

Bebras Australia Computational Thinking Challenge

2023 Solutions Guide Round 2



Secondary School
Grades 7–12

bebras.edu.au

Bebras Australia Computational Thinking Challenge

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

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

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

The Bebras international community has now grown to 60 countries with over 3.9 million students participating worldwide!

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

In Australia, the Bebras Challenge takes place in 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

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

581

Australian schools participated in Round 1 2023



31,837

Australian students participated in Round 1 2023



3.29 million

Students participate worldwide



digital
careers



What is a Solutions Guide?

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

Within this Solutions Guide you will find all of the questions and tasks from Round 2 of the Bebras Australia Computational Thinking Challenge 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 a Solutions Guide?	3
What is Computational Thinking?	6
Computational Thinking skills alignment	7
Australian Digital Technologies curriculum key concepts	9
Digital Technologies key concepts alignment	10
Years 7+8	12
Jumping Game	13
Strange Payment	16
Colourful Candles	18
Forest Pictures	19
Tennis Club	21
Waterworks	23
Super Security System	25
Apple Packing	27
Miss Infinity	29
Upcycling	31
Nuts and Bolts	33
Overlapping Villages	35
Cipher 8	37
Filling Green	39
Years 9+10	41
Chez Connie	42
Stacks of Tokens	45
Cheese Snack	47
Bebras Runs	49
Antivirus	51
Seating Arrangement	53
Decorations	55
Longest Sequence	57
Secret Entrance	59
Rock Paper Scissors	61
Guess Poker Game	63
Even Teams	65
Movie Night	67
Ordering	69
Words	71

Years 11+12

You Won't Find It	73
Word Chains	74
WhatDoesItDo	75
Napping Together	76
Collecting Stones	78
Downloads	81
Popularity	84
Railway Electrification	86
Tree Farming	88
Connection of Islands	90
Perfect Partners	93
Longest Word Chain	95
Video Compression	97
Robot's Path	99
A Game of Cut and Mouse	100
	103

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 2 Questions	Grade level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Years 7+8							
Jumping Game	Easy						
Files	Easy						
Strange Payment A	Easy						
Colourful Candles	Easy						
Forest Pictures	Easy						
Tennis Club	Medium						
Waterworks	Medium						
Super Security System	Medium						
Apple Packing	Medium						
Miss Infinity	Medium						
Upcycling B	Hard						
Nuts and Bolts	Hard						
Overlapping Villages	Hard						
Cipher 8	Hard						
Filling Green	Hard						
Years 9+10							
Chez Connie 2	Easy						
Stacks of Tokens A	Easy						
Curd Snack	Easy						
Bebras Runs B	Easy						
Antivirus	Easy						
Seating Arrangement	Medium						
Decorations B	Medium						
Longest Sequence A	Medium						
Secret Entrance	Medium						
Rock Paper Scissors	Medium						
Guess Poker Game	Hard						
Even Teams	Hard						
Movie Night	Hard						
Ordering C	Hard						
Words	Hard						

Computational Thinking skills alignment

2023 Round 2 Questions	Grade level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Years 11+12							
You Won't Find It	Easy						
Word Chains	Easy						
Whatdoesitdo B	Easy						
Napping Together B	Easy						
Collecting Stones C	Easy						
Popularity	Medium						
Downloads	Medium						
Railway Electrification	Medium						
Tree Farming	Medium						
Connection of Islands	Medium						
Perfect Partners	Hard						
Longest Word Chain	Hard						
Video Compression	Hard						
Robot's Path	Hard						
A Game of Cut and Mouse	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 2 Questions	Grade level	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Years 7+8											
Jumping Game	Easy										
Files	Easy										
Strange Payment A	Easy										
Colourful Candles	Easy										
Forest Pictures	Easy										
Tennis Club	Medium										
Waterworks	Medium										
Super Security System	Medium										
Apple Packing	Medium										
Miss Infinity	Medium										
Upcycling B	Hard										
Nuts and Bolts	Hard										
Overlapping Villages	Hard										
Cipher 8	Hard										
Filling Green	Hard										
Years 9+10											
Chez Connie 2	Easy										
Stacks of Tokens A	Easy										
Curd Snack	Easy										
Bebras Runs B	Easy										
Antivirus	Easy										
Seating Arrangement	Medium										
Decorations B	Medium										
Longest Sequence A	Medium										
Secret Entrance	Medium										
Rock Paper Scissors	Medium										
Guess Poker Game	Hard										
Even Teams	Hard										
Movie Night	Hard										
Ordering C	Hard										
Words	Hard										

2023 Round 2 Questions	Grade level	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Years 11+12											
You Won't Find It	Easy										
Word Chains	Easy										
Whatdoesitdo B	Easy										
Napping Together B	Easy										
Collecting Stones C	Easy										
Popularity	Medium										
Downloads	Medium										
Railway Electrification	Medium										
Tree Farming	Medium										
Connection of Islands	Medium										
Perfect Partners	Hard										
Longest Word Chain	Hard										
Video Compression	Hard										
Robot's Path	Hard										
A Game of Cut and Mouse	Hard										

Bebras Challenge 2023 Round 2

Years 7+8



Jumping Game

Veronica loves jumping. She has found 17 paving stones in a line and made a game plan that uses them. Veronica put a coin at one end of the line and then stands on the opposite end, facing the coin.

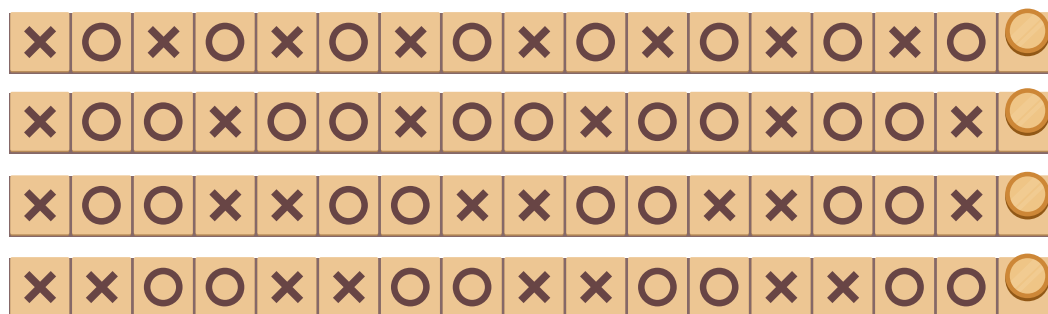


She has two jumping rules:

- If you are standing on a paving stone marked “X”, jump 3 paving stones forward.
- If you are standing on a paving stone marked “O”, jump 1 paving stone backward.

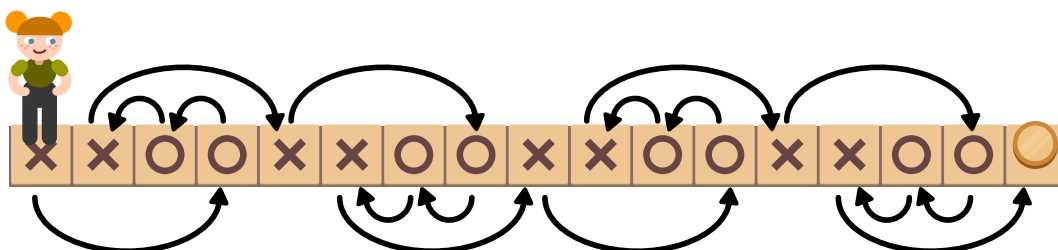
Question

Which of the game plans will bring her to the coin?



EXPLANATION

Answer



Explanation

The arrows show how to jump. Because there is only one type of forward movement and one type of reverse movement, the task has only one solution. If the girl jumped forward and left some empty tiles behind, she would not be able to return to them.

BACKGROUND INFORMATION

Finding a way of consecutive actions to achieving a goal according given rules is called algorithmisation. A robot or a computer application can be programmed so that it works according to given rules. If we then need a specific output from it, we must set a suitable input.

For example, if we have a long narrow corridor of a shape of L and our cleaning robot can only go straight, turn to the right by 90°, and then go forward, we have to put it on the end of the shorter part of the “L” otherwise it couldn’t clean the whole corridor.

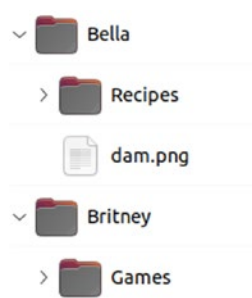


Files

Bruce and Beth Beaver are studying the text based user interface (a “command line”) on their computers.

Bruce looked at the files on his computer, both from the command line (left) as well as from the graphical view (right):

```
/Britney
/Britney/Games
/Britney/Games/dam_sweeper.app
/Britney/Games/happy_beavers.app
/Britney/diary.doc
/beaver_family.jpg
/Bella
/Bella/dam.png
/Bella/Recipes
/Bella/Recipes/cake.txt
```

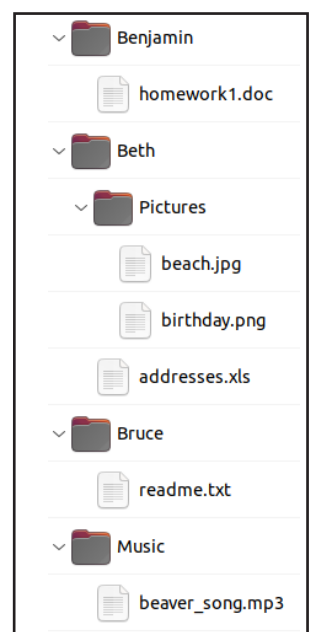
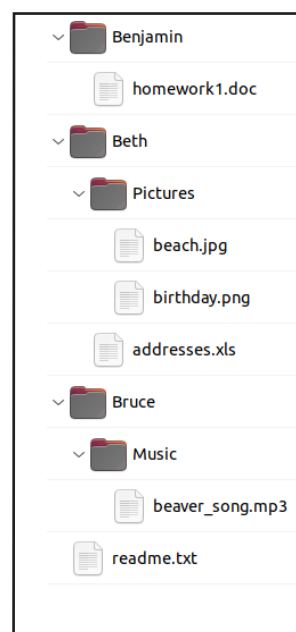
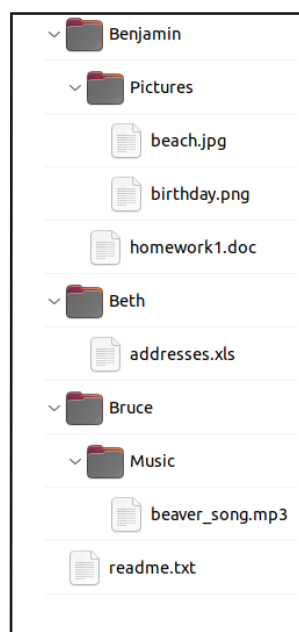
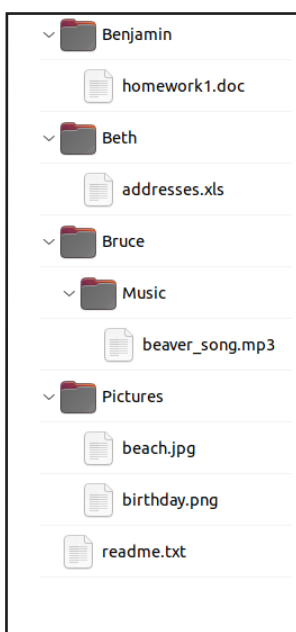


Beth also looked at the files on her computer from the command line:

```
/Bruce/
/Bruce/Music
/Bruce/Music/beaver_song.mp3
/Beth/
/Beth/Pictures
/Beth/Pictures/birthday.png
/Beth/Pictures/beach.jpg
/Beth/addresses.xls
/readme.txt
/Benjamin/
/Benjamin/homework1.doc
```

Question

Which of the following graphical views corresponds to the file listing of Beth’s computer?



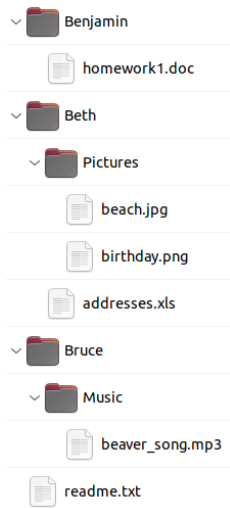


Files – continued

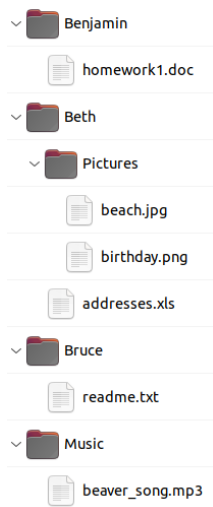
EXPLANATION

Answer

The correct answer is:

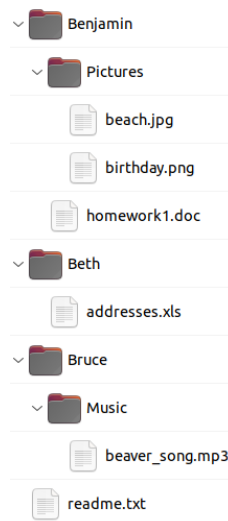
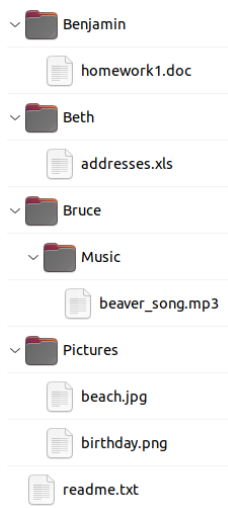


And this answer shows the “Music” folder in the wrong place.



Explanation

The following answers show the “Pictures” folder in the wrong place.



BACKGROUND INFORMATION

People may store files on a computer in a hierarchical directory structure. This means that there are files and folders. Each file is in exactly one folder, and except for the top folder (the ‘root’) every folder will also be in exactly one other folder. Note that a folder may contain multiple files and folders.

The idea of storing files in folders is actually a very traditional idea that stems from the way people used to store paper documents before the invention of computers.

Using a hierarchical way to store your files makes it easy to enforce security; you can allow access to individual folders, including all subfolders inside that folder.

