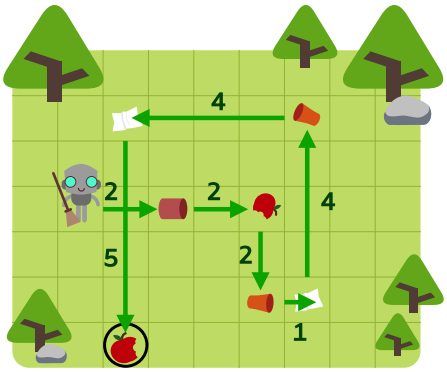
EXPLANATION

Answer

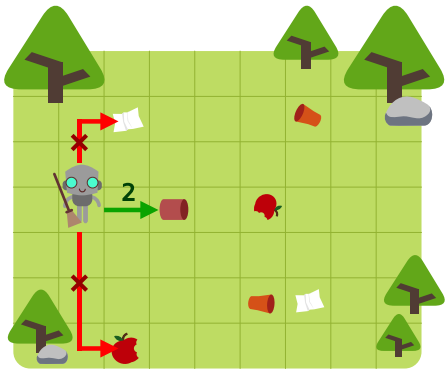
The correct answer is the apple at the bottom left.

Explanation

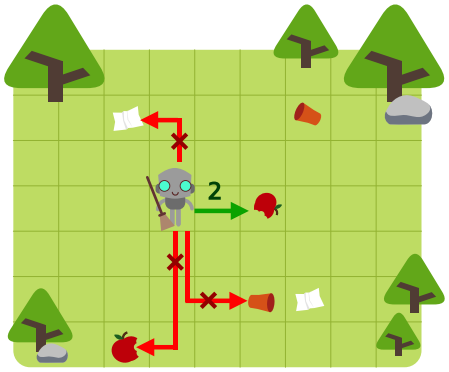
TUX follows this path:



TUX starts and detects the first, closest piece of garbage:



Then TUX moves to the piece of garbage and picks it up. Then TUX has to detect the next closest piece of garbage:



At every new position, TUX has to detect where the closest piece of garbage is, and ends up moving and cleaning as shown in the first picture.

BACKGROUND INFORMATION

Artificial intelligence (AI) is an area in the field of informatics. The goal of AI is to create an agent (software or hardware) that can act intelligently. An intelligent agent is an agent who can make the best (or at least an optimal) decision from all given possibilities. Depending on the problem, various agent types can be made, such as learning agents, logical agents, or search agents.

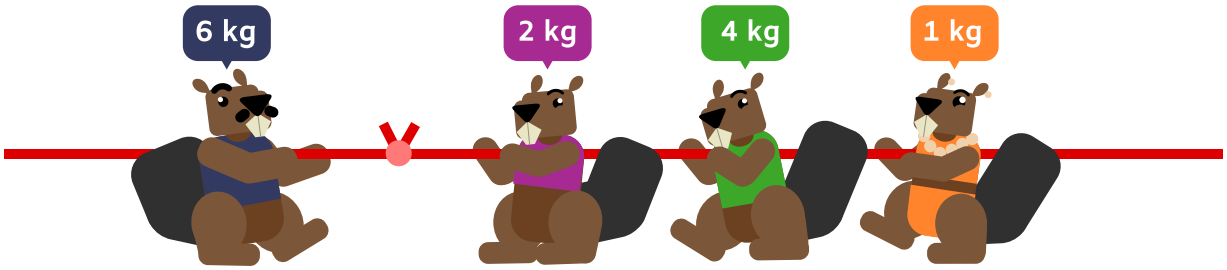
In this task, the robot can be considered a searching agent. This robot is tasked with collecting garbage. To be able to collect all the trash, it needs to define a route. In this case, the route search is performed using the rule: The nearest garbage must be picked up first.

This search technique is known as a greedy *search algorithm*. The use of greedy search algorithms cannot guarantee the best solution. For example, it cannot be guaranteed that the number of moves made by the robot is minimal. That is not a problem for this task, because the goal is to simply collect all the garbage, and not to collect all the garbage within a minimum number of moves.

Tug of war

Beaver School is having a tug-of-war competition. In order to have a fair competition, the school divides the beavers into teams based on their weight. The goal is to have the weight difference between these two teams as close as possible.

In this example, the weight difference between the two teams is only 1 kg.



Question

Today five beavers sign up for the competition. Drag the beavers into the Team A and Team B rectangles so that the weight difference between the two teams is as close as possible.

Team A

Team B

Continued on next page

This question comes from
Taiwan

Tug of war - continued

EXPLANATION

Answer

There are 4 possible correct answers:

Team A: 1, 4, 5 (10 kg), Team B: 4, 9 (13 kg), or swapping teams

Team A: 4, 9 (13 kg), Team B: 1, 4, 5 (10 kg)

or

Team A: 4, 4, 5 (13 kg), Team B: 1, 9 (10 kg), or swapping teams

Team A: 1, 9 (10 kg), Team B: 4, 4, 5 (13 kg)

Explanation

The total weight of all five beavers is 1+4+9+4+5=23kg. Ideally, the total weight of each team would be exactly the same, which is half of 23 (11.5). However, each beaver's weight is a whole number, so the total weight of each team will also be a whole number.

Under this constraint, the ideal total weight of one team is 11 and the other is 12. Unfortunately, no combination of any of the five beavers results in a total of 11 or 12. So, we have to try making the weight of the two teams to be 10 (11-1) and 13 (12+1).

In this case, the 10kg team can be (1, 4, 5) or (1, 9), and the other team can be (4, 9) or (4, 4, 5).

BACKGROUND INFORMATION

The concept in this task is the *knapsack problem* in computer science, which is a *combinatorial optimisation problem*. The goal of a knapsack problem is to choose a combination of items of various weights and values, and maximize the total value within a limited total weight. We can treat each team in this task as a knapsack. A 0/1 knapsack problem is when we only consider putting 0 or 1 of each item in the knapsack.

In life, similar problems often arise in the fields of business, combinatorics, computational complexity theory, cryptography, and applied mathematics. For example, going on the most rides in an amusement park, or traveling the longest distance within the budget. The optimal solution to this sort of problem can be obtained through *dynamic programming* methods.

Bebras Challenge 2023 Round 1

Years 5+6

Tortoise and hare

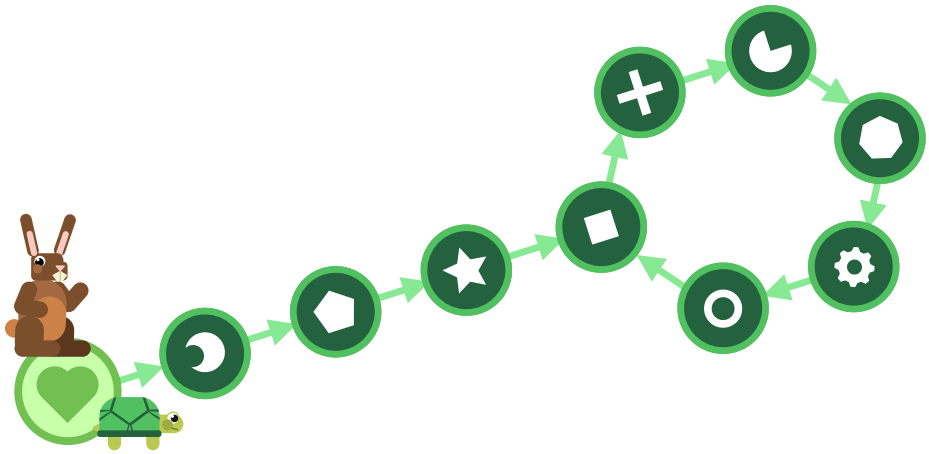
A tortoise and a hare are about to race each other.

They both start at the same time in the circle with a heart in it. They follow the direction of the arrows on the track.

- The tortoise moves one circle  every minute.
- The hare moves two circles  every minute.

Question

Select the circle where the tortoise and the hare meet for the first time after starting the race.