Note that these days you will see other file storage mechanisms as well; quite often in a way where files are ‘tagged’, and a file can have multiple tags.

A directory structure with folders is an example of a tree structure. Trees are a very common way to organize hierarchical information in many computer science applications.






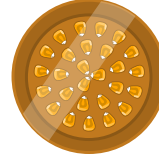
Strange Payment

Delilah wants to buy some rope to strengthen a dam. She can buy it with beavercoins at the local hardware store.

The clerk tells her she must pay 21 beavercoins for the rope.

Beavercoins are only available in the face value of 1, 3, 9, and 27 beavercoins.

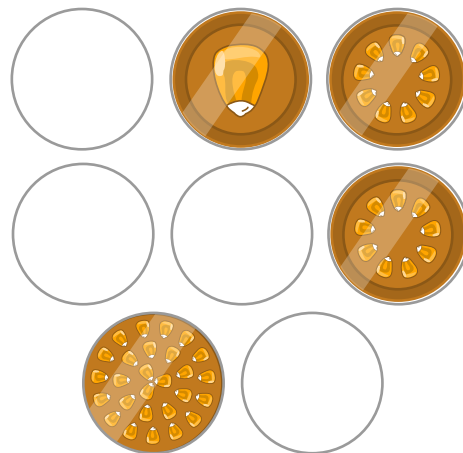
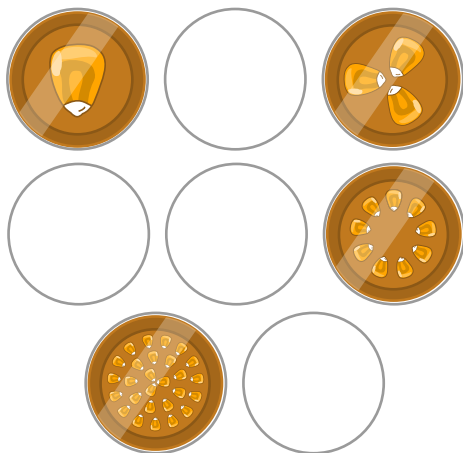
Today Delilah and the clerk only have one coin of each face value.

			
1	3	9	27

Question

Move coins from Delilah's purse (on the left) and the clerk's purse (on the right) so that Delilah has paid 21 beavercoins for the rope.

Remember to press 'Save' when you have finished.



Save

Continued on next page



Strange Payment – continued

EXPLANATION

Answer

The correct solution is that Delilah gives a 27 beavercoin and a 3 beavercoin to the clerk and receives a 9 beavercoin change. That way she pays $27 + 3 - 9 = 21$ beavercoins.

Explanation

There is no other possible solution for this problem. Each coin has three times the value as the next coin with lesser value. Delilah and the clerk only have one coin of each type. So starting from any value they may only add or subtract one coin's value each. In order to reach the value of the next coin, however, one of them would have at least two coins of the same type: only then one could give two (or more) coins to the other and receive (at most) one back. Because of this, each payment is unique and $27 + 3 - 9$ is the only solution.

Image with the correct final situation: Delilah has two coins of value 9, and one coin of value 1; the clerk has two coins of value 27, two coins of value 3, and one coin of value 1.

BACKGROUND INFORMATION

The face value of the coins in this task is chosen on purpose. Each coin has three times the value as the next coin with lesser value. That results in face values of 1, 3, 9, 27, and so forth. These numbers are integer powers of 3: $3^0 = 1$, $3^1 = 3$, $3^2 = 9$, $3^3 = 27$, ... The situation of Delilah and the clerk is that she can either give one coin to the clerk (a payment of +1, +3, +9, +27), receive one from him (-1, -3, -9, -27), or not bother with a type of coin (+/- 0). This way any value from 0 to the sum of the coins available to Delilah (40 in this case) can be represented.

This type of setup is called a *numeral system*, in this case a *balanced ternary numeral system*. The numeral system is *ternary*, because it is based on the number 3. And it is called balanced, because not only the “digits” +Coin and 0 could be used, but also the “digit” -Coin. A payment can easily be represented by writing down for any type of coin whether Delilah gives one to the clerk, no coin of this type gets moved, or the clerk gives one to Delilah. A *place-value numeration* for this numeral system could use the digits 1, 0, and $\bar{1}$. As for the decimal system also, the last digit has the least value. So in the balanced ternary numeral system, Delilah would pay $1\bar{1}01$ beavercoins.

The balanced ternary numeral system was actually used in 1958 in the Setun computer built in the Soviet Union by a team lead by Sergei Sobolev (1908–1989) and Nikolay Brusentsov (1925–2014). Although the balanced ternary numeral system has some technical advantages over the binary numeral system today virtually every computer uses, it isn't clear yet whether computers like that ever will be produced again.

Besides the historical use of the balanced ternary system in computers, this task shows that computer scientists sometimes have to combine complex theoretical concepts with practical applications in order to produce efficient computers. Today when boundaries of miniaturization are pushed further for instance when storing data using magnetism or when creating microprocessors, physics, chemistry, engineering, and mathematics have to be combined to create reliable products.



Colourful Candles

Sarah has candles in the shape of the numerals 0 to 9. There are two of each numeral.

The candles come in three colours: orange, red, and blue.

All 0-shaped candles are orange, all 1-shaped candles are red, and so on (see image).
Each year for her birthday, Sarah places candles on her cake to spell out her new age.

Today is Sarah's 11th birthday and because both candles are the same colour her family gives her an extra birthday present. She must wait three years until she is 14 before both of her candles will have the same colour again. Then there is a three-year wait until she is 17, and a further five year wait until she is 22.



Question

If Sarah uses this system from today until she is 99 years old, what is the maximum number of years she has to wait between any two birthdays where two candles of the same colour are used to spell out her age?

5

6

7

8

EXPLANATION

Answer

5 years.

Explanation

Let XY be the current age of a person where X and Y are two numerals from 0 to 9 representing the age, and X and Y are the same colour according to the table in the task body. Suppose Y is one of 0, 1, 2, 3, 4, 5, or 6. Then you must wait three years until the age $X(Y+3)$ to see two candles of the same colour.

Next, let's look separately at the three cases where Y is 7, 8, or 9:

- If Y is 7, then X is also one of the reds (1, 4, 7) and you must wait 5 years until the age $(X+1)2$ for two candles of the same colour.
- If Y is 8 then X is one of the blues (2, 5, 8) and you must wait 2 years until the age $(X+1)0$ to see two candles of the same colour.
- If Y is 9, then X is one of the oranges (0, 3, 6, 9) and you must wait 2 years until the age $(X+1)1$ to see two candles of the same colour.

Therefore, the maximum wait is 5 years. One will never need to wait 6 or more years.

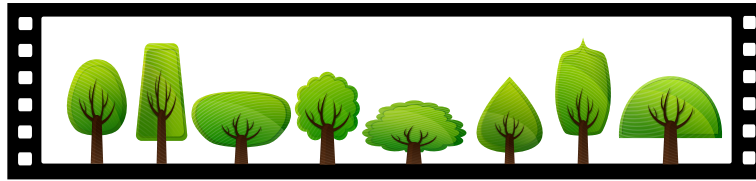
BACKGROUND INFORMATION

Quite often in computer science, we are told the rules of how a *sequence* is formed and must simulate the sequence until certain conditions are met. In this task, the sequence of two-digit decimal numbers is mapped to a sequence of pairs of colours. The n th element of this abstract sequence is the colour of the two candles used to spell out (represent) the number n .



Forest Pictures

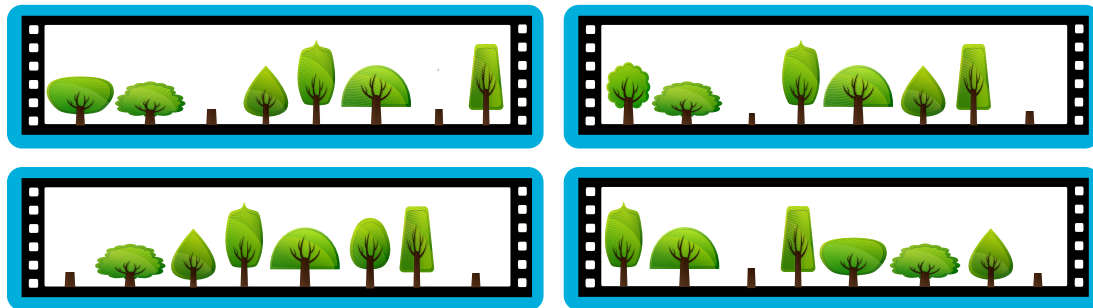
Daniel stood in the middle of a circle of eight trees and took a 360 degree photo of them.



After a few days, Daniel returned to the same spot in the forest and took another photo. He saw that two of the trees had been cut down.

Question

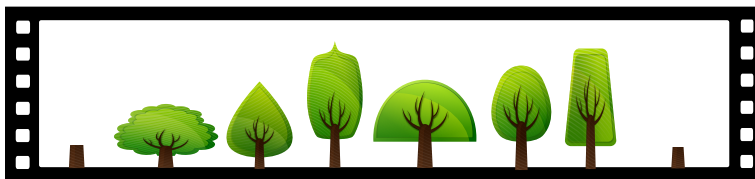
Which photo did Daniel take?



EXPLANATION

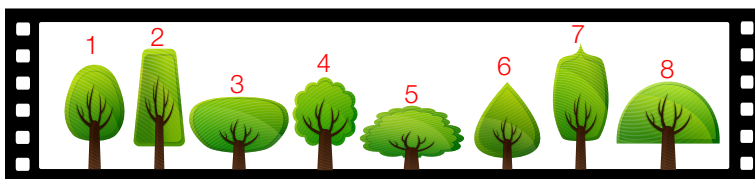
Answer

The correct answer is:



Explanation

If we number the trees in the first photo, we get this:

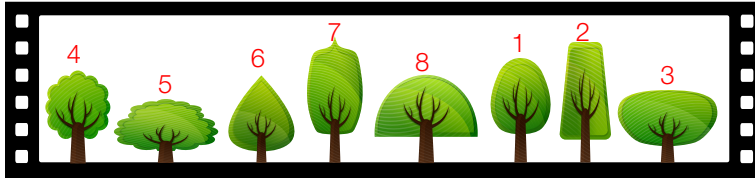




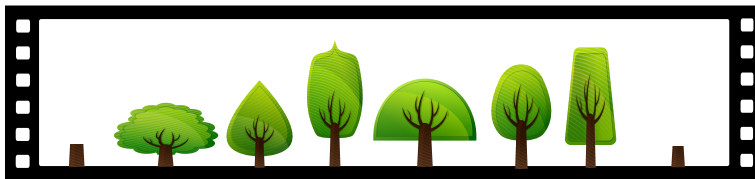
Forest Pictures – continued

If we changed the view a little bit and make the second photo, the trees would be in the same order but shifted.

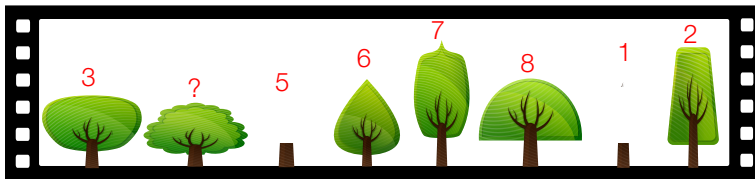
We number them so that each tree has the same number on both these photos.



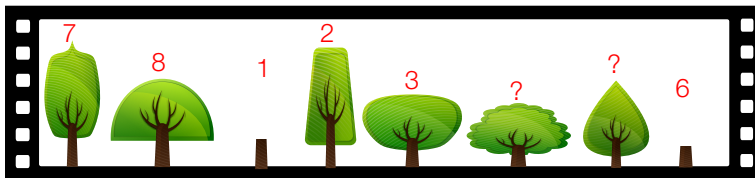
Then we will check whether all trees on the pictures below have not changed their numbers and are in the correct order. When we compare the correct answer to the second photo above, we can see that trees 4 and 3 were cut down and all the other trees are in the correct order.



In this option, the trees 1 and 5 are cut down. But the tree 4 has a different shape than in the first picture which is wrong.



In this option, the trees 6 and 3 are cut down. But the tree 1 has a different shape than in the first picture which is wrong.



In this option, the trees 1 and 6 are cut down. But the trees 4 and 5 have different shapes than in the first picture which is wrong.

BACKGROUND INFORMATION

This task involves finding the pattern the trees follow and apply that in the answer options. The ability to recognize patterns allows us to find possible errors in a sequence. Hence, in this task, the order of the trees is important. By evaluating the answer options, when we find a difference in the order of the trees, we can call it an error. This process is called identifying bugs/errors which is a vital part in debugging (in programming). When there is information missing (like the trees that were cut down), identifying the errors is more difficult.



Tennis Club

All tennis players must be registered with the National Tennis Association in order to take part in official tournaments. This is why each local club keeps track of their players and their registration information in tables, which have rows and columns.

There is one row for each player. The club will use this data to determine if any players need to renew or extend their registrations for a certain length of time.

Question

Which combination of columns is exactly sufficient (contains enough, but no extra data) to determine which players need to have their registration renewed before the next tournament?

**Player's registration number, player's date of birth, registration expiration date.
It is possible, if she turns right exactly 2 times**

Player's registration number, registration fee, registration expiration date.

**Player's registration number, date of last registration renewal, length of
registration period. It is possible, if she turns right exactly 2 times**

**Player's registration number, date of last registration renewal, player's age
category.**

EXPLANATION

Answer

The correct answer is: "Player's registration number, date of last registration renewal, length of registration period."

Explanation

From the date of the last registration renewal and the length of registration period we can calculate when the registration expires. The player's registration number tells us which player needs their registration renewed.

"Player's registration number, player's date of birth, registration expiration date" is not correct as we do not need the player's date of birth.

"Player's registration number, registration fee, registration expiration date" is not correct as we do not need to know the registration fee for this scenario.

"Player's registration number, date of last registration renewal, player's age category" is not correct as we do not need to know the player's age category for this scenario. The date of the last registration renewal alone is not sufficient, we do not know how long the registration period is and therefore cannot determine whether it has already expired or not.

Continued on next page



Tennis Club – continued

BACKGROUND INFORMATION

The world today runs on tables of data, called databases. Computer scientists need to design the structure of these tables: which columns do we need, in what formats, and how data from one table might relate to data in another table.

For example, there might a table of players' personal data, where we can find their name, age and other details related to each registration number. The registration number is the link between the row of personal data and the person's registration data. The 'column headings' are sometimes referred to as attributes.

There is an inherent trade-off with how much data is stored and we need to keep a balance. If we do not collect enough data, the table may become useless or perhaps difficult to process, if getting the answers from available data requires too much computation.

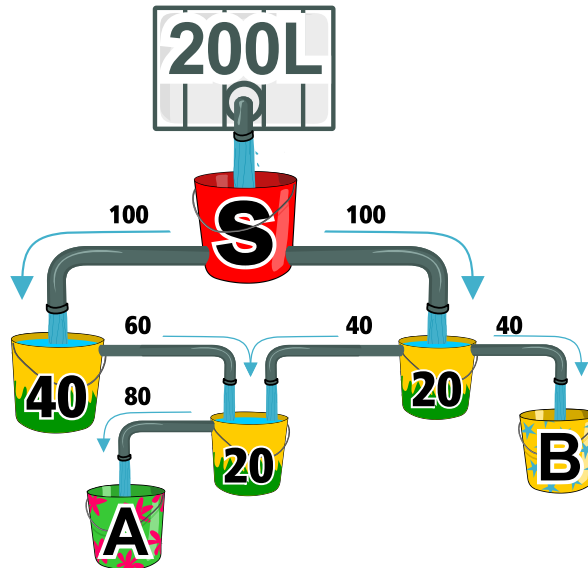
On the other hand, if we collect too much data 'just to be sure', we needlessly clutter our data storage, possibly introduce inconsistencies (imagine if the date of last registration renewal, the length of registration period and the registration expiration date do not add up) and potentially create data security issues. This is where computer scientists use methods (such as normalisation) to determine a suitable structure for data in tables.



Waterworks

We have a system of buckets, arranged vertically and connected by overflow pipes.

Each bucket collects a certain amount of water: the amount which the bucket is labeled with. After the bucket fills to the indicated level, all remaining water splits evenly between the overflow pipes leading down from it and continues to pass down the system of buckets.

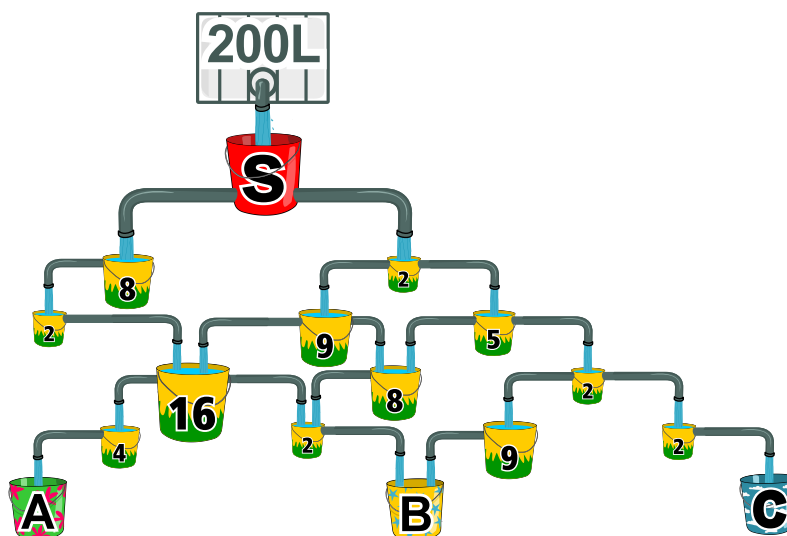


The top most bucket doesn't hold any water.

200 L of water is poured into the top most bucket (marked "S"), where it splits evenly between the overflow pipes leading down from it and continues to pass down the system of buckets to the final ones. For example, in the system of buckets to the right, 40 L of water would accumulate in the bucket "B".

Question

How much water will accumulate in bucket "A" if 200 L of water is poured into bucket "S" of the following system of buckets?



66.7 L

43 L

100 L

131 L

137 L

Continued on next page

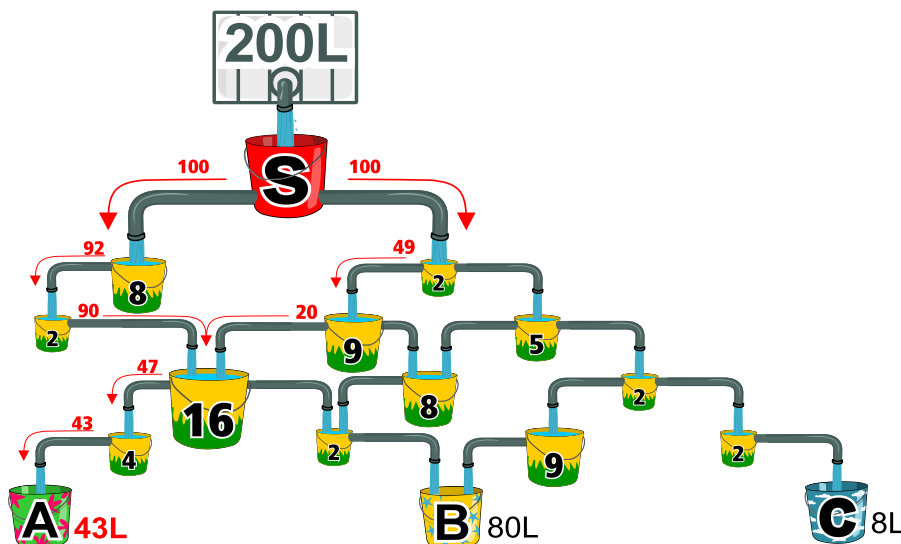


Waterworks – continued

EXPLANATION

Answer

The correct answer is 43 L.



Explanation

We could calculate every amount of water that flows from one bucket to each next bucket, but actually just the amounts on the left side (8 numbers exactly) have to be calculated.

BACKGROUND INFORMATION

The buckets and the pipes can be generalized by a graph. Other than a *flow network* for which the capacity of the pipes is limited, the pipes of this graph have an unlimited transport capacity. The buckets, however, where the pipes connect, “steal” a certain amount of water.

In general a graph consists of *edges* (in this case the pipes) and *nodes* (in this case the buckets). This graph is *directed*, that means that an edge can only be traversed in one direction. In this case this is presented by the downdraft of gravity. Because the graph has a single source for water, that node is called the *root* of the graph. The three buckets where the water ends up are sometimes called *sinks*.

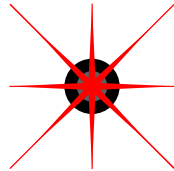
Finding the amount of water in the final buckets can be done with a variant of the aptly named *flood fill* algorithm. This algorithm simulates the filling of the buckets by putting all 200 L of water in the first bucket and puts it in a “to do” list. Then it repeatedly checks the “to do” list and as long as there are buckets in the “to do” list, it splits the amount of water that’s more than the bucket’s capacity into all buckets that it overflows into. Then the bucket is removed from the “to do” list. Depending on the order in which the “to do” list is emptied, it could happen that the same bucket ends up being put more than once on the “to do” list. For instance the second bucket from the left in the row above the named buckets gets water poured into twice. So it could happen that at first the overcapacity from the first pour is distributed and then the overcapacity from the second pour.



Super Security System

Beaver Business Banking needs a new security system. They want to secure the room in front of their safe from intruders.

They bought five detectors. Each detector sends out laser beams in eight directions until they hit an object or a wall.

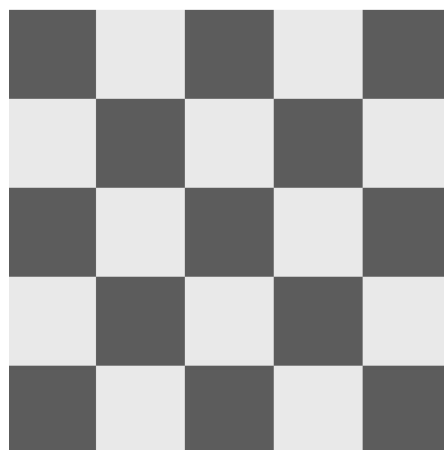
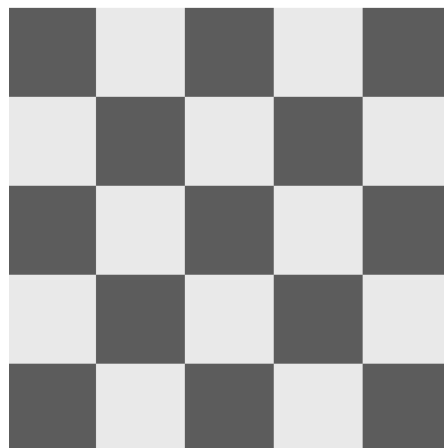


If an intruder crosses the laser beam of any detector, the alarm is activated. But if one of the detectors stands in the way of a laser beam of another detector, the alarm goes off as well!

Question

Position the five detectors on the square floor tiles, so that they secure the remaining tiles in the room, but do not activate each other.

(Remember to press 'Save' when you have finished.)



Save

Continued on next page

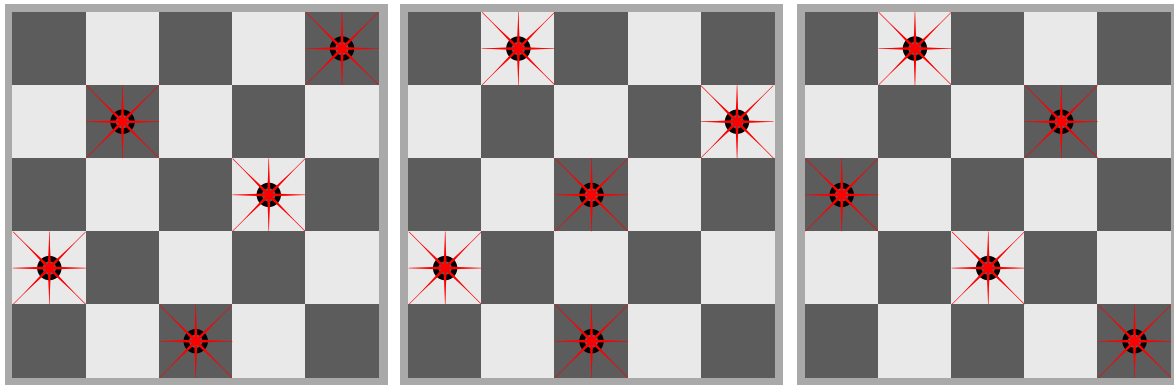


Super Security System - continued

EXPLANATION

Answer

There are multiple solutions to this problem. Here are three:



Explanation

To create a solution, we can first identify that one detector at most can be placed in each row and in each column. This is because they send out horizontal and vertical beams, and must not detect each other. But since the detectors also send out beams diagonally, care must be taken so that no two detectors will be placed on the same diagonal line.

More generally, in a correct solution, the following conditions must exist for any pair of two different detectors. x is the number of the column in which the first detector stands, and y is the number of the row of that detector. a is the number of the column in which the second detector stands, and b is the number of the row of that detector:

- x and a cannot be the same due to the vertical beams;
- y and b cannot be the same due to the horizontal beams;
- when subtracting the smaller of x and a from the larger one, and doing the same for y and b , the results of those two subtractions cannot be the same, due to the diagonal beams.

BACKGROUND INFORMATION

This problem is a variation of the eight queens puzzle, which is an example of a *constraint satisfaction problem*. You have multiple items, which can have different values, and for every item you must satisfy the values. You must do this in a way that does not violate a given set of rules.

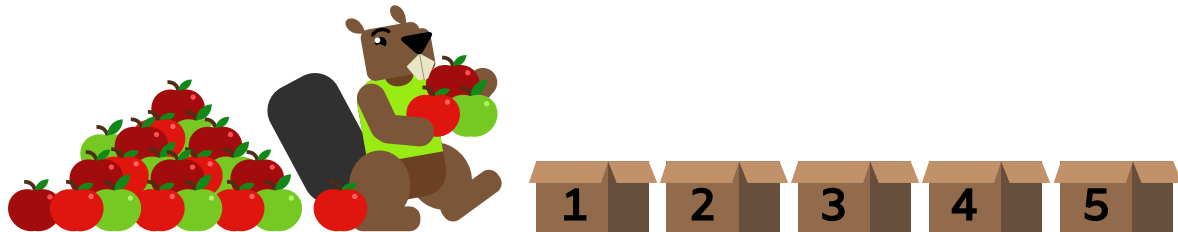
Such problems often occur in *automated planning* which is used to make robots work, in *computational linguistics* which are required for language recognition, or in *resource allocation* when businesses decide who uses certain assets. These problems can get complicated very quickly to the extent that a computer is required to solve them. Computers typically go ahead in the same way a human probably would; they try to build a solution little by little. As soon as a rule is violated, they go back one or more steps, then try the next option until they find a solution that fits all constraints. This process is called *backtracking*.



Apple Packing

A beaver went apple picking and gathered 31 apples.

He wants to put them into five boxes for storage. Afterwards, he wants to be able to retrieve any possible number of his apples by selecting a subset of the five boxes, taking all the apples from the selected boxes.



Question

How many apples should the beaver put in each box?

(Remember to press 'Save' when you have finished.)

	1	2	3	4	5
A	1	3	6	9	12
B	1	2	4	8	16
C	6	6	6	6	7
D	1	2	3	4	5

Save

EXPLANATION

Answer

The right answer is B.

Continued on next page



Apple Packing - continued

Explanation

To arrive to this answer, we must observe from the question that we need to be able to obtain any number between 1 and 31 based on adding the numbers from any selection of boxes. That is, we need a numbering system that allows us to obtain all numbers from 1 to 31 by combining 5 numbers (the ones we need to find, i.e. the number of apples in the boxes).