EXPLANATION

Answer

The tortoise and the hare will meet on the following spot:

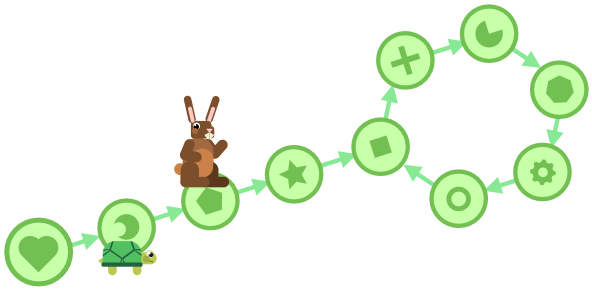
Continued on next page

Tortoise and hare - continued

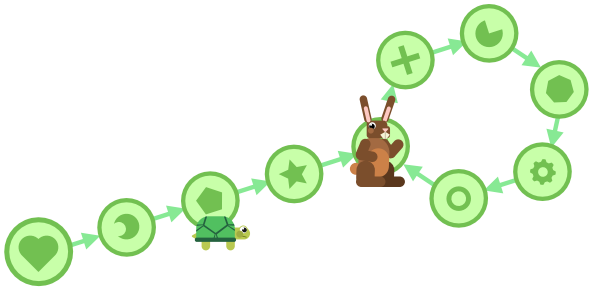
EXPLANATION

The figures below show the locations of the tortoise and the hare after every minute:

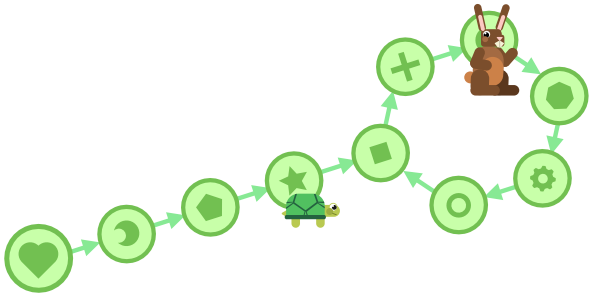
After 1 minute:



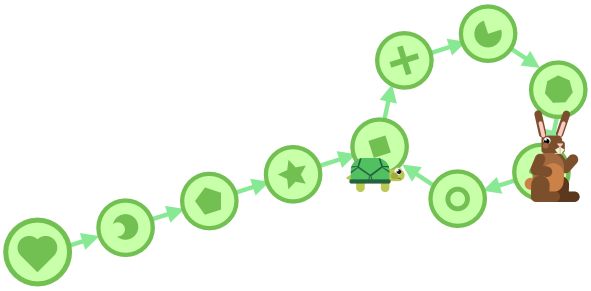
After 2 minutes:



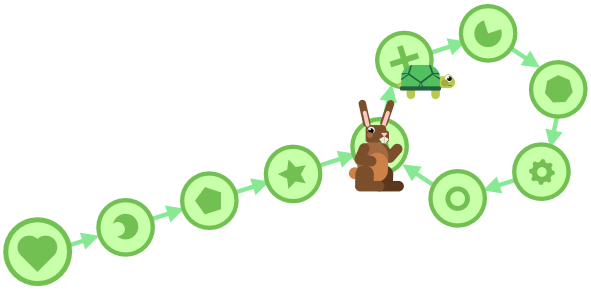
After 3 minutes:



After 4 minutes:



After 5 minutes:



After 6 minutes:

BACKGROUND INFORMATION

The task is based on traversing a specific data structure with one-way traffic, a *directed graph* with a loop or a cycle.

Cycle detection is an important task in computer science. For instance, it can be used to check if code is repeating a sequence of tasks endlessly *(infinite loop),* which prevents the program from stopping.

A more advanced application is related to the analysis of the quality of *random number* generators, especially those that are used in the encryption or protection of sensitive data. They usually have a pre-cycle that does not repeat and then cycle. The part of the path in this task represented as a straight line corresponds to the pre-cycle, and the round part corresponds to the cycle. Having longer cycle lengths is a key characteristic that makes secure algorithms stronger and more difficult to crack.

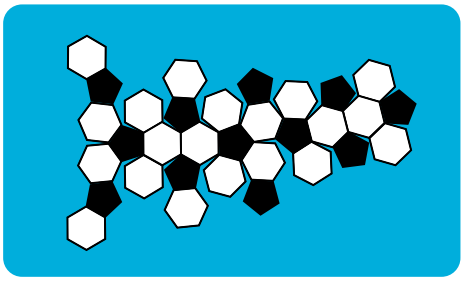
Beaver world cup ball

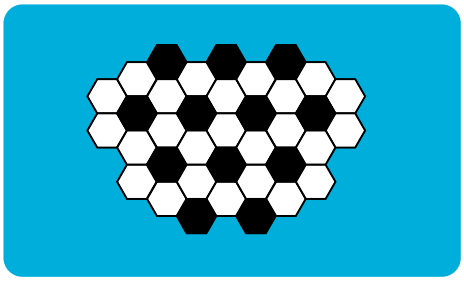
Petra wants to make a classic soccer ball.

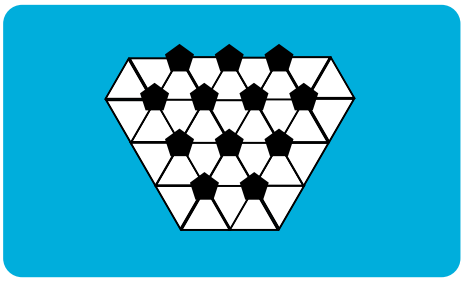
She has four different options of flat shape-based patterns that can be used.

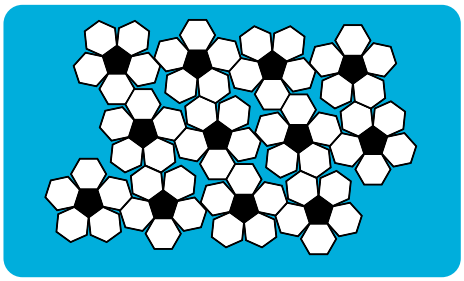
Question

With which of the below options is it possible to make a classic soccer ball?



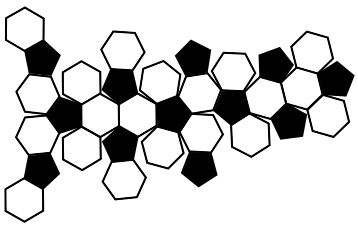






EXPLANATION

Answer

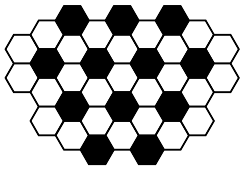


Continued on next page

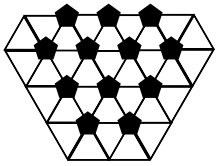
Beaver world cup ball – continued

Explanation

Looking at the image of the classic soccer ball, we can see that the white shapes are 6-sided hexagons, and all the black shapes are 5-sided pentagons. Therefore, all patterns that have shapes other than these can immediately be eliminated:

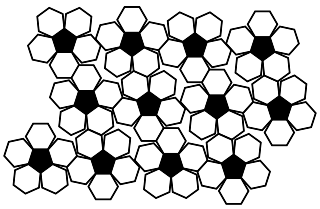


is wrong as it uses black hexagons instead of pentagons.



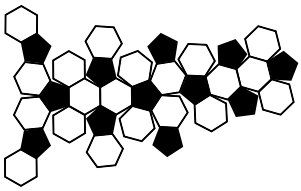
is wrong as it uses white triangles instead of hexagons.

We can also see that there is only one hexagon edge between any two black pentagons on the ball.

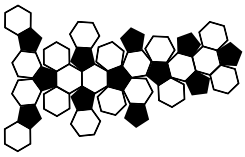


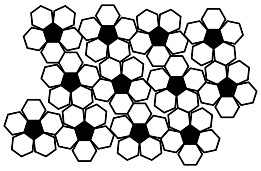
This eliminates this option, as the black pentagons are separated by at least two edges of a hexagon.

This is the only remaining option. After verifying that it indeed does follow the two rules above, we can confirm that this is the correct answer.

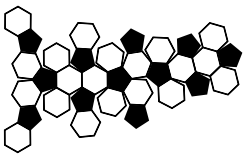


Note that the patterns on and do not fit perfectly together, as they are a design for a 3D surface transcribed onto a flat 2D sheet.





This is why it might look like there are 2 hexagons between pentagons for , when this is not the case after folding it into a ball shape.



BACKGROUND INFORMATION

Pattern recognition is one of the cornerstones in computational thinking. Recognising patterns in a problem or between problems can help us solve them with greater ease.

Apart from being a cornerstone of computational thinking, pattern recognition is also an area in the field of informatics. With pattern recognition, computers can recognise or distinguish between one object and another. In principle, the way computers recognise an object is the same as the way humans recognise an object. For example, humans can recognise cats by the characteristics they have. Computers can also determine whether an image is an image of a cat by extracting features or characteristics from the image.

Pattern recognition has helped humans in many fields. In the healthcare field, for example, pattern recognition can be used to help diagnose disease and predict risk. In agriculture, pattern recognition can be used to detect weeds and plant diseases.

Heart graphics

Tom started with one circle and one square:

Then he created this heart from these shapes:

Tom only used these kinds of operations on the shapes:

- **Rotate**: Rotate the shape by any amount in either direction.
- **Move**: Move the shape anywhere.
- **Duplicate**: Create a copy of the shape at the same location

Question

Which of the following sequence of operations could Tom have used?

Duplicate circle. Rotate square. Move circle. Move circle.

Duplicate square. Rotate square. Move square. Move circle

Duplicate circle. Rotate circle. Move circle. Move square.

Move circle. Move circle. Duplicate circle. Move square.

EXPLANATION

Answer

Duplicate circle. Rotate square. Move circle. Move circle.