Since we need number 1 (one apple), we may start by putting one apple (number 1) in the first box. We also need number 2 (to get two apples), so we can put two apples (number 2) in the second box. At this point we can obtain either one (select first box), two (select second box) or three apples (select both first and second boxes). Therefore, the next number we need to obtain is 4 (to get four apples), so we can put four apples (number 4) in the third box. At this point we may retrieve one, two, three (as before), four (select third box), five (select first and third box), six (select second and third box) or seven apples (select first, second and third box). In the same manner we arrive at the full distribution of apples from the solution, i.e. having one apple in the first box, two apples in the second box, four apples in the third box, eight apples in the fourth box and sixteen apples in the fifth box. In this manner we can retrieve any number of apples, between 1 and 31, by selecting a given combination of the five boxes.

Answer A) is not correct because we cannot retrieve any number of apples using a selection of boxes. For example, we cannot obtain two apples with the given boxes.

Similarly, Answer C) is not correct because we can only retrieve multiples of 6 or 7 or a combination of these numbers.

Finally, Answer D) is not correct either, as we cannot retrieve more than 15 apples in total. In this case, the beaver did not store all the apples.

BACKGROUND INFORMATION

This task is related to the binary system, which is the most common numbering system used with computer systems. As seen in the solution, we have used the powers of two (1, 2, 4, 8, 16) to be able to represent any number between 1 and 31. This is the same as having a computer that operates with 5 binary digit numbers. That is, a computer that works only with numbers that can be represented with 5 binary digits (bits), as shown in the table of the powers of two below:

Bit	1	2	3	4	5
Power of 2 (p)	0	1	2	3	4
2^p	1	2	4	8	16

The last row (2^p) gives precisely the number of apples in each box in our solution. By adding numbers in this last row we can obtain any number between 1 and 31.



Miss Infinity



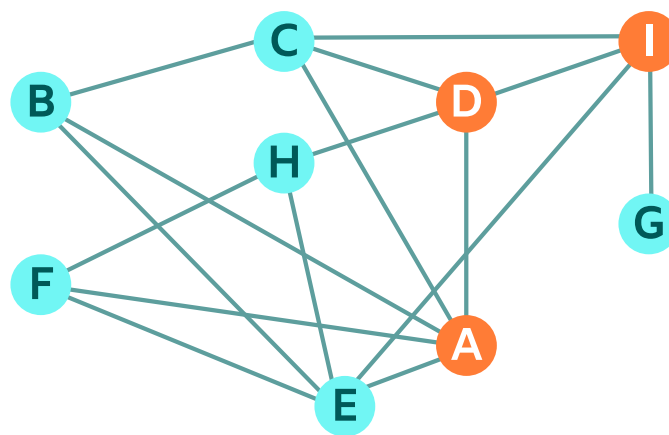
The Beaver College maths class has nine students: A, B, C, D, E, F, G, H & I .

On Monday the class got a new teacher. Three students, A, D and I , immediately started calling her “Miss Infinity”.

The diagram below shows how the students talk to each other. If they are connected by a line, they talk to each other.

For example, H only talks to D, E and F during the day. The nickname spreads among the students like this:

If a student hears more than half of the other students they talk to using the nickname, this student will also start using it the next day, and for the rest of the week.



Question

What is the first day of the week when all the students in the class use the nickname “Miss Infinity”?

Tuesday

Wednesday

Thursday

Friday

EXPLANATION

Answer

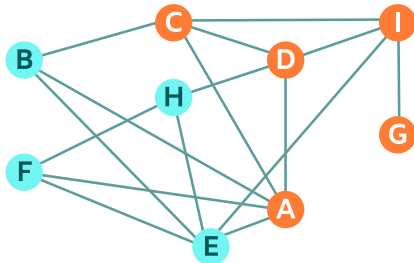
Friday.



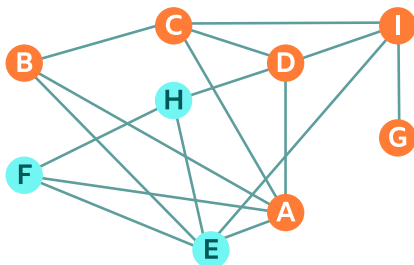
Miss Infinity – continued

Explanation

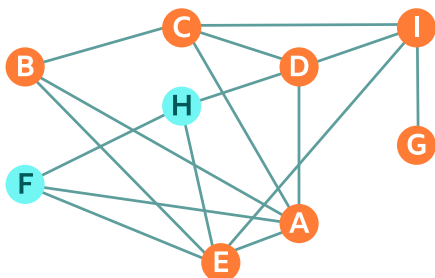
We can trace the steps by which the nickname spreads. On Tuesday, students *G* and *C* start using the nickname.



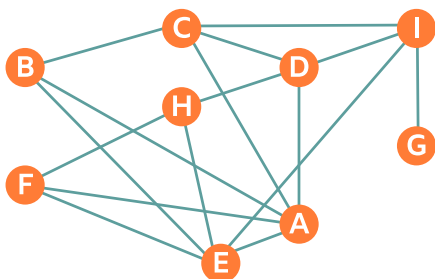
G talks to only one classmate (*I*) and this classmate used the nickname on Monday. *C* talks to 4 classmates and 3 of them used the nickname on Monday. 3 out of 4 classmates is more than half. So *C* also uses the nickname on Tuesday.



On Wednesday, *B* starts using the nickname.



On Thursday, *E* starts using the nickname.



And on Friday, *F* and *G* start using the nickname.

BACKGROUND INFORMATION

Social networks play a fundamental role in spreading information, infections, ideas, and influence among their members. An idea or innovation that appears in a social network will either cascade quickly through social interactions, or it will vanish. Maximising the spread of influence through social networks is a very active and hot trend in computer science, economics, and management science.

This task considers a special model of diffusion called the *threshold model*. In this model, there is a threshold for each individual, i.e., the fraction of their connections that must be active (using the nickname) in order for the individual to become active.



Upcycling

Beavers hate waste. They like to use old worn out things to make new useful things. This is called upcycling. The table below shows some new items, how much they can be sold for, and what is required to make them.

New items	Value	Needed to make
wheel	\$1	tyre, bag of iron
bicycle	\$10	two wheels, bag of iron
barrow	\$5	tyre, bag of wood
tricycle	\$15	wheel, bicycle

Alison loves upcycling, but has nowhere to store things.

Question

If Alison has six tyres, six bags of iron, and two bags of wood, what is the most money she can make by upcycling and selling what she makes?

(Give your answer in the form of an integer. Remember to press 'Save' when you have finished).

Save

EXPLANATION

Answer

The answer is \$30.

Explanation

It might be tempting to think that Alison should build a tricycle because this is the item that can be sold for the most total money. If she does, it requires 2 wheels and one iron bar (to make a bicycle) and another wheel. The 3 wheels need 3 tyres and 3 iron bars. So this leaves Alison with 3 tyres, 2 iron bars, 2 wood pieces.

Now, Alison no longer has the three iron bars needed to make another bicycle so she can only make wheels and barrows. Since she has the same number of iron bars as wood pieces, and barrows can be sold for more than wheels, she should then make two barrows. This uses 2 tyres and 2 iron bars, leaving 1 tyre and 2 iron bars.

It would now be possible to make a wheel, leaving 1 iron bar which cannot be used for anything else.

The total value of items made is 1 tricycle + 2 barrows + 1 wheel = $15 + 10 + 1 = \$26$. So \$26 is the most money Alison can make if she builds a tricycle.

But what if Alison does not build a tricycle?

Alison could make two bicycles using 2 tyres and 4 iron bars, leaving 4 tyres, 2 iron bars and 2 wood pieces. Then, she could also make 2 barrows, leaving just 2 iron bars unused.

The total value of this second strategy is 2 bicycles + 2 barrows = $20 + 10 = \$30$. This is the maximum that can be earned given the resources available.

Can you convince yourself that no other scenarios can produce items with a greater total value?



This question comes from
the United Kingdom

Years 3+4

Years 5+6

Years 7+8 **Hard**

Years 9+10

Years 11+12





Upcycling – continued

BACKGROUND INFORMATION

Efficient use of resources is a common problem in society, that computer scientists are often asked to write programs to optimise. There are many algorithms used. Allira's first strategy was a Greedy Algorithm, so called because the algorithm always tries to make the highest value items first. Although this works in some cases, there are situations where picking the highest value item first, limits how many other items you can make and does not get you to the highest total value possible. This is the situation found in this question!



Nuts and Bolts

At the Beaver Construction factory, Benoit works on the nut  and bolt  assembly line.



His job description is as follows:

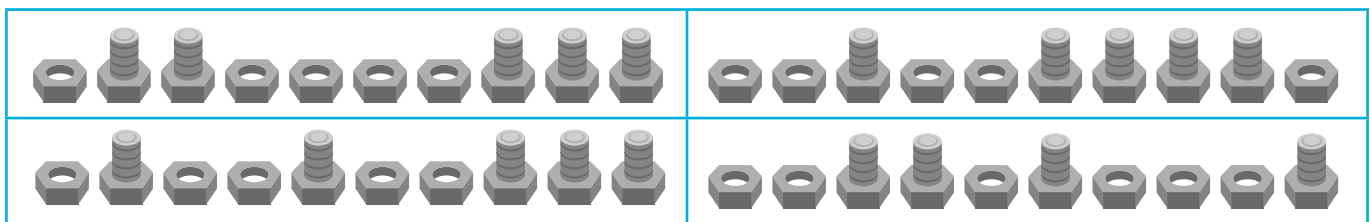
- Stand at one end of a conveyor belt, which brings him a line of nuts and bolts.
- Take each item, either a nut or a bolt, off the conveyor belt as they arrive.
- Put the nuts from the conveyor belt in the red bucket.
- Attach any bolt taken from the conveyor belt, to a nut from the red bucket, and place the assembled part onto the table behind him.

Things can go wrong for Benoit in two different ways:

1. When Benoit takes a bolt from the conveyor belt there may not be a nut in the bucket.
2. There may be no more nuts or bolts on the conveyor belt, even though there are still nuts in the bucket.

Question

Which sequence of nuts and bolts, when processed from left-to-right, will not cause things to go wrong for Benoit?



EXPLANATION

Answer



Continued on next page




Nuts and Bolts – continued

Explanation

We can keep track of the state of the bucket and conveyor belt, from left-to-right:

N = 

B = 

Bucket	Conveyor Belt
empty	N B N N B N N B B B
N	B N N B N N B B B
empty	N N B N N B B B
N	N B N N B B B
N N	B N N B B B
N	N N B B B
N N	N B B B
N N N	B B B
N N	B B
N	B
empty	empty



Looking at the other answers:

will go wrong after N B B, since there will be no nut in the bucket when the second B is encountered.



will go wrong after N N B N N B B B B, since there will be no nut in the bucket when the fifth B is encountered: notice there are only 4 N's before this B.



will go wrong after the entire sequence is processed, as there will be two nuts in the bucket, since there are 6 N's and 4 B's.

BACKGROUND INFORMATION

This task highlights the use of a *push-down automaton* (PDA). A PDA is a way of describing an *algorithm* that relies on the current *state*, but also has an unlimited amount of memory in the form of a *stack*. In this task, the state is either having a nut or having a bolt on the conveyor belt, and the stack is the bucket which holds the nuts.

A PDA can be used to recognise, or *parse*, *context-free languages*.

To recognise or parse a language means to determine if a given sequence of symbols belongs to the language. In this case, we can think of the nuts and bolts as a representation of *balanced parentheses*, where N = (an opening bracket, and B =) a closing bracket. That is, balanced parentheses are valid arrangements of parentheses in arithmetic expressions.

Examples of a sequence of parentheses which are not balanced are (((() or ()). Detecting balanced parentheses is important in *compilers*, since many programming languages rely on parentheses to indicate *nested scopes* as well as arithmetic expressions.



Overlapping Villages

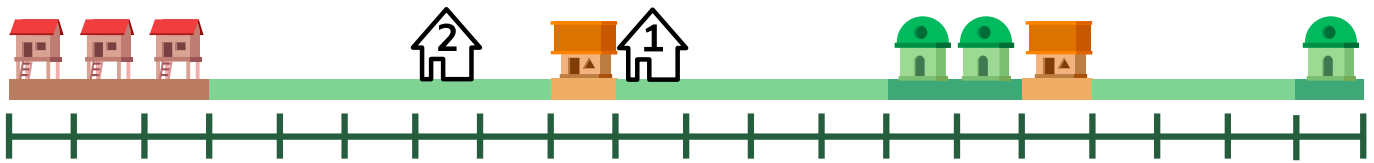
As years went by, the villages of Cabbageville , Strawbererton , and Carrotford , grew and started to overlap.

Whenever a new house is built, the villagers use the following rules to decide which village the house is assigned to:

The new house belongs to the village most houses belong to, amongst the X nearest houses.

- If there are multiple villages with the same number of nearest houses, ties are broken by assigning the new house to the same village as its nearest neighbour house.

Two new houses are to be built as per the image below. They will be assigned to villages using the value of X . House 1 will be built and assigned before House 2.



Question

What is the smallest value of X so that House 2 is assigned to Strawbererton ?

(Write a whole number from 1 to 8. Remember to press 'Save' when you have finished.)


Save

EXPLANATION

Answer

5.

Explanation

If $X=1$, both House 1 and House 2 are assigned to Carrotford .



If $X=2$, House 1 is still assigned to Carrotford , since the nearest of its two neighbours is from Carrotford. House 2 is also assigned to Carrotford, since its nearest neighbours are both from Carrotford.


If $X=3$, House 1 is assigned to Cabbageville , since two of its neighbours are from Cabbageville. House 2 is assigned to Carrotford, since its neighbours are all different, but the nearest one is from Carrotford.



Continued on next page



Overlapping Villages – continued

If $X=4$, House 1 is assigned to Carrotford  , since it has two neighbours from Cabbageville  and two from Carrotford, and the nearest one is from Carrotford. Therefore House 2 also is assigned

to Carrotford, since it has two Carrotford neighbours and two Strawberryton  neighbours, but the nearest one is from Carrotford.

If $X=5$, House 1 is assigned to Carrotford  , similar to when $X=4$. House 2 is assigned to Strawberryton  , since three out of its five neighbours are from Strawberryton.

When $X=6$ or $X=7$, House 2 is also assigned to Strawberryton, but these are not the smallest values of X when this will occur.

BACKGROUND INFORMATION

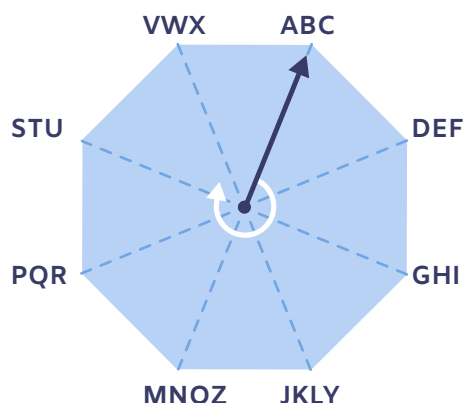
The rule to assign each house to a village is an example of a classification algorithm called *k-nearest neighbors* (kNN). kNN classifies each new data item by looking for the most similar items that have already been classified, using the value of k . In this task, the variable X plays the role of k .

In *machine learning*, kNN is often used because it is easy to implement and does not require fitting a complex model to the given data.



Cipher 8

An octagon has three or four letters written at the corners. An arrow points from the centre of the octagon to a letter group. The arrow can rotate clockwise.



Messages can be encrypted using the octagon and arrow.

At the beginning of the encryption of a new message, the arrow always points to the letters ABC.

Each letter of the message is encrypted so that:

- The **first number** indicates how many corners of the octagon the arrow should be rotated, from its current position.
- The **second number** indicates the position of the encrypted letter in the letter group to which the arrow points.

Example

TREE is encrypted with the sequence 62-73-42-02.

Question

What is the encrypted message for WATER?

72-11-26-32-53

62-11-62-22-43

62-11-26-22-53

72-11-62-32-43

EXPLANATION

Answer

72-11-62-32-43

Explanation

We can find the correct answer by creating an encrypted text for the message WATER ourselves, using the octagon and the rules provided.

The encrypted text will contain five codes, each made of two digits:

- At the start of the encryption, the arrow points to the letter group ABC. The letter W is located within the letter group that the arrow will point to after rotating 7 vertices. The letter W is the second letter in the letter group VWX, so the first code is 72.
- The second letter of the encrypted message is A. This letter is in the letter group ABC, which the arrow will point to after rotating once from its current position at letter group VWX. A is in the first position, so the second code is 11.
- We get to the letter group containing T by rotating a further 6 vertices. The letter T is the second letter in this set, so the third code is 62.
- The letter E is in the letter group that the arrow will point to after the arrow is rotated another 3 vertices. E is the second letter in the group, so the code is 32.
- The last letter is R, found in the letter group that can be reached after rotating 4 vertices from this point. R is the third letter, so the last code is 43.

In full, the final encryption is 72-11-62-32-53.

Continued on next page



Cipher 8 – continued

BACKGROUND INFORMATION

One method of protecting data from unauthorised people is *encryption*. Cryptography began as early as 3500 years ago. One of the simplest methods of encryption is to replace each letter with a different letter.

In this Bebras problem, by introducing the rotating arrow and by grouping the letters in sets the encryption method becomes more complicated. By introducing these factors it may be more difficult to crack the code if you don't understand the rules behind it. However the cipher is simple enough that once a person understands how to solve it, it is quite easy to do.

This system of encrypting messages is reliable is because it follows an algorithm. This means we could write a program to encrypt messages this way. It also means we could just as easily write a program to decode such messages - if we know how the system works.



If we altered the letters in each section, or if we put them in different groups of two or three, we would get other means of encryption. Cryptanalysis experts, who try to crack codes, might have more trouble unraveling our encryption scheme this way.



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:



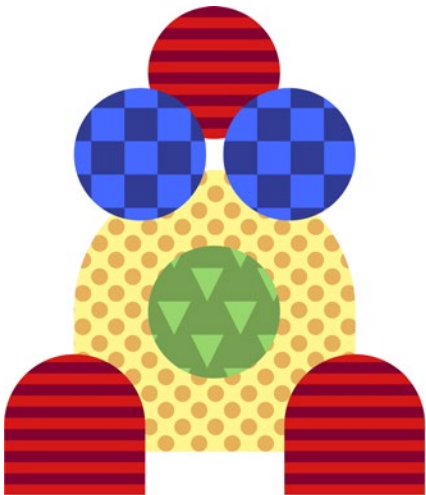
- 1. Select yellow, then paint the red part of the face and get: 
- 2. Select blue, then paint the yellow part of the face and get: 


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

Question

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

(Remember to press ‘Save’ when you have finished.)



Moves0

Save

Erase



Filling Green - continued

EXPLANATION

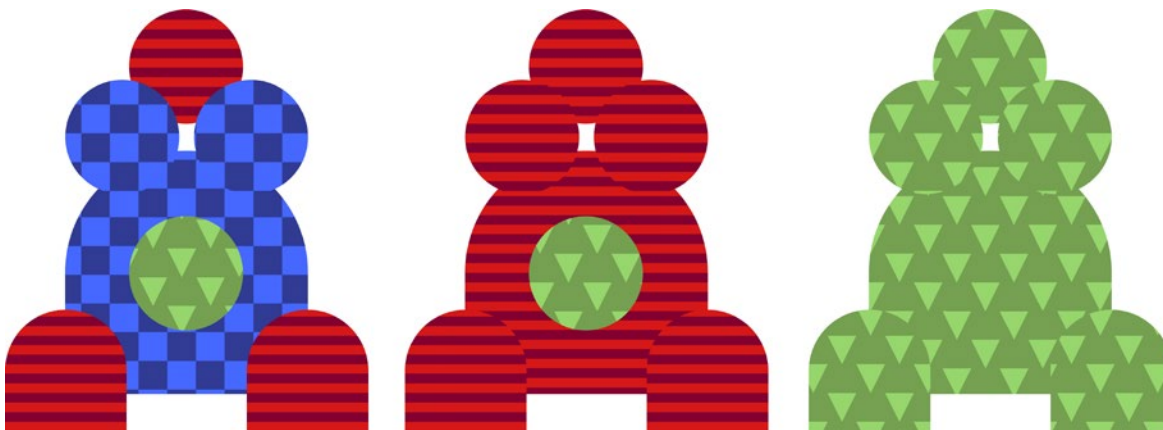
Answer

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.

Bebras Challenge 2023 Round 2

Years 9+10



Chez Connie

The takeaway restaurant “Chez Connie” is always busy at lunchtime because it sells three delicious menu items:



- Ice cream, which can be prepared in 3 minutes;
- Crêpe, which can be prepared in 8 minutes;
- Pizza, which can be prepared in 12 minutes.
- There are three queuing windows A, B, and C. All three menu items can be prepared at any window.

Connie wants to organise the orders so that the clients get served as quickly as possible. She notes the incoming orders on numbered pieces of paper so that she knows which order arrived first.

This is her first order today:



Then Connie distributes the orders to the windows A, B, and C. She always assigns the next order to the first available window. If two or more windows become available at the same time, the orders are assigned in the windows' alphabetical order - in other words, window A first, then B, then C.

Question

Connie has already distributed the first four orders. Can you distribute the next six orders?

(Remember to press 'Save' when you have finished.)

Save Erase

Continued on next page

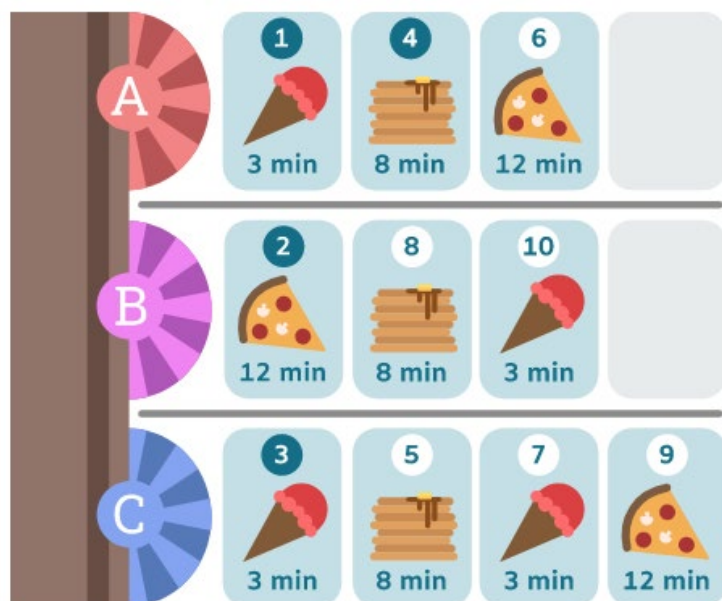


Chez Connie - continued

EXPLANATION

Answer

The correct solution is:



Explanation

To distribute the orders to the different windows, Connie has to compare the time needed to prepare the items that are already in each window's queue.

When we start allocating orders, the windows already have the following preparation time needed:

- Window A: 11 minutes
- Window B: 12 minutes
- Window C: 3 minutes

The next item (5) will be allocated to Window C, as it has the shortest preparation time at this point. This makes the new preparation times:

- Window A: 11 minutes
- Window B: 12 minutes
- Window C: 11 minutes

Since we have the same time needed for preparation at both Window A and Window C, we assign the next two orders in alphabetical order. Since we know that assigning anything to A will make C the shortest time needed, we can go ahead and assign orders 6 and 7 at once, assigning #6 to Window A, and #7 to Window C. This makes the new preparation times:

- Window A: 23 minutes
- Window B: 12 minutes
- Window C: 14 minutes

Continued on next page



Chez Connie - continued

At this point, Window B has the shortest preparation time needed, so we can assign the next order (8) to that window. This makes the new preparation times:

- Window A: 23 minutes
- Window B: 20 minutes
- Window C: 14 minutes

The next order (9) will be allocated to Window C, since it has the shortest preparation time needed by far. This makes the new preparation times:

- Window A: 23 minutes
- Window B: 20 minutes
- Window C: 26 minutes

This leaves us with the last order (10) - the window with the shortest preparation time at this point is Window B, so order 10 will be allocated to Window B.

BACKGROUND INFORMATION

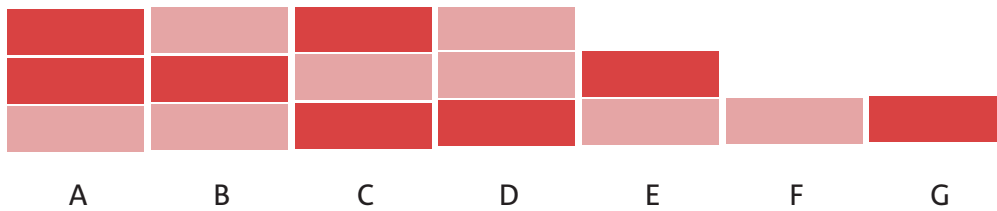
In modern computers, there are several *processors*, or several *processor cores*, which can carry out basic operations (like addition or multiplication) mostly independently of each other. In this task, the analogy with the three windows is direct: the three windows can each work on an order independently of the others.

Rather than preparing food like the windows in this question, processors execute *programs*: sequences of instructions of varying lengths designed to solve a given task. Often, hundreds of these programs (which are referred to as *processes* here) are waiting for a processor to become free and to start executing them. The attribution of processes to processors is called *scheduling* and is handled by the computer's operating system. In order to solve this task, we must play the role of the operating system ourselves - distributing orders to windows is similar, albeit in a simplified way, to assigning processes to processors at a given time.



Stacks of Tokens

Stephan placed seven stacks of tokens (light and dark red) on the table, as shown in the figure below.



They want to put these stacks on top of each other without splitting them to form two perfectly equal stacks; the resulting two stacks must have the same height (eight tokens) and the same sequence of colours (from the bottom up).

Question

Possible stacks are represented below with a bracketed notation. (x, y, z, ...) indicates that the stack is made by placing y onto x, z onto y, and so on.

Which of the following four pairs of stacks is NOT a valid solution?

(A, F, B, G) and (E, D, C)

(B, E, A) and (F, D, G, C)

(B, A, E) and (F, D, G, C)

(B, E, A) and (F, D, C, G)

EXPLANATION

Answer

The correct answer is: 3. (B, E, A) and (F, D, G, C).

Explanation

The other answers are all valid solutions to the proposed problem.

Since there are 16 tokens, of which 8 are white and 8 red, each of the two resulting stacks must be made up of 4 white and 4 red tokens. Of the four stacks A, B, C, and D (3 tokens high), two must be in one stack and the other two in the other stack. Examining the 3 token high stacks, we can see that we cannot put A and C in the same stack, and similarly we cannot put B and D in the same stack. Either of those combinations would make it impossible to balance the colours (two white tokens would need to be added to the pair A-C, and two red tokens to the pair B-D). It can also be shown that putting A and D in the same stack (and B and C in the other stack) does not lead to any result.

The quickest way of finding the solution in this case is to examine the top and bottom of each stack. Since none of the 3 token stacks are identical, any solution which uses one of the 3 token stack on the top (or bottom) of both parts of the final answer is incorrect. Since the incorrect result has A and C at the top of the two stacks, we can quickly compare those two stacks and see that the top of each stack will be different.

Continued on next page



Stacks of Tokens - continued

BACKGROUND INFORMATION

Imagine you have a bag with a lot of coins and you want to divide them into two parts of equal value. If all the coins had the same value and were even in number, then the problem would be easily solved: it would be enough to divide them into two heaps that have the same number of coins; but if the coins have many different values, then the task becomes a little more difficult.

In “more mathematical” terms, the partition problem is to decide whether a multiset (i.e., a set with a multiplicity for each element) of positive integers can be partitioned into two sub-multisets such that the sum of the numbers in the first sub-multiset equals the sum of the numbers in the second sub-multiset. This problem (in general) is NP-complete (i.e., in practice, there is no known procedure for solving it that is efficient in any case); but it can be solved using a pseudo-polynomial time algorithm, based on dynamic programming.

Our task (in general) is certainly not simpler than solving a partition problem: in fact, it is not enough that the height of the two resulting stacks is the same, but the colour sequence of the tokens must also be the same. A brute force method can however be implemented through an exhaustive search algorithm.