Continued on next page

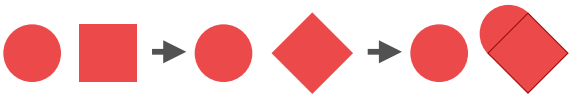
Heart graphics – continued

Explanation

The square needs to be rotated so that one corner is facing down. This only happens in sequences "*Duplicate circle. Rotate square. Move circle. Move circle.*" and "*Duplicate square. Rotate square. Move square. Move circle.*"

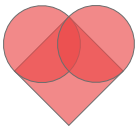
Next, you can see that you need two circles to create a heart.

Sequence "*Duplicate square. Rotate square. Move square. Move circle.*" cannot be correct because it does not create a second circle. This leaves the first one: "*Duplicate circle. Rotate square. Move circle. Move circle.*" as the only correct sequence. The below image shows this sequence after three of the four steps have been taken.



After the final fourth operation, the heart is created. The images below show the resulting shapes (made see-through).

This verifies that Tom could have used sequence "Duplicate circle. Rotate square. Move circle. Move circle."



BACKGROUND INFORMATION

When you construct images with a graphics editor, you can perform different operations. For the heart image in this task, you duplicate a circle, rotate a square and then move the circles. If you wanted a computer to do this job, you would have to add some more information. You would have to indicate where to move each circle and how much to rotate the square by. The operation sequences in the answer options are incomplete descriptions of what to do.

Computer scientists call these *abstractions* of commands. They give some idea of what to do, but they do not explain it in enough detail for a computer to execute. This is helpful because ignoring the details lets us focus on what is most important. When computer scientists develop a computer program, they often start with abstractions to get an idea of how the program will work, and add in the missing details later.

Filling green

Beaver Raj is painting pictures, made of different shapes, on a computer.

Raj can paint any of the shapes. First he selects one of four paints, then he clicks inside the shape to change its colour.

For example, Raj could paint this 'face'  completely blue with just two moves:

- Select yellow, then paint the red part of the face and get:  the entire face is now filled yellow.
- Select blue, then paint the yellow part of the face and get:  the entire face is now filled blue.

Raj makes a game of painting a picture with the *least number of moves*.

Question

Colour the pattern below so it is completely filled with green triangles in the least number of moves possible.

EXPLANATION

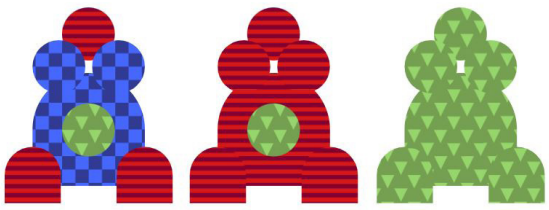
Answer

The picture can be painted completely green in three moves.

Explanation

The initial drawing contains four colours. Only one colour can be changed at a time. The minimum number of moves is three, as per below:

1. Yellow to blue: We got rid of the yellow colour and made a single blue part.
2. Blue to red: We made one single red part.
3. Red to green: We are done!



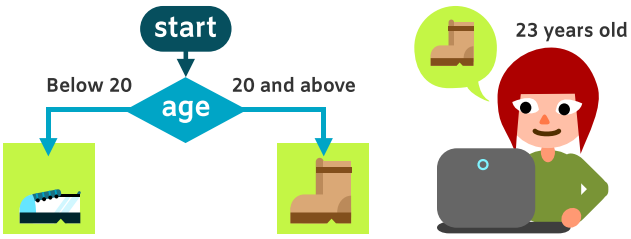
BACKGROUND INFORMATION

The goal of the task is for students to transform a figure, by interpreting the information and rules contained in the question. They are also required to follow these instructions in an optimised manner.

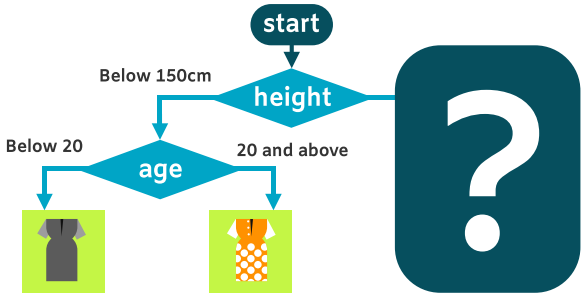
This can serve as an introduction to the kind of thinking required when writing programs using procedural programming languages. At its core, *procedural programming* involves creating a list of instructions for how to solve a problem, using logical steps.

Recommendation system

An online fashion shop recommends clothes based on the customer. For example, when a customer who wants to buy shoes puts in their information, the software follows the rule shown in this chart and recommends a pair of boots.



One day, parts of the rules are erased! The chart below shows the remaining part of the rules:



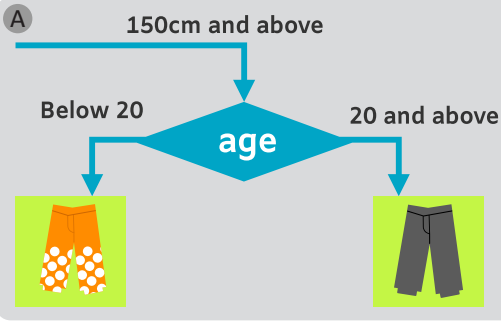
Luckily, records of past recommendations are stored on the server in the table below:

Customer	18y 140cm	32y 145cm	28y 155cm	15y 160cm	10y 152cm
Recommendation	plain skirt	patterned skirt	patterned trousers	plain trousers	plain trousers
					

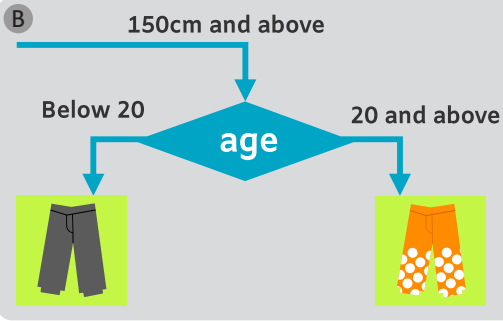
Question

What does the missing part of the system's flow chart look like based on the information in the table above?

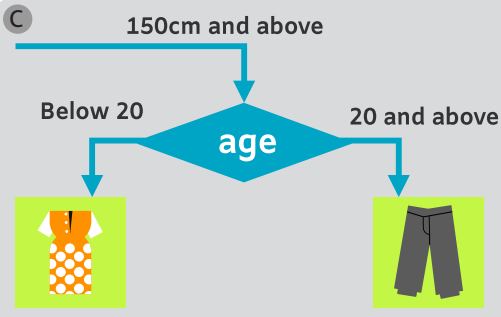
A



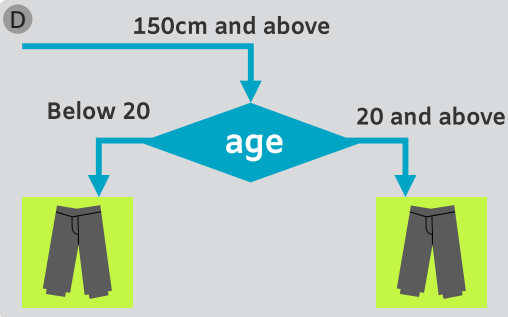
B



C



D



Continued on next page

Recommendation system – cont'd

EXPLANATION

Answer

The correct answer is B.

Explanation

The first condition checks a customer's height. Customers with a height below 150cm are already listed, so all we need to consider are customers with a height above 150cm.

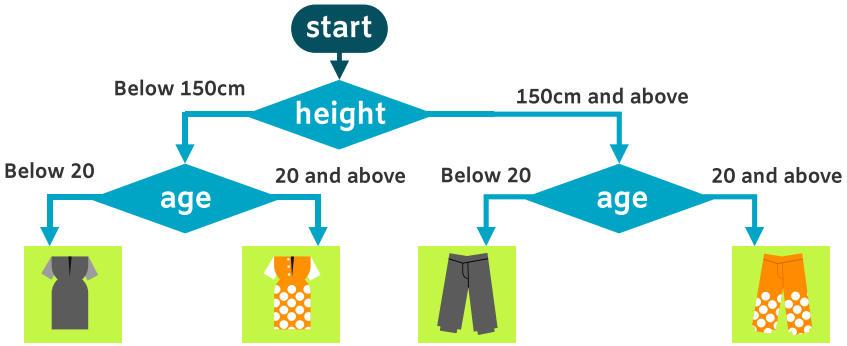
We can see from the table that all of these customers were recommended to buy trousers. Therefore answer D is incorrect.

Because all of the customers younger than 20 were recommended plain trousers, the answer A is incorrect.

Because all these customers of age 20 and above were recommended patterned trousers, the answer C is incorrect.

Customer	28y 155cm	15y 160cm	10y 152cm
Recommendation	patterned trousers	plain trousers	plain trousers
			

We can see that customers get different trousers recommended to them based on their ages. Therefore, the completed chart should look like the figure below:



BACKGROUND INFORMATION

The chart which the software follows is called a *flowchart*. In computer science, a flowchart can be used to represent an algorithm, a workflow, or a process. A flowchart is comprised of different shapes connected by arrows. The shapes represent different type of actions, and the arrows represent the order of executing those actions.