In the case of our task, let's list all the subsets of the seven stacks for each of which the sum amounts to eight tokens: {A, B, E}, {A, B, F, G}, {A, C, E}, {A, C, F, G}, {A, D, E}, {A, D, F, G}; here we can stop, since continuing we would find the complements of these already listed sets. Moreover, we can eliminate the sets {A, C, E} and {A, C, F, G}, since the stacks that belong to them have a total of five red tokens. Therefore, only four sets remain; for each of them and the respective complement, it is now a question of trying all the possible stackings, in order to verify if any of these produces two identical stacks... In our case, the set {A, B, E} leads to two solutions, while the set {A, B, F, G} leads to only one solution, and the last two sets to none. It is then understood that the time required by this procedure grows very quickly as the initial number of stacks (only seven, in our case) increases.



























Cheese Snack

Four beavers, Alma, Bruno, Charles, and Paula are going on a 3 week hike.

Each beaver carries their favourite cheese snack and eats a specific amount over consecutive days.

On each day, a beaver will either eat no snack or one snack.

As the chart shows, Alma eats four snacks every eight days; Bruno eats four snacks every seven days; Charles eats two snacks every four days; and Paula eats six snacks every nine days.

Days	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
 Alma	   																				
 Bruno	     																				
 Charles	 																				
 Paula	       																				

Question

If we don't know on which days the beavers will eat their snacks, what will be the maximum number of snacks eaten during the hike?

EXPLANATION

Answer

The table shows the maximum number of snacks eaten by each beaver.



































































Cheese Snack – continued

Explanation

The key to this task is to understand that the beavers may wish to eat their snacks at the beginning of their consecutive days, rather than spreading them out. So, for Alma, for example, she will eat 4 snacks during the first 8 days, 4 snacks during the next 8 days, and then there are only 5 days left before the end of the hike. Alma may wish to eat as many of her snacks as she can on during those 5 days, in which case she would eat all 4 of her snacks during the hike. So, in 21 days Alma will eat at most 12 snacks.

Similarly, Bruno will eat at most 12 snacks in 21 days.

Days	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
 Alma	   								   								   				
 Bruno	   								   								   				
 Charles	 		 		 		 		 				 				 				
 Paula	     						     						     								

Charles will eat at most 11 snacks in 21 days. There is one day left over after as you can see from the picture, and he can eat a snack on that day.

Paula will eat at most 15 snacks in 21 days. Paula can eat one snack during each of her last three days. This totals 50 snacks in 21 days as a maximum.

BACKGROUND INFORMATION

The computational thinking concept illustrated in this task is algorithms.

This task is an example of a constraint satisfaction problem. A constraint is a limitation, or a rule that cannot be broken. The term satisfaction refers to a situation where it is possible to follow all of the rules. In this task, you have to figure out what is the maximum number of snacks that can be eaten while following all of the rules. This problem is straightforward, and can be solved in a single step using algebra. Often, however, constraint satisfaction problems require one to make a guesses and change your mind many times while solving them, a strategy known as backtracking. Constraint satisfaction is a common problem to be solved in the world of business, economics, public services, and healthcare. Often, computer software can help. But computer scientists have also been able to show that there are many constraint satisfaction problems that even computer software has great difficulty in solving.



Bebras Runs

A fellowship of nine beavers must solve a riddle to open an ancient door on their quest.

To open the doors, the beavers must sort a sequence of numbers from smallest to largest. Beaver Borris believes he has a great idea that can sort any list of numbers they encounter.

He goes through the list of numbers from left to right and performs the following steps:

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

Borris performs one pass on the following list of numbers:

5	3	5	6	7	4	3	6	8	4
---	---	---	---	---	---	---	---	---	---

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

1. 3556743684
2. 3556743684
3. 3556743684
4. 3556743684
5. 3556473684
6. 3556437684
7. 3556436784
8. 3556436784
9. 3556436748

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

3	5	5	6	4	3	6	7	4	8
---	---	---	---	---	---	---	---	---	---

Question

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

(Enter your answer into the text box. Remember to press 'Save' when you have finished.)

Save

Continued on next page



Bebras Runs – continued

EXPLANATION

Answer

3543564678.

Explanation

Performing two more passes leads to the following list order:

1. 3556436748
2. 3556436748
3. 3556436748
4. 3554636748
5. 3554366748
6. 3554366748
7. 3554366748
8. 3554366478
9. 3554366478

And,

1. 3554366478
2. 3554366478
3. 3545366478
4. 3543566478
5. 3543566478
6. 3543566478
7. 3543564678
8. 3543564678
9. 3543564678

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

3	5	4	3	5	6	4	6	7	8
---	---	---	---	---	---	---	---	---	---

BACKGROUND INFORMATION

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

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



Antivirus

Annabelle developed a new security software that can detect dangerous software.

It scans the incoming program using this list consisting of four items:

- save in the system folder
- read from the system folder
- add to autostart
- send via e-mail

If it finds at least two specific statements that contain all the words from one item in the list, the program is considered dangerous and is quarantined.

Question

Which of the following programs will be quarantined?

A. read the file from the document folder; open the file; save the file in the document folder; send the file via e-mail;

B. read the file from the document folder; open the file; add the file to the autostart;

C. read the file from the system folder; open the file; save the file in the document folder; send the file via e-mail;

D. read the file from the autostart folder; open the file; save the file in the document folder; send the file via e-mail;

EXPLANATION

Answer

The correct answer is C.

Explanation

In the following table statements of the four possible answers are bolded.

In the answer options A, B and D you can find only one of these types of statements.

A	C
read the file from the document folder;	read the file from the system folder;
open the file;	open the file;
save the file in the document folder;	save the file in the document folder;
send the file via e-mail;	send the file via e-mail;
B	D
read the file from the Document folder;	read the file from the autostart folder;
open the file;	open the file;
add the file to the autostart;	save the file in the document folder;
	send the file via e-mail;

Continued on next page



Antivirus – continued

BACKGROUND INFORMATION

Security software has been developed to detect and remove dangerous programs that can harm the computer. Sometimes it is possible to identify a dangerous program like a virus from the name of the file. The security software always scans the code and tries to find critical code parts, using information from its database.

Sometimes the security software uses a “sandbox” – a protected environment – where it executes the code and checks whether the result is different from the expected result.

The newest software use AI, as well and they “learn”, extend the database to detect easier and faster a virus.



Seating Arrangement

Eight friends are sitting in a circle. They are all facing inwards.

We know the following facts about where they are sitting:

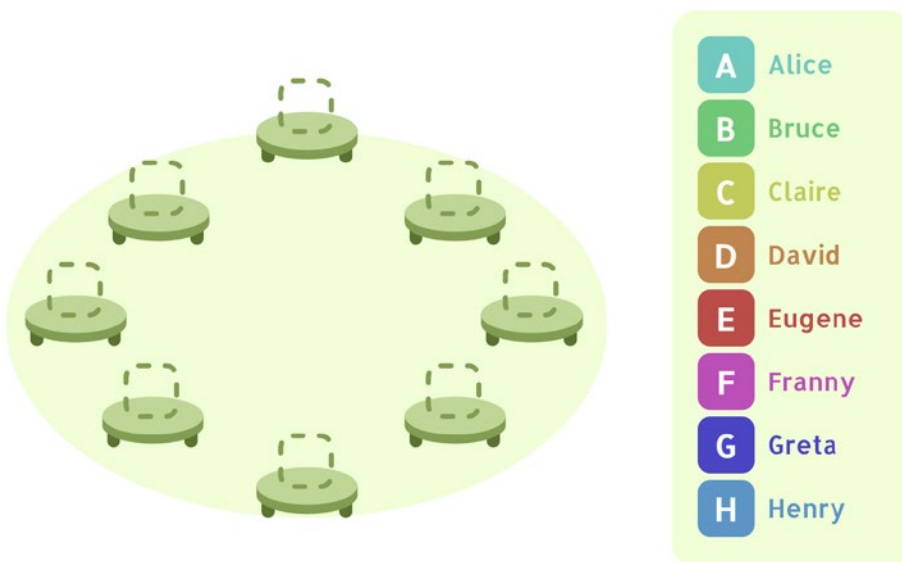
1. Alice is sitting directly opposite of David.
2. Henry is sitting between Greta and Eugene.
3. Franny is not next to Alice or David.
4. There is one person between Greta and Claire.
5. Eugene is sitting immediately to David's left.

Question

Place the friends in their correct places in the circle by dragging the letter next to their name to their chair.

There may be multiple correct solutions; you only need to find one.

(Remember to press 'Save' when you have finished.)



Save

Erase

EXPLANATION

Answer

The following answer assumes that Alice is placed in the top seat. There are in fact 8 different answers, as an answer can be found corresponding to Alice being in any of the seats.

Continued on next page



Seating Arrangement – cont'd

Explanation

Fact 1, Alice sits directly opposite to David, enables us to seat David.

Now, Fact 5, Eugene is beside David, on David's left, enables us to seat Eugene.

At this stage, Fact 2, Henry sits between Greta and Eugene, tells us where Henry sits. Knowing where Henry sits we now can place Greta.

With five friends placed, Fact 3, Franny is not beside Alice or David, leaves only one space for Franny.

Fact 4, There is one person between Greta and Clare, now tells us where Clare sits.

Finally there is only one seat and one friend to place so we can place Bruce.

BACKGROUND INFORMATION

Logic is about following rules, understanding ordering and negation.

For example, the key logic operator used in this question is negation, which is used whenever we say **NOT**. So, the fact that states that "Franny is not beside Alice or David" can be stated differently as "Franny not beside (Alice or David)" which is the same as "(Franny not beside Alice) and (Franny not beside David)".

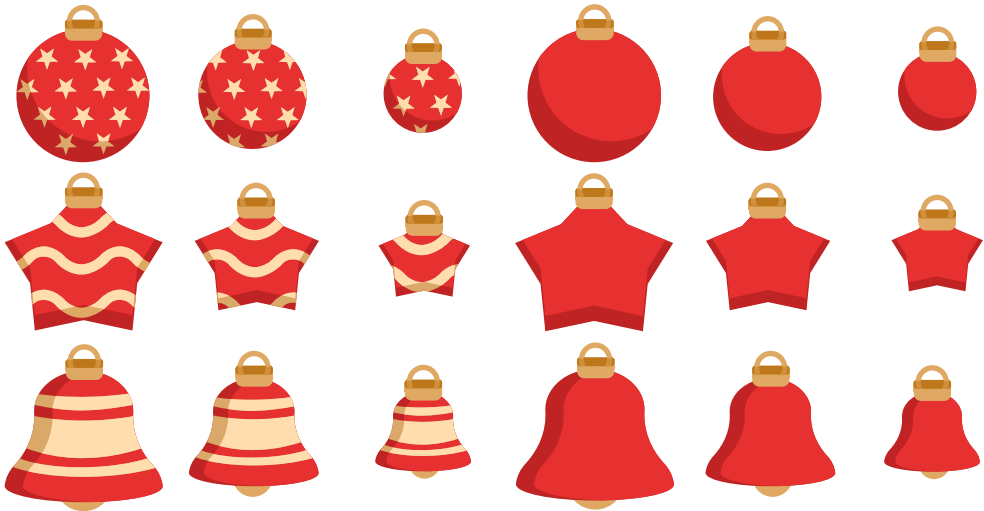
Older students may be interested that the negation operator, not, changes the logical expression of not (A or B) to an equivalent (not A) and (not B) which is an example of De Morgan's law.

Knowing how logical expressions can be rewritten, combined, or simplified is a very useful skill in computer science.



Decorations

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



Michaela has a favourite ornament and Lara has to figure out which one it is, by asking questions about the shape, size, and pattern of the favourite ornament.

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

Can you figure out which four questions Michaela asked? Drag them to the proper question spot in the table below.

Note: there is more than one correct solution, you only have to provide one.

Question

Drag the questions Lara may have asked next to the answers Michaela has given.
(Remember to press 'Save' when you have finished.)

Lara's questions	Michaela's answers
Is it a star?	Yes
Is it a bell?	No
Does it have a pattern?	Yes
Is it large?	No
Is it small?	
Is it medium?	
Is it round?	

Save Erase



Decorations – continued

EXPLANATION

Answer

Here are four solutions:

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

Explanation

Since there is only one way to ask about a pattern: Does it have a pattern? and the resulting favourite ornament has no pattern, one of the questions Lara asked was 'does it have a pattern?' And the answer to it was no. We have 3 questions left.

To guess the shape or size, we need 1 question (if we guess correctly right away) or 2 questions (if our first guess was wrong).

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

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

She had to ask the question 'does it have a pattern?', to which the answer was no, at the end, because, as shown before, one of the other questions needs no as an answer, followed by a yes.

BACKGROUND INFORMATION

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

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

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



Longest Sequence

Here is a sequence made using three different shapes:



You may change exactly three of the shapes.

Question

What is the length of the *longest unbroken chain* of identical shapes possible?

6 7 8 9

EXPLANATION

Answer

The correct answer is 7. To show this, we need to prove two things: (1) that an unbroken chain of length 7 is possible, and (2) that an unbroken chain of length greater than 7 is not possible.

Explanation

The first part is easy to prove. Here is how an unbroken chain of 7 triangles can be made:



To prove that an unbroken chain of length greater than 7 is not possible, consider any chain of length 8. Since we are only allowed to change three shapes, any chain of length 8 in the original sequence must already have five identical shapes in it.

There are nine chains of length 8 in the original sequence, a few of which are shown below. In no chain can five identical shapes be found. Convince yourself of this for the remaining unshown five chains.



Continued on next page



Longest Sequence – continued

Since it is not possible to have an unbroken chain of length 8, it is certainly not possible to have an unbroken chain of length greater than 8.

Thus, we have shown that the length of the longest unbroken chain of identical shapes possible is 7.

BACKGROUND INFORMATION

This task is related to finding a *longest substring* that matches some given criteria.

There are many instances in informatics where finding a longest substring is useful, in particular, finding a longest *common* substring given two strings.

Finding the longest common substring can help detect plagiarism, and help compress data by data deduplication (removing redundant copies of data).