Because flowcharts show the process of programming, engineers often used them as a guide while designing a program or showing how to get certain results. In our daily life, some people use flowcharts to make decisions, such as deciding what to eat for lunch based on different preferences. Some use them to formulate an emergency procedure according to different situations, and some others use them as a guide for explaining how certain plans should be carried out.

Dominoes

Angus and Penny are playing a game with dominoes.

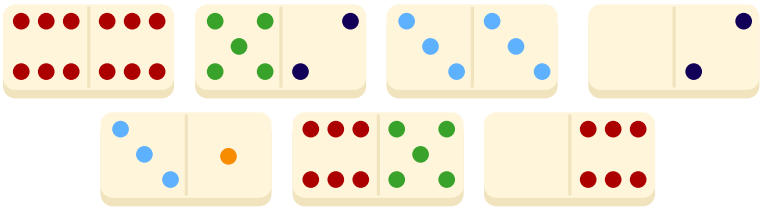
They make rows of tiles by placing them end-to-end, so long as the touching ends have the same number of dots. A player is allowed to flip the domino tiles around to achieve this.

The row above could be increased to 3 dominoes by:

- adding another domino with six dots on one side, to the right.
- adding another domino to the left, that has one side with no dots.

Question

Given the seven domino tiles below, what is the longest row of tiles we can make?

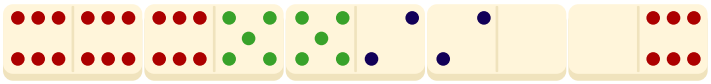


1 2 3 4 5 6 7

EXPLANATION

Answer

The longest achievable row of domino tiles is 5. There are multiple ways to achieve a row of 5. One way is shown below:



Explanation

The two remaining domino tiles with 3 dots on them can be attached to each other, but there are no other domino tiles with 3 or 1 dot remaining to join them to. Using these tiles would give a maximum row length of 2.

This means that the maximum row length that we can achieve is 5 dominoes, and cannot be extended further.

BACKGROUND INFORMATION

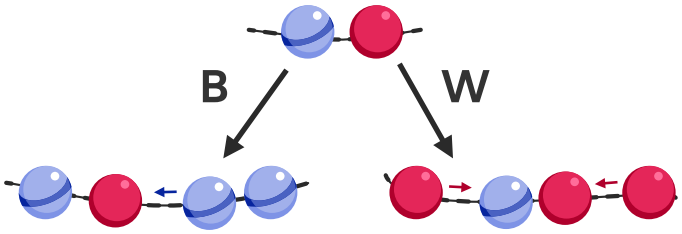
Each domino is an object that has carefully defined properties. Objects with carefully defined properties can be represented well in computer programs. There are also rules for how the dominoes can be aligned next to each other. That is, the object's behaviours are carefully defined too. This is similar to many scenarios we encounter in computer science. In fact there is a whole programming paradigm that makes programming in this way much easier: *Object-oriented programing* or *OOP*.

This problem is similar to a common situation computers have to deal with called sorting, where a collection of things need to be put in order according to some set of rules. The fact that dominoes can be flipped also adds an interesting consideration.

Sailor necklace

Sailors make necklaces using these instructions:

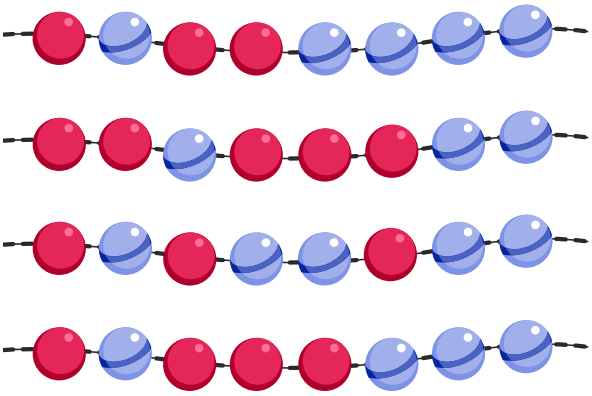
1. Use only blue striped beads and solid red beads.
2. Every sailor necklace starts by placing a blue striped bead next to a solid red bead on a string. The blue is always on the left and the red is always on the right.
3. The sailor necklace can then be made longer by either:
   - Adding two blue striped beads to the right end of the string.
   - Adding a solid red bead to each end of the string.



These two actions can be done many times in any order to build longer and longer necklaces. The necklaces cannot be rotated.

Question

Which necklace below is NOT a sailor necklace?



EXPLANATION

Answer

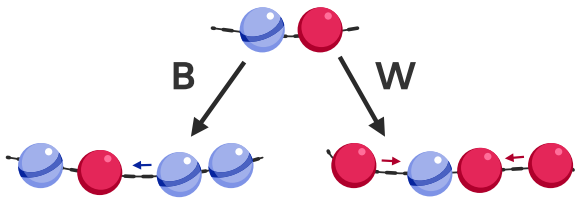
Continued on next page

Sailor necklace – continued

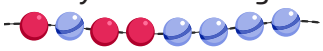
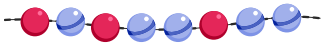
Explanation

There are many ways you could have solved this task.

We will name action one “B” and action two “W”.



Here are three possibilities:

4. You could have constructed each necklace by first locating the two starting beads and then perform a series of B and W actions. Necklace  can be made starting with the second and third beads, and then performing actions W-B-B.
   Necklace sailor  can be made starting with the third and fourth beads, and then performing the actions W-W-B.
   Necklace sailor  can be made starting with the second and third beads, and then performing the actions B-W-B.
   However, if you look at necklace  , the starting beads must be the second and third beads. You can perform action W once, but then there are no actions that will construct the rest of the necklace.

5. This approach would not work well if the necklace was very long and had many possible starting beads. In this case, you could instead try a deconstructive approach where you repeatedly remove beads by reversing the B and W actions until you only have two starting beads left.

6. A third strategy is to notice something about the parity of the beads. According to the sailor necklace instructions, there will always be an odd number of solid red beads and an odd number of striped blue beads. Necklace  has an even number of both types of beads and therefore cannot be a sailor necklace.

BACKGROUND INFORMATION

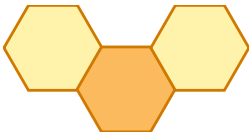
In this task, you could only add beads to the ends of the necklace string. You could not insert a bead into the middle. Similarly, if you wanted to take the necklace apart, you would have to remove beads from the ends of the necklace string. You could not easily remove a bead from the middle. This type of structure where you can easily add and remove from the ends but not from the middle is called a *double-ended queue* or *deque* in computer science.

Deques can be used to store a browser’s history, to schedule print jobs, and also to verify the validity of mathematical expressions. Checking for matching brackets can be done in a way that is quite similar to how you verified the validity of the sailor necklaces!

Beehive

Can you help put the bees in their hive?

Below each bee is a rule they must follow, shown as a section of the hive. The space the bee must be in, is represented by the darker orange cell.



For some bees, there may be multiple positions that obey their rule.

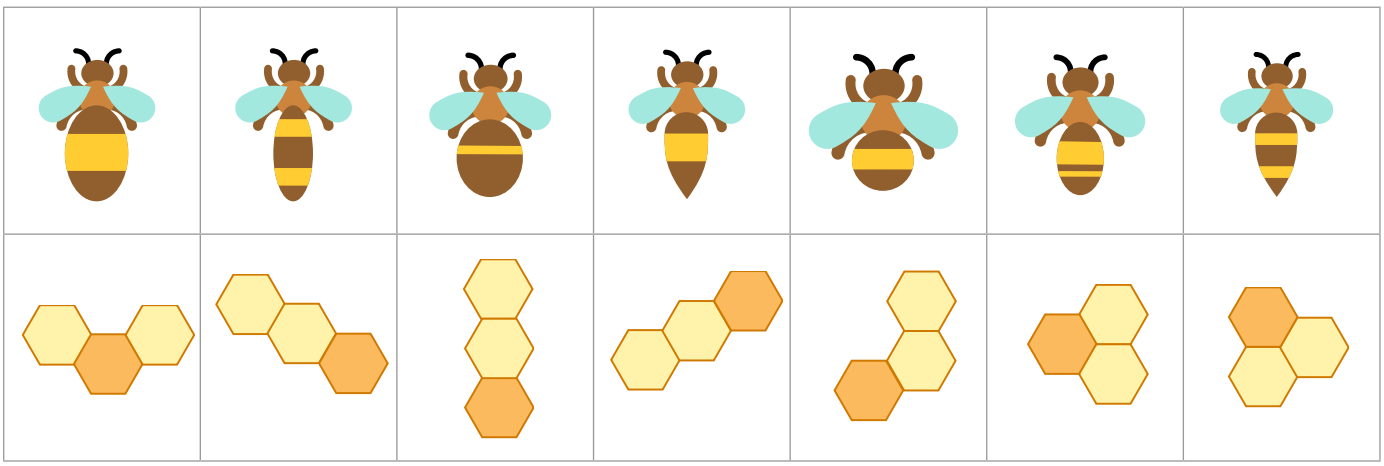
To help the bees understand the rules, a cell lights up green if a bee is dropped there and is obeying its rule. When a bee is not obeying its rule, the cell lights up red.

bee is allowed here

bee is not allowed here

Question

Place all the bees into the hive, obeying their rules.

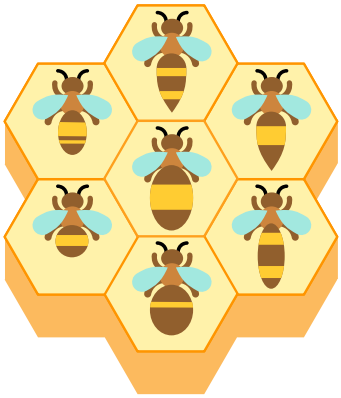


Continued on next page

Beehive – continued

EXPLANATION

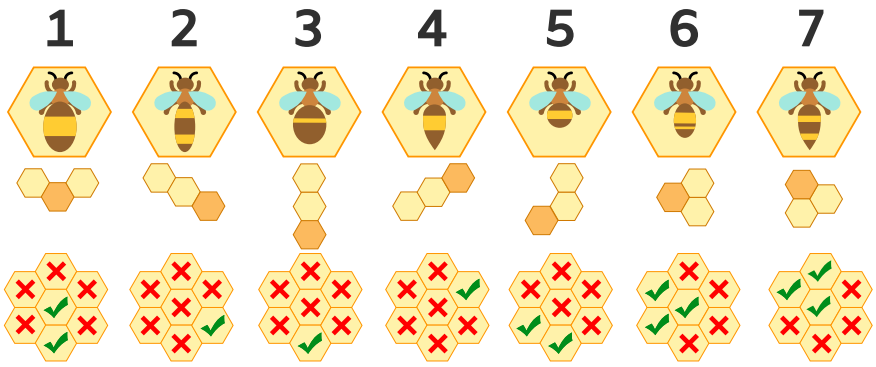
Answer



Explanation

You can solve the task just by trying it out, but this can take a lot of time. A quicker way to solve this task, is by taking a closer look at the bees' rules. In the following image, you see each bee and its rule. The cells, in which the bee can be placed according to its rule, are coloured green.

You see that some bees can be placed in only one cell of the hive and others can be placed in different cells. For example, bee 2 can only be placed in one cell, because there is only one way to place the bee's part of the hive in the complete hive.



To solve the task, you proceed like this: First place the bees for which only one placement is possible, that is to say bees 2, 3 and 4.

There is then only one possible place left for bees 1 and 5. In the same way, you place in order bee 6 and finally bee 7.

BACKGROUND INFORMATION

The key for solving the task is proceeding in the right order, which greatly reduces the number of possible solutions. In this case, we started by dealing with the most constrained elements to limit the number of cases to be explored.

Finding a solution for a problem is like finding a path through a graph that connects the starting position with the solution. The *nodes* represent states, and the *edges* of the graph (shown as lines between nodes) represent decisions. Some problem-solving strategies try to find measures that will reduce the complexity of the path, some strategies focus on how the path is traversed. For example, sometimes it is easy to find steps leading to a solution by working backwards from the solution to the starting point.

Sewing machine

Havel has a programmable sewing machine. The machine can sew a + stitch or a × stitch. It can also move the fabric forward 1 stitch space.
The sewing machine can sew both of the stitches in the same place (in any order), to get a ✳ stitch.
The program for the machine consists of a sequence of characters +, x and →.
When sewing fabric, the machine keeps repeating the program while the pedal is held down.

For example, running the program + → + x → x → will sew this pattern:

+✳×+✳×+✳×+✳×+✳×+✳×+✳×+

Question

Which program did Havel enter into the machine to get the following pattern?

××✳×✳××✳×✳××✳×✳××✳×✳××

- + x → + x → x → + x → x →
- x → x → x + → x → + x →
- x → x → x + → x → x →
- x → + x → x + → x → + x → x → x →

EXPLANATION

Answer x → x → x + → x → + x →

Explanation

Looking at the program, we see that the repeating pattern is ××✳×✳.
The first stitch is ×, after which the fabric should be moved, so the program starts with x →.
The same instruction should be used for the second × stitch.
Then we have to sew a star - which can be done by sewing x and + on the same place (the order does not matter) and after this the fabric is moved. To do this, we add x + → or + x → to our program.

Another cross is "made" by commands x → and another star by the + x → or x + → commands.
So the whole program is x → x → x + → x → + x →.

Option A: x → x → x + → x → + x → x → x → is incorrect, because it sews a sequence of ××✳×✳×× which is not Havel's pattern.

Option B: x → x → x + → x → x → is incorrect, because the resulting pattern is ××✳×× , which is not the pattern Havel was aiming for.

Option C: x → x → x + → x → + x →

Option D: + x → + x → x → + x → x → is also incorrect. The beginning of the program + x → + x → will sew ✳✳ which is not in Havel's pattern.

BACKGROUND INFORMATION

A *program* is a list of instructions in a programming language. Following instructions is a very important concept in computer science. The order of instructions is very important. By changing the order, we can change the output of the program.

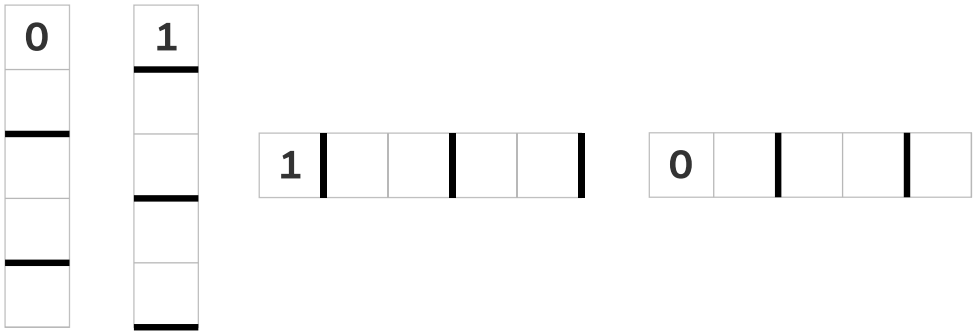
A different sequence of commands produces a different sequence of stitches - which means different patterns can be sewn.

Note that the "program" (pattern) in this task is a pattern that will give different results when executed differently. In this case, the pattern is run only while the pedal is pressed.

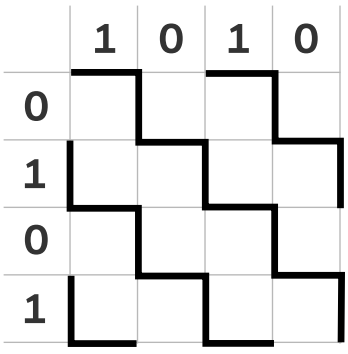
Grid paper pictures

Beaver Simon likes making little pictures on grid paper. He writes an 8-digit binary code (using 0s and 1s) to create each picture. Simon breaks the 8-digit binary number into 4 digits across and 4 digits down. He then follows these rules:

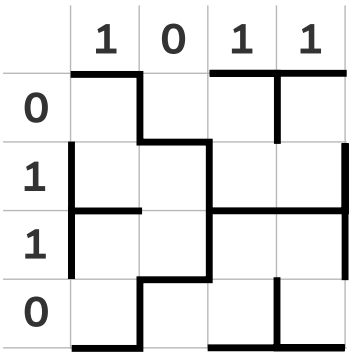
1. If a row or column starts with a 1, draw a line, then skip 1 line, then draw a line again etc., as shown below.
2. If a row or column starts with a 0, skip a line, then draw a line, then skip a line again etc., as shown below.



After each line has been drawn as per the rules for every row and column, Simon is left with a little picture. For example:



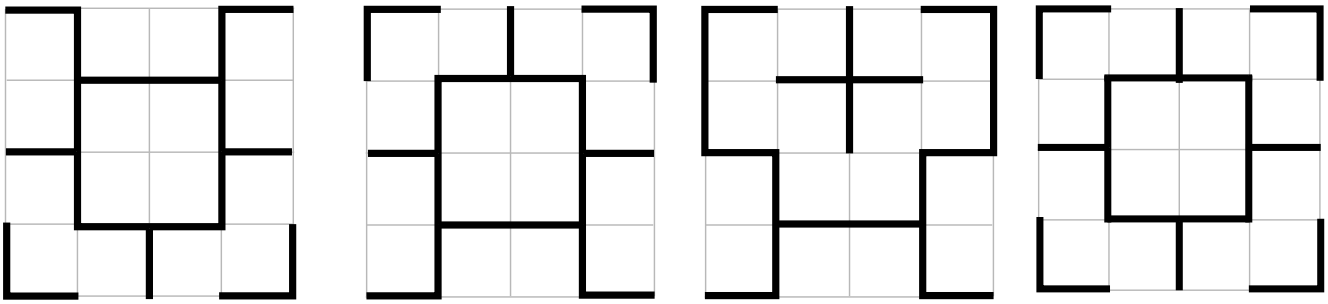
This is the picture created by the code 1010 across and 0101 down.