Some techniques that can be used to find longest sequences include the two pointer method, and the sliding window



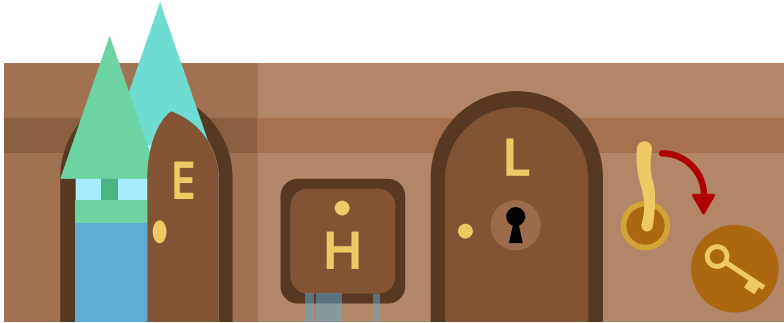
Secret Entrance

Yulia the beaver wanted to protect her lodge so that only she could enter it.

She built a secret chamber that is entered through a door called the 'entrance-door'.

Inside the chamber are two inner doors: one with a key-hole that opens into her lodge (the 'lodge-door') and one that opens to the water behind the dam (the 'hatch' marked 'H').

She then added a lever to the wall. Pulling the lever reveals a key that will open the lodge-door.



If the entrance-door is closed without first pulling the lever then the hatch will open and a powerful gush of water will come from behind the dam and flood the chamber.

There are two other cases that will cause the hatch to open. If the key is turned in the door:

- before closing the entrance-door, the hatch will open and the gush of water will flush the intruder out into the river below,
- before returning the lever back to its original position, the hatch will open and the gush of water will flood the chamber.

Question

A rogue beaver tries to get into Yulia's lodge. From the list below, identify which attempts to gain access would **not** be successful.

(Remember to press 'Save' when you have finished.)

A. Enter the chamber, pull the lever, take the key, close the entrance-door, turn the key in the lodge-door, return the lever back to its original position.

B. Enter the chamber, pull the lever, take the key, return the lever back to its original position, close the entrance-door, turn the key in the lodge-door.

C. Enter the chamber, close the entrance-door, pull the lever, take the key, return the lever back to its original position, turn the key in the lodge-door.

D. Enter the chamber, pull the lever, take the key, return the lever back to its original position, turn the key in the lodge-door, close the entrance-door.

E. Enter the chamber, pull the lever, take the key, close the entrance-door, return the lever back to its original position, turn the key in the lodge-door.

Save

Erase

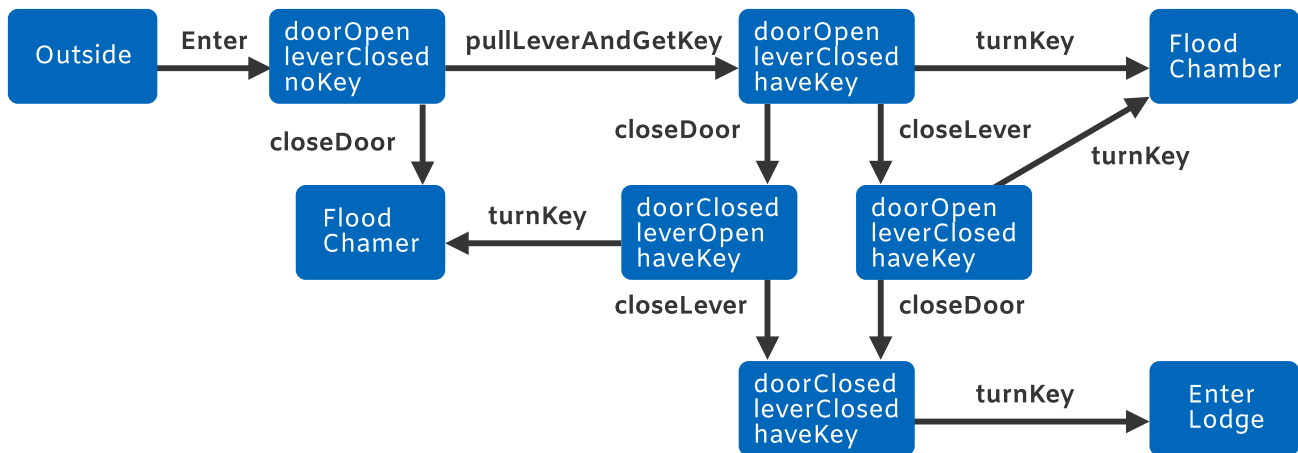
Continued on next page

Secret Entrance – continued

EXPLANATION

Answer

The correct answer is that attempts **A**, **C** and **D** will cause the rogue beaver to be unsuccessful and get gushed with water.



Explanation

Attempt A forgets to return the lever back to its original position before turning the key in the lodge-door, causing the chamber to flood.

Attempt C closes the entrance-door before pulling the lever, causing the chamber to flood.

Attempt D forgets to close the entrance-door before turning the key, causing the intruder to be flushed out into the river.

Attempts B and E are the only options that remembers to (1) leave the entrance-door open while the lever is pulled, (2) return the lever to its original position before turning the key, and (3) close the entrance-door before turning the key.

BACKGROUND INFORMATION

This task involves analyzing the design of the secret entrance to Yulia's lodge. It involves tracing through the different conditions (i.e. if the key is turned but the entrance-door is open, what happens? If the entrance-door is closed, but the lever is not returned to its original position, what happens?). The task also involves concluding about the final outcomes (does the intruder get flushed, does the chamber get flooded, or does the intruder gain access?). The different possibilities that can occur are considered a 'logical pathways'. The overall task involves 'algorithmic thinking'.

This problem is sufficiently simple that we can abstract the computational properties of the problem as a finite-state automaton. (See diagram in the 'Answer Explanation' section).





Rock Paper Scissors

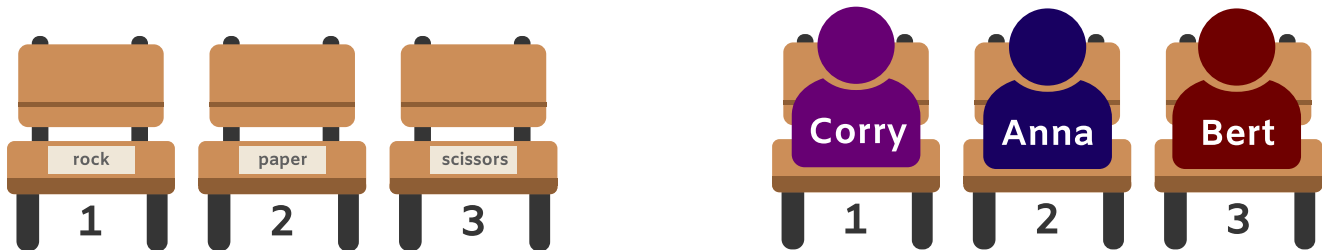
You are playing rock paper scissors with three of your friends.

On each of the three chairs numbered 1, 2, 3, you have placed a piece of paper that says 'rock', 'paper', or 'scissors'.

Remember:

- rock beats scissors
- scissors beats paper
- paper beats rock

Your friends, Anna, Bert and Corry sit down on the chairs as indicated.



Now, the following three things happen:

- first, each friend puts the piece of paper on their chair in an envelope
- now, you turn around so you cannot see what happens
- then, two friends will swap chairs, leaving the envelope behind
- then, two friends will swap envelopes, remaining in their seat
- finally, two friends will swap chairs, taking their envelopes with them.

Question

You really want to beat Anna. Which place and envelope should you take?

It does not matter, you will always win

It does not matter, you will always lose

It is impossible to know, it could be either way

Take Bert's place

Take Corry's place

Continued on next page



Rock Paper Scissors – continued

EXPLANATION

Answer

The correct answer is E, Take Corry's place.

Explanation

If you look at the the initial situation: Anna beats Bert, Bert beats Corry and Corry beats Anna

When two friends swap places it does not matter which two swap, the order of who beats who will always change.

So now, the situation is $B > A$, $C > B$ and $A > C$.

Now two friends will swap envelopes, which means that the order reverts back to the initial situation.

Now two friends will swap places, but they keep their own envelope. This means that nothing changes.

So, if you want to beat Anna you have to take Corry's place.

BACKGROUND INFORMATION

This is about analysing a problem and what happens with the situation in the problem when things change.

The interesting thing is that we are analysing a famous game from game theory where there is no generic optimal solution. This is called a Nash Equilibrium.

But because we know the starting point of the rock/paper and scissor symbols we can draw a small graph of which person beats which other person.

The interesting fact here is that when two people swap envelopes (either by moving to another seat and leaving their envelope behind, or by directly swapping envelopes) the graph turns around: so all 'X beats Y' turn to 'Y beats X'.

Because informatics is about keeping track of information and using an algorithm to infer things from that, you can easily see whose place you should take in order to win.



Guess Poker Game

Anna, Bob, and Chris are good friends who like to play guessing games using cards.



For this game Chris selects a card from a small handful of cards. There are 13 cards in total. He tells the suit of the card to Anna and the number to Bob. Both Anna and Bob know that the cards include:

- Spade J, 8, 4, 2
- Heart A, Q, 4
- Diamond A, 5
- Club K, Q, 5, 4

Anna and Bob are good at reasoning. They have the following conversation:

Bob: “I don’t know the card”.

Anna: “I already knew you couldn’t know this card”.

Bob: “Now I know the card”.

Anna: “And now, I know too”.

Question

According to the above conversation, can you infer which card Chris selected?

Diamond A

Spade 4

Heart Q

Diamond 5



Guess Poker Game – continued

EXPLANATION

Answer

The correct answer is Diamond 5.

Explanation

This problem is designed to challenge our logical reasoning skills. It can be solved by working through the information provided. In this scenario Anna and Bob know all the potential answers Anna is sure of the suit and Bob is sure of the number. The rest of this explanation will be broken down following the information provided through their conversations.

“I don’t know the card” Bob’s statements indicates that the card cannot be a unique number/picture. If it were, Bob would be able to determine the card from the number alone. So the card must have a number shared by multiple cards.

“I already knew you couldn’t know the card” Anna’s statement tells us that Anna knew the card could not be either a club or a spade as they contained unique numbers.

“Now I know the card” Bob’s statement tells us that understands that Anna cannot have either a spade or club the suit must be either a heart or diamond.

“And now I know too” Anna had deduced that if Bob now knows what the card is the value must be one that is only in one suit. In this case the only value that is unique is the five. Therefore the answer must be the Five of Diamonds.

BACKGROUND INFORMATION

Deductive reasoning, also deductive logic, is the process of reasoning from one or more statements (premises) to reach a logical conclusion. It consists of drawing particular conclusions from a general premise or hypothesis. For this problem, this card must be one of the 13 kinds of cards, such as spade J. This is the premise, and by the information transmitted between Anna and Bob, we can finally deduce the answer.



Even Teams

Beaverball is a team game played between two teams consisting of six field players.

Now two teams, “Team River” and “Team Forest”, are about to start a friendly match. Both teams have seven members, and they now need to select which six members will play (the remaining members will act as referees). Each member has an experience rating (shown below under each member) that reflects how experienced the member is.








The teams want to make the match as even as possible, so they decide to select both teams’ players in such a manner that the players in both teams have exactly the same average experience ratings.

Question

Select six players from Team River and six players from Team Forest in such manner that the average experience rating of Team River is equal to the average experience rating of Team Forest.

(Click on a player to select them and click again to deselect. Remember to press ‘Save’ when you have finished.)

Team River

						
27	42	50	29	47	20	38

Team Forest

						
45	24	44	41	35	21	28

Save

Erase

Continued on next page



Even Teams – continued

EXPLANATION

Answer

Team River



Team Forest



Explanation

Both team selections have equal rating totals

$27 + 42 + 29 + 47 + 20 + 38 = 45 + 24 + 44 + 41 + 21 + 28 = 203$,
and thus also their average ratings are equal.

The solution can be found with reasonably little effort by first calculating the overall sums of both teams (River: 253, Forest: 238) and then looking for a River-Forest player pair where the River player's rating is $253 - 238 = 15$ higher than the Forest player's rating. Leaving out the bird with rating 50 and the elephant with rating 35 fulfills this condition (and no other pair does). Selecting bird and elephant as referees will leave both teams with equal total player ratings $253 - 50 = 203$ and $238 - 35 = 203$.

BACKGROUND INFORMATION

The task involves finding equal subset sums from two sets (here: teams). This is a simple variant of a problem known as “fair subset sum problem”, which in turn is related to the famous basic subset sum problem in computer science. The basic subset sum problem asks us to check if a given set of numbers contains a subset whose sum equals some predefined value x . The subset sum problem is very easy to formulate but surprisingly difficult to solve efficiently.



Movie Night

Ms. Naya is organising a movie night for her seven students. She will be using the school's TV room.

To avoid seating conflicts, Ms. Naya sets specific seating rules:

- Birthday rule: each student adds together their birth day (DD) and birth month (MM). This is noted as (DD/MM) below their name. Ms. Naya works out how many 7s can fit into the sum, and uses the whole number that is left over to assign their seat.

For example, Joud is assigned chair number 1 because when we add $24 + 12$, we get 36, and the number 7 can fit into 36 five times with 1 leftover.

- Collision rule: if one student's chair is already used by another student, they should move to the next chair.

For example, if chair 3 is taken, the student moves to chair 4. If both chairs 4 and 5 are taken, then the student moves to chair 6. If chair 6 is taken, then the student moves to chair number 0.

Question

The students enter the room in the order seen below. Drag their names to their correct seats.

(Remember to press 'Save' when you have finished.)

Save Erase

EXPLANATION

Answer

Continued on next page



Movie Night – continued

Explanation

When applying the seating rules, the students will end up using chairs as represented in the following seating map:

The black borders mean the student is using the chair that matches their given number. The red borders means that the student moved at least one seat before they found their final seat.

We arrive at the seating map by using the following instructions:

First, calculate the sum of the birth day value and the birth month value. Then we calculate the result by dividing the sum by 7, and identifying the whole number which remains (modulo 7). This gives us the expected seat for each student.