This is the picture created by the code 1011 across and 0110 down.

Question

Which of these pictures did Beaver Simon make using the code 1001 across and 1000 down?

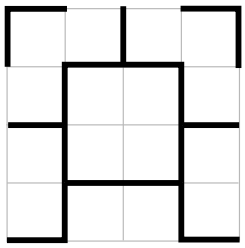


Continued on next page

Grid paper pictures – continued

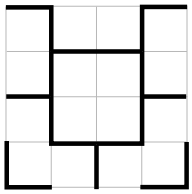
EXPLANATION

Answer

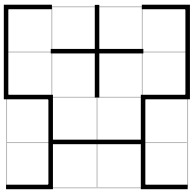


Explanation

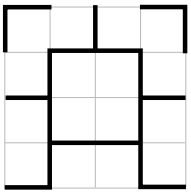
Drawing out the picture created by the code 1001 across and 1000 down in its entirety will lead to finding the right option. However, the correct answer can be found much faster just by checking what the numbers down would be for each of the four options.



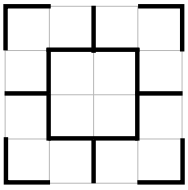
is incorrect as the rows start by skipping, skipping, skipping, and drawing a line respectively, corresponding to 1110 down.



is incorrect as the rows start by drawing, drawing, skipping, and skipping a line respectively, corresponding to 1100 down.



is possibly correct as the rows start by drawing, skipping, skipping, and skipping a line respectively, corresponding to 1000 down.



is incorrect as the rows start by drawing, skipping, skipping, and drawing a line respectively, corresponding to 1001 down.

Given that all other answers are incorrect, we are left with  as the only possible answer.
We can then further verify that this answer is correct by checking the number across:

The columns of  start by drawing, skipping, skipping, and drawing a line respectively, corresponding to 1001 across. Therefore,  is indeed the correct answer.

BACKGROUND INFORMATION

A number system that only uses 1s and 0s is called a *binary* number system. Binary numbers form the basis of how computers operate. The most basic unit of information in computing is called a *bit*, which comes from 'binary digit', which is a logical state of two possible values – in this case, a 1 or a 0. Eight bits together, as seen in this question, form a unit of digit information called a *byte*. Traditionally, a single byte would encode a single text character in a computer.

While a binary number could be represented as eight digits (such as 10011000), it could also be represented as a picture that follows a set of rules, as shown in this question. This is an example of *data representation*, the way in which data is stored, processed, transmitted and visualised. Computer scientists often make decisions on how to go about doing this - sometimes a visual representation of data may be more helpful to the user than a text representation, and vice versa.

Lights on

Beaver Sofia and her group of friends are playing a game called 'Lights On'.

The game has 8 large buttons which, when stood on, send a signal down a wire.

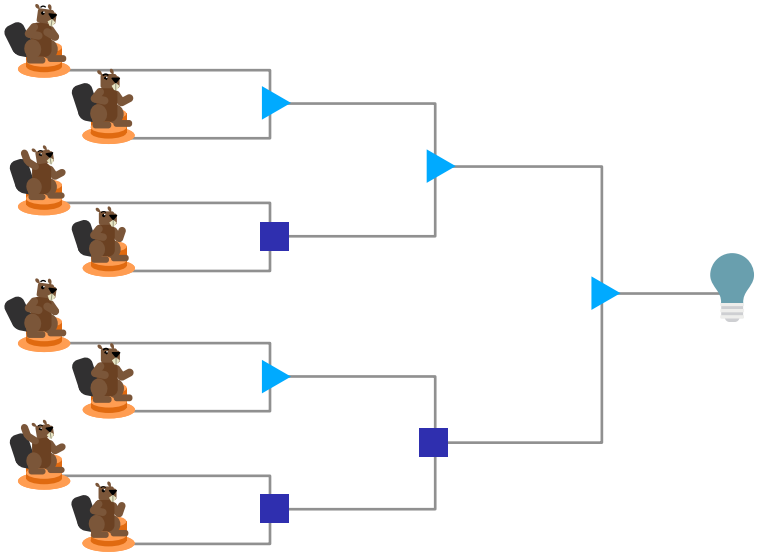
These wires pass through some triangular or square boxes before eventually reaching a light bulb.

- A triangle will only send a signal if BOTH incoming wires send it a signal.
- A square will send on a signal if exactly ONE of the incoming wires send it a signal.

The friends win the game if they can turn the light on.

Question

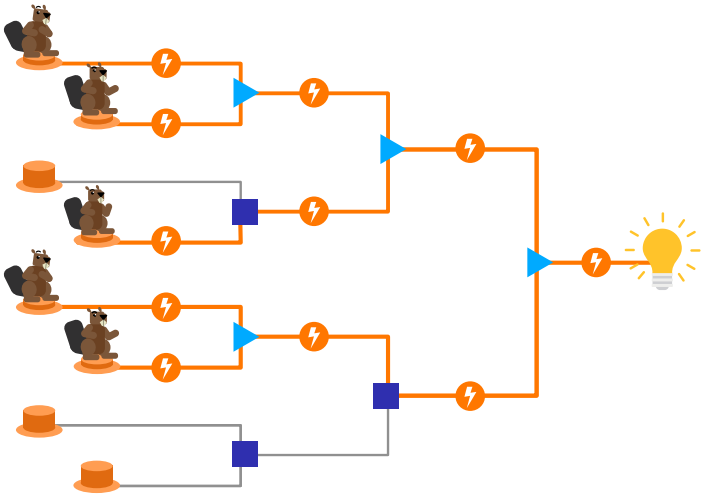
Select which buttons the beavers could stand on to turn on the light and win the game.



EXPLANATION

Answer

There are 16 possible combinations of the 8 buttons that will turn the light-bulb on.
One is shown below:



Continued on next page

This question comes from Australia

Lights on – continued

Explanation

A good approach to this question is to work backwards. The light-bulb at the end is connected to a wire in column 4 that comes from a triangle. For this wire to be ON, we know that the two wires leading into the triangle must also be ON.

These wires are connected to a triangle and a square.

We know this triangle must be ON, therefore the wires connecting to it must both be ON.

We know the square must be ON, therefore one of the wires connecting to it must be ON, and the other must be OFF.

For columns 1 and 2, we will look at the wires in the top half and bottom half separately.

Top half:
Both wires in column 2 must be ON. Therefore, the top 2 buttons in column 1 must be pressed, and exactly one of the buttons of the bottom 2 buttons must be pressed.

Bottom half:
Given the square is in the ON position, exactly one of the wires leading into it must be ON, and the other is OFF. Therefore, either the triangle leading into it must be ON and the square OFF or vice versa. There are multiple ways of doing this - the solution shown above demonstrates one possible approach. Here, the triangle is ON and therefore the top two buttons must be pressed, meaning the square must be OFF and the bottom two buttons are either both pressed or not pressed (in this case, both are not pressed).

BACKGROUND INFORMATION

In this question, the wires can either be ON or OFF. Computer scientists say that things that can only be in two different states carry *Boolean data*. Other examples of this are binary numbers (using 0s and 1s) and checking if something is true or false.

Boolean data is very important to the working of computers. Computers are made up of billions of tiny switches called *transistors*. These transistors can either be OFF or ON, and everything a computer can do is just combinations of transistors changing between these two states.

Computers also use another important component we see in this question. The triangles that turn on only when both incoming wires are on is called an *AND gate*, and the squares that turn on only when exactly one incoming wire is on is called an *XOR gate*. These are just two of several types of gates that turn on a wire depending on whether the other wires attached to it are on or off - these gates are examples of electronic components that allow computers to use *Boolean logic*.

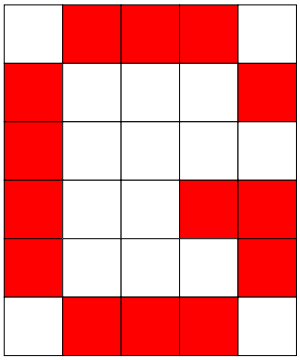
Another place that we find Boolean logic is in computer programs. Sometimes a program will need to 'make decisions' of what to do next based on if one thing (or sometimes many things) have happened before. Two common ways computer programmers put this into their programs is using *IF statements or AND statements*.

This question comes from Brazil

Turning images into numbers

Michael likes turning images into numbers. Each line can be turned into numbers using the following pattern, starting from the left:

- The 1st number says how many squares in a row should be painted white.
- The 2nd number says how many squares in a row should now be painted red.
- The 3rd number says how many squares in a row should be painted white.
- This pattern repeats until all the squares in a line are painted.



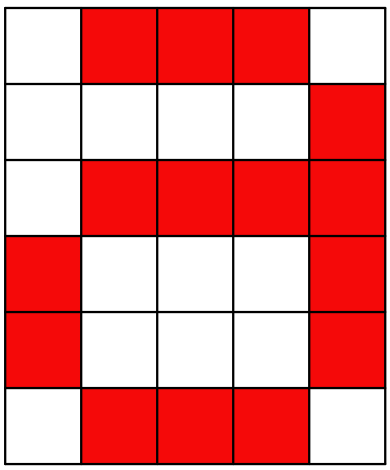
1, 3, 1
0, 1, 3, 1
0, 1, 4
0, 1, 2, 2
0, 1, 3, 1
1, 3, 1

The sequence of numbers is shown on the right of the image. Because there are 5 squares in each line, the sum of each individual line of numbers is equal to 5.

Finally, we can join all the numbers in each line into a single sequence. Michael's example above would make the following sequence:
1, 3, 1, 0, 1, 3, 1, 0, 1, 4, 0, 1, 2, 2, 0, 1, 3, 1, 1, 3, 1.

Question

What is the sequence of numbers that describes the image below?



0, 1, 3, 4, 1, 1, 3, 1, 0, 2, 2, 1, 0, 1, 3, 1, 2, 2, 1

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

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

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

Continued on next page

Turning images into numbers – cont'd

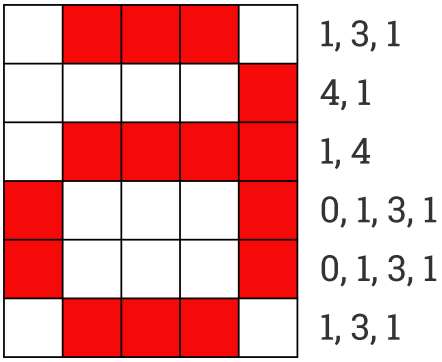
EXPLANATION

Answer

The correct answer is: 1, 3, 1, 4, 1, 1, 4, 0, 1, 3, 1, 0, 1, 3, 1, 1, 3, 1

Explanation

As the second and fourth sequences are fairly similar, in order to find the right answer, it is necessary to convert up to the fourth line of the image to numbers, and then join them:



The other three answers would represent other images, as there is a unique representation for each sequence choice.

BACKGROUND INFORMATION

In order to store or transmit images between electronic devices, it is necessary to convert them into numbers. There are many ways of doing this and this task presents one of them, known as *run length encoding*. By using a single number to represent a sequence of squares of the same color, this method uses *data compression*, an important field in computer science. The method described in this Bebras question was used by fax machines to transmit the content of documents.

This method can be improved in many ways, for instance, by allowing a number to refer to a sequence of squares spanning more than one line.

Finding ways to represent data has been a challenge since the very first electronic machines, because the choices involved may impact the time necessary to process or send and receive the information. Currently, this is still an issue in particular on the internet; creating new ways of encoding images, videos and other media files can speed up our browsing experience.

Bombom's message

Bombom passes food orders to Lala by flashing a leaf-shaped lamp and a flower-shaped lamp.

Each dish has its own code made from these two shapes. Bombom turns the lamps on and off in a specific sequence, and Lala prepares the dish accordingly.

Bombom and Lala started with these codes for two of their dishes.

Menu	Code
Burger	
Fried rice	

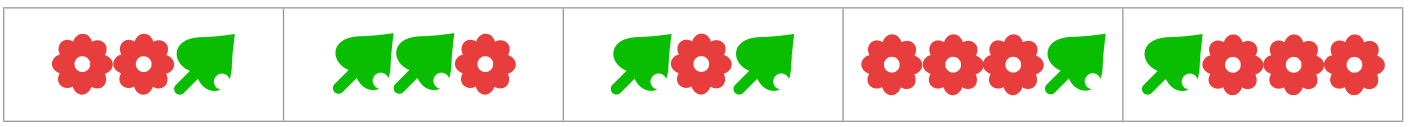
However, a problem occurs because Lala starts to prepare the dish as soon as she sees a complete code. Whenever Bombom turns on the leaf lamp twice (as part of the code for fried rice), Lala starts preparing a burger instead. Now fried rice never gets prepared!

To solve the problem, they decide to change the codes. The new codes for all the dishes on the menu are given right.

Menu	Code
Burger	
Fried rice	
Sandwich	
Pizza	
Tart	

Question

One day, the restaurant decides to add french fries to the menu. Which code can be used for french fries so that Lala won't get confused?



EXPLANATION

Answer

Continued on next page

Bombom's message – cont'd

Explanation

Lala starts to prepare the wrong dish if its complete code is identical to the first part of another code. Therefore, no item's code should include the complete code of another item at its beginning.

 is not the answer because the complete code for Burger is identical to the beginning of option  .

Menu	Code
Burger	
Fried rice	

The same applies to the code for Tart and option  and the code for Pizza and option  .

Menu	Code
Tart	
Option	
Pizza	
Option	

However,  is also not the answer because the complete code of this option is identical to the beginning of the code for Fried Rice.

Menu	Code
Fried rice	
Option	

 is the answer because the complete code of this option is not identical to the beginning of any other code and none of the other dishes' complete codes are identical to the beginning of the code of option  .

BACKGROUND INFORMATION

Lala and Bombom need an encoding rule where none of the complete codes are the first part of any other codes. Codes with this property are called prefix codes. In Computer Science, *prefix codes* are used in data compression.

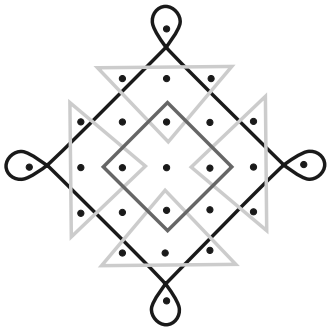
When encoding words without prefix codes, each letter is encoded by a *bit sequence* (made up of 0's and 1's) of the same length. For instance, ASCII encoding uses 7 bits per letter. With prefix codes, letters that occur more frequently are encoded using shorter bit sequences. That way, words take less bits when coded.

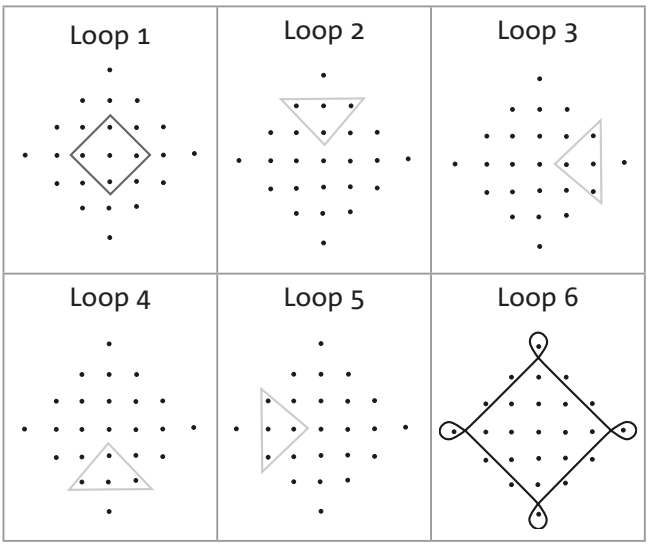
Tangled loop kolam

Kolams are decorative patterns, traditionally drawn on the floor. Dots are placed first to guide the artist, then loops are drawn around the dots creating elegant kolams.

To draw a loop, you start at a point, draw a line around a set of dots and return to the starting point without lifting your hand. The line may cross other lines, or even cross itself, but cannot trace over the same pathway as another line.

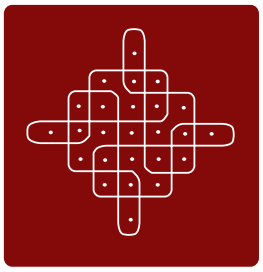
Every loop must go around at least one dot. In the kolam illustrated below, there are 6 loops. Notice that the loops cross each other at different places and the outermost loop in black also crosses itself.

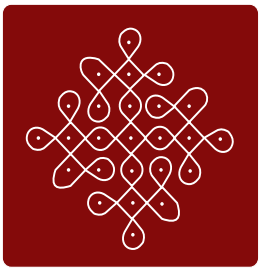




Question

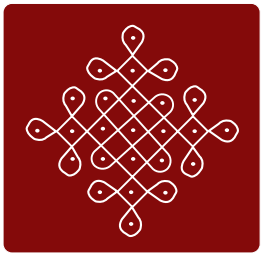
Which Kolam has the fewest loops?





EXPLANATION

Answer



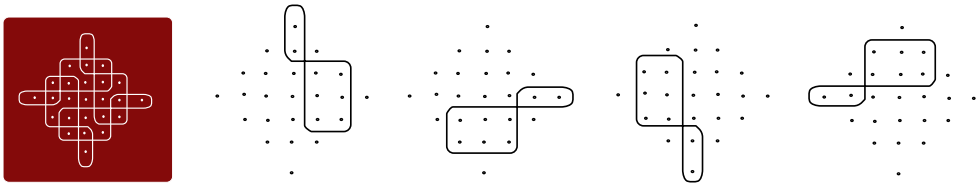
Continued on next page

Tangled loop kolam – cont'd

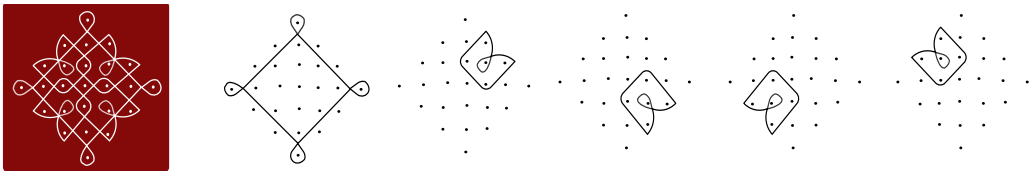
Explanation

The Kolam below has just 1 loop. The start and end point coincide.

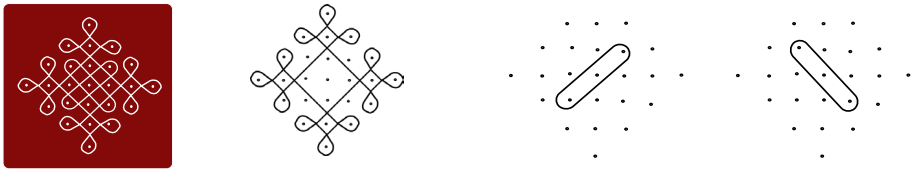
This option has 4 loops:



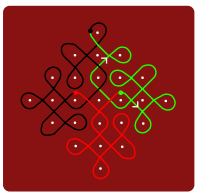
This option has 5 loops:



This option has 3 loops:



To make this clearer, the correct answer is shown below, broken into three segments (each using different colors). The dots are the starting point for each segment. We continue from the dot in the next colored line and keep tracing until we reach the point where we started from.



BACKGROUND INFORMATION

Kolams are intricate patterns found in towns and villages across India. One theory on how these patterns are remembered and reproduced is that they are generated by simple rules which are passed down from one generation to the next.

In computer science, a *grammar* is a set of rules for generating sequences of letters. Similarly, one can write more complicated two-dimensional array grammars that generate pictures like kolams. They are also related to graph rewriting where grammars are used to generate families of graphs.

Greedy trolls

Charlie wants to go from his home village with the red flag tower to the castle with the two blue flags.

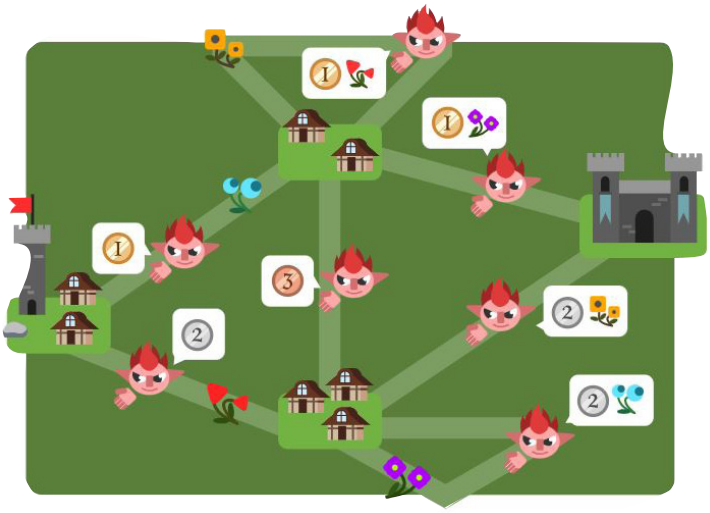
Greedy trolls on the paths demand a toll.

- All of them demand a coin (worth 1, 2, or 3)
- Some also demand a herb

The trolls are scary and impatient so Charlie fills a tube with coins before leaving. Charlie can gather herbs on his journey. He has to make sure he already has any herb a troll wants, before going down their path.

On the map below, you can see each troll's toll. Charlie will only be able to gather herbs after paying any trolls that are on that path. Charlie can take the same path twice, but he needs to pay the toll twice, too!

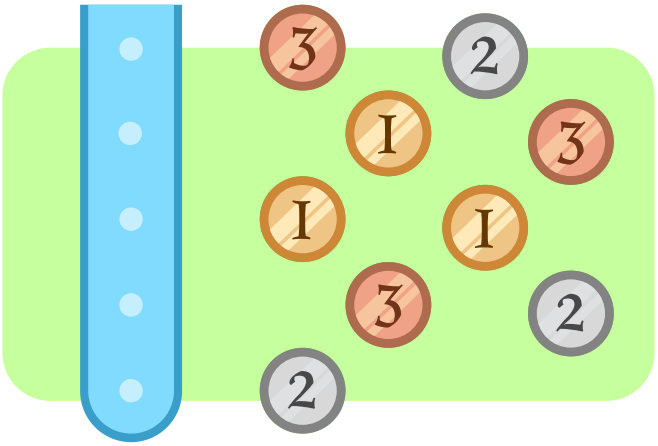
Charlie fills his tube so that he has the correct coin ready at the top for each troll he meets. He figures out he only needs five coins for his journey.



Question

Fill Charlie's tube with five coins in the right order, so that he can pay the greedy trolls on his way to the castle.

Don't forget about the herbs he must collect as well!



EXPLANATION

Answer

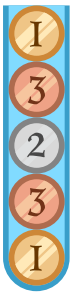
There are two possible correct answers.

Explanation

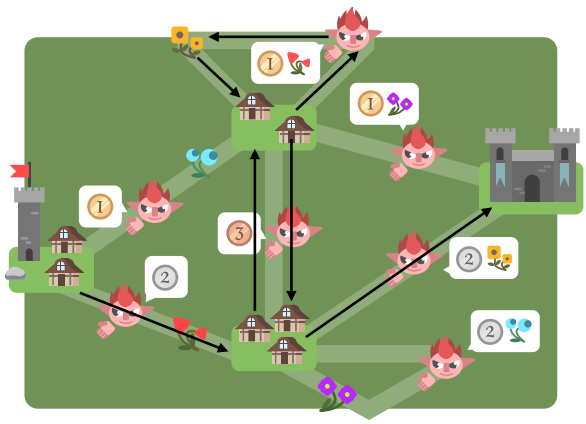
These are the only solutions because Charlie either needs or to get to the castle. To get the he needs to gather first. With the restriction of five coins, this leaves only solution 1. To get the he needs to gather first. And that leaves only solution 2. Every other way would cost him more than five coins.

Continued on next page

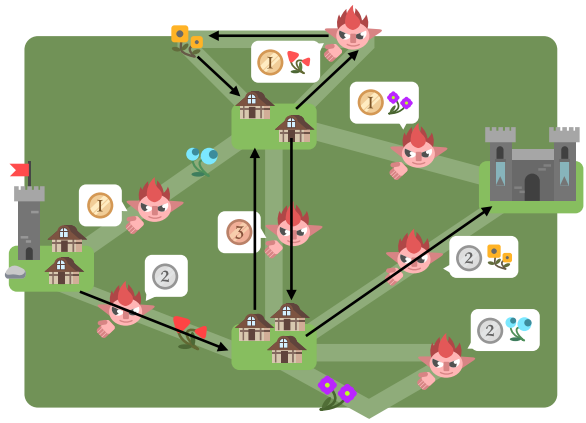
Greedy trolls – cont'd



Charlie first goes to the village with the two houses. On his way he pays 1 and collects one . Then he goes to the village with the three houses and pays 3. Then he takes a tour to collect a and pays 2 and . After that, he returns to the village with two houses and pays 3 again, to continue on to the castle, where he pays 1 and before he arrives.



Charlie goes to the village with three houses and has to pay 2 but collects a . He then goes to the village with two houses and has to pay a 3. Taking a detour, he pays 1 and but gathers . He then returns to the village with three houses and pays 3 again. After that, he can go to the castle and pays 2 and .



These are the only solutions because Charlie either needs or to get to the castle. To get the he needs to gather first. With the restriction of five coins, this leaves only solution 1. To get the he needs to gather first. And that leaves only solution 2. Every other way would cost him more than five coins.

BACKGROUND INFORMATION

The villages, the castle and the paths between them can be seen as a graph. Each path has a certain cost (the coin) and can put or take something from the coin tube, which works as a *stack*. A stack is a data structure following the last-in-first-out principle supporting the following operations: put an object on top of the stack (push) and remove the object at the top of the stack (pop).

Therefore this graph can be seen as a *push-down automaton*. A push-down automaton consists of different states (here represented by the villages and the castle), an input alphabet (here represented by the coins), a stack alphabet (here represented by the herbs), a set of transition functions (here represented by the paths), a start state (here represented by Charlie's village) and a set of accept states (here represented by the castle).

What is special is that all paths go in both directions, something that is not necessarily the case with push-down automata. Push-down automata are connected with a special class of formal coding languages (context-free languages).

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

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

Contact us

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