Student	Joud	Mary	Han	Mo	Jon	Sara	Emy
Birth (DD/MM)	24/12	05/01	08/02	16/04	09/09	02/12	30/01
Sum (DD + MM)	$24+12=36$	$05+01=6$	$08+02=10$	$16+04=20$	$09+09=18$	$02+12=14$	$30+01=31$

Modulo 7	1	6	3	6	4	0	3
----------	---	---	---	---	---	---	---

The table shows that Mary and Mo both are expected to sit in chair 6. However, only one of them may sit in chair 6, and the other must sit in a different chair. The same is true of Han and Emy, who both are given number 3. The final result is based on the order in which they enter the room.

BACKGROUND INFORMATION

This method of evaluating and prioritising is also used when computers save files to a disk. Computers use file names to find a place to put our files. As a way to prevent files with the same data from being saved in the same place, your computer may prompt you to save them in a different place, change their file name, or change their file name itself. We call this method *hashing* and prevent it by using a *collision avoidance method*.

Basically, it allows quicker access to information once it is placed in the data structure.

This question is an example of *modular arithmetic*. Modular arithmetic relates to numbers and cycles. One of the ways we use it daily, is when we think about 12 hour time on a clock. Programmers can also use modular arithmetic to track time, and loops of code.



Ordering

Form a ten-digit number, using each decimal digit (0-9) exactly once.

The number must be the smallest number that satisfies all these conditions:

0→9, 1→0, 1→2, 1→6, 2→3, 3→5,
3→4, 6→5, 5→7, 7→8, 9→7

Condition $a \rightarrow b$ means that, in the number, digit a must appear left of digit b .

Question

Create the smallest number that satisfies all conditions by dragging all numbers to the correct places.

(Remember to press 'Save' when you have finished.)

0123456789

Save

Erase

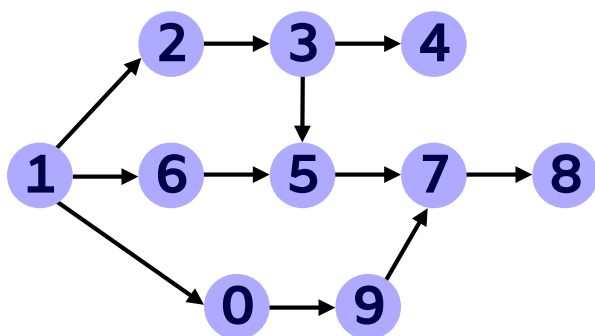
EXPLANATION

Answer

This is the correct answer: 1 0 2 3 4 6 5 9 7 8. This number satisfies all conditions, and it is the smallest such number.

Explanation

How can we reliably determine that number? Let us represent the set of conditions as a graph, which we set up and draw this way: The digits (the nodes of the graph) are put in circles, and there is an arrow pointing from one digit node to another (a directed edge in the graph) according to the conditions.





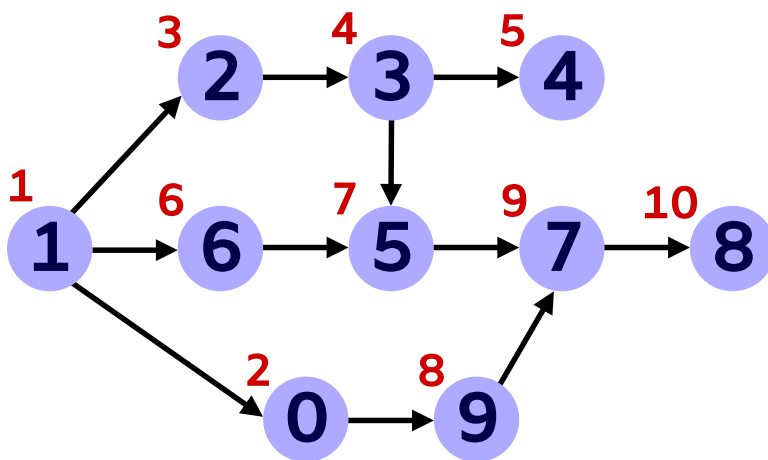
Ordering – continued

Now we can see that the number we are looking for needs to start with a digit with no incoming arrow: In this case, there is no condition demanding another digit to appear more left. Based on this insight, we form the number using this algorithm:

Until all digits are used in forming the number, repeat:

1. *From all digit nodes without incoming arrows, choose the node with the smallest digit and append this digit to the digits already used in forming the number.*
2. *Remove all arrows starting at the chosen node.*

While executing this algorithm, the nodes of the graph are visited in the order given by the red numbers. Hence, the algorithm results in the number 1 0 2 3 4 6 5 9 7 8, as mentioned above.



BACKGROUND INFORMATION

In this Bebras task, we were given only partial information on how the digits should be put in order to form the desired number. Still we were asked to form the number, as a fully ordered sequence of digits. In mathematics, a fully ordered sequence of objects (numbers or other) that satisfies such partial ordering information is called a topological sorting of these objects. Luckily, computer scientists know methods to compute such a topological sorting from the partial information – or to find out if this is impossible. The algorithm used in the answer explanation is the standard method. For instance, there is no number that will satisfy all these conditions: $2 \rightarrow 4$, $4 \rightarrow 3$, $3 \rightarrow 2$. Do you see why?

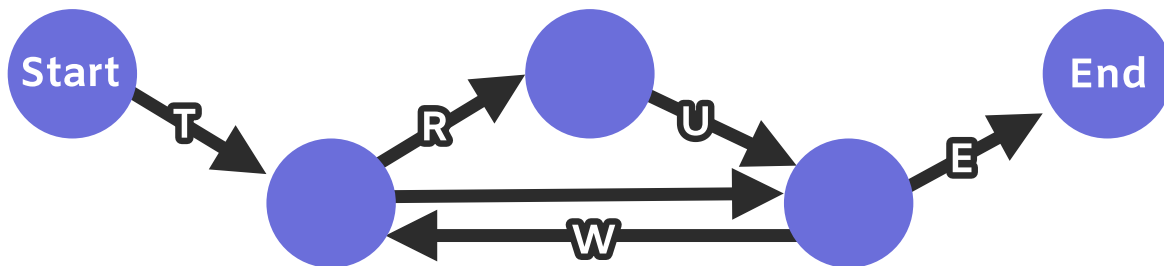


Words

Bridgit is creating random “words” using a specific diagram that she has found.

She always starts in the circle labelled ‘Start’ and follows the arrows until she ends up in the circle with the label ‘End’.

Every letter that she passes while doing this, she writes down.



Note: Only arrows with letters on them will add letters to the word.

The diagram above shows, for instance, that she could create the “words” ‘twww’e or ‘true’.

Question

How many different words of 8 letters can Bridgit create?

9 10 12 14

EXPLANATION

Answer

The correct answer is 9.

Explanation

Note that Bridgit always has to use ‘T’ at the beginning, and ‘E’ at the end. So we’re looking for all the six letter words with ‘R’, ‘U’ and ‘W’.

Let’s turn the problem into a smaller problem: how many different ways are there to create words of a specific length:

- length 1: 1 (W; follow the empty line, follow the W, follow the empty line)
- length 2: 2 (WW and RU; follow the R, follow the U)
- length 3: 3 (WWW; WRU; RUW)
- length 4: 4 (WWWW; WWRU; RUWW; WRUW)
- length 5: 6 (WWWWW; WWRUW; WWRUW; WRUWW; RUWWW; RUWRU)
- length 6: 9 (WWWWWW, WWWWRU, WWWRUW, WWRUWW, WRUWWW, RUWWWW, WRUWRU, RUWWRU, RUWRUW)



This question comes from
the United States

Years 3+4

Years 5+6

Years 7+8

Years 9+10 **Hard**

Years 11+12



Words – continued

Important patterns about the words (excluding the ‘T’ at the beginning and ‘E’ at the end) include:

- the letters ‘R’ and ‘U’ must always appear as a pair.
 - two ‘RU’ pairs cannot appear consecutively.
-

BACKGROUND INFORMATION

This question demonstrates the usefulness of an algorithmic “machine”, or *routine*, that generates a sequence of letters with certain constraints. Searching through all possibilities is what a computer is really good at. In this question the task asks us to make a conscious effort of writing down all words, but we have to remember to not count words twice.

These sort of systems are sometimes used for generating passwords for students competing in Bebras, or to produce number plates for cars. In these cases unique sequences of characters are required but they must follow a specified format.

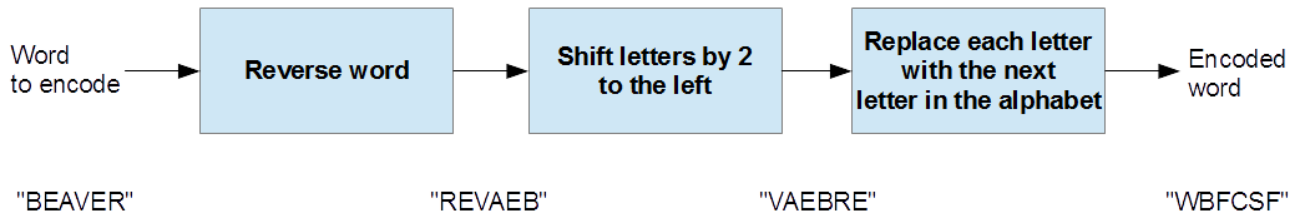
Bebras Challenge 2023 Round 2

Years 11+12



You Won't Find It

Beaver Alex and Beaver Betty send each other messages using the following sequence of transformations on every word.



For example, the word "BEAVER" is transformed to "WBFCSF".

Beaver Betty receives the encoded message "PMGEP" from beaver Alex.

Question

What did Alex want to say?

FLOOD

RIVER

KNOCK

LODGE

EXPLANATION

Answer

Correct answer is FLOOD.

Explanation

The steps of the transformation, applied in the reverse order, are:

"PMGEP" → "OLFDO" → "DOOLF" → "FLOOD"

That is:

- Replace each letter with the previous in the alphabet;
- Shift letters by 2 to the right;
- Reverse word.

The other answers are not correct.

BACKGROUND INFORMATION

The image in this task is part of a simple flowchart, that explains how to change a word step by step. Flowcharts are a way to describe algorithms. In this task the algorithm changes text so that nobody can understand it. This is called ciphering.



Word Chains

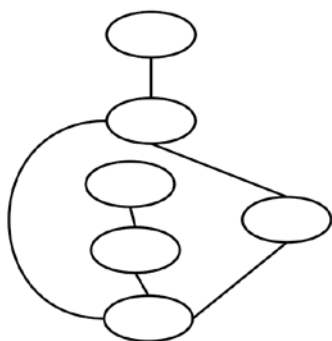
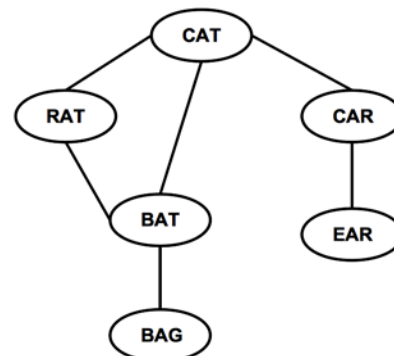
For her homework, Maisie had to write words on cards and connect them with rubber bands.

The teacher told her to connect any two words that differ by changing one letter.

Maisie did this, as you can see in the picture on the right.

When Maisie returned from having a break she got a surprise.

Sophie, her friend, had erased all the words!



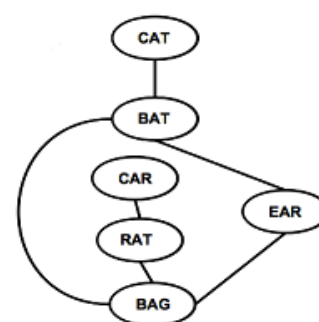
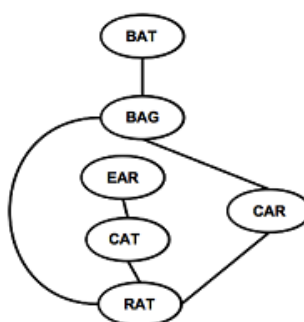
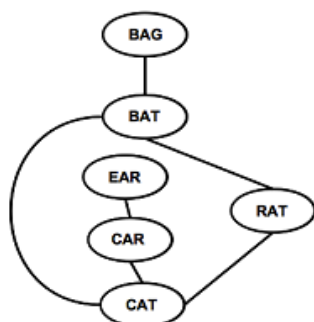
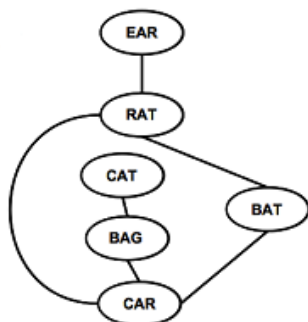
Also, the cards were completely mixed up, as you can see in the image on the left.

Importantly, the rubber bands still connected them as before.

Maisie was sure she could put the words back in the correct place.

Question

Which of the pictures below contains the words in exactly the right places?



EXPLANATION

Answer

The answer is B.

Explanation

We can proceed by counting the edges going from each node. There are 2 nodes with 3 edges, 2 nodes have two edges and 2 nodes have 1 edge. There is only one node with one edge connected to a node that has two edges. So we have identified the node for “EAR” and “CAR”. We can continue with this method.

BACKGROUND INFORMATION

This is a problem about graphs: a graph is a set of objects, where some pairs of objects are connected. This task asks us to build a graph.



WhatDoesItDo

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

For example:

- 90 `div` 7 is 12, and
- 90 `mod` 7 is 1.

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



Question

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

(Type your answer into the text box. Remember to press 'Save' when you have finished.)

Answer:

Save

Continued on next page



WhatDoesItDo - continued

EXPLANATION

Answer

The correct answer is 30241.

Explanation

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

Function Level	Input	$p(m < 10)$	$m \text{ div } 10$	$m \text{ mod } 10$	Output
1	30241	False	3024	-	-
2	3024	False	302	-	-
3	302	False	30	-	-
4	30	False	3	-	-
5	3	True	-	-	3
4	30	-	-	0	0
3	302	-	-	2	2
2	3024	-	-	4	4
1	30241	-	-	1	1

BACKGROUND INFORMATION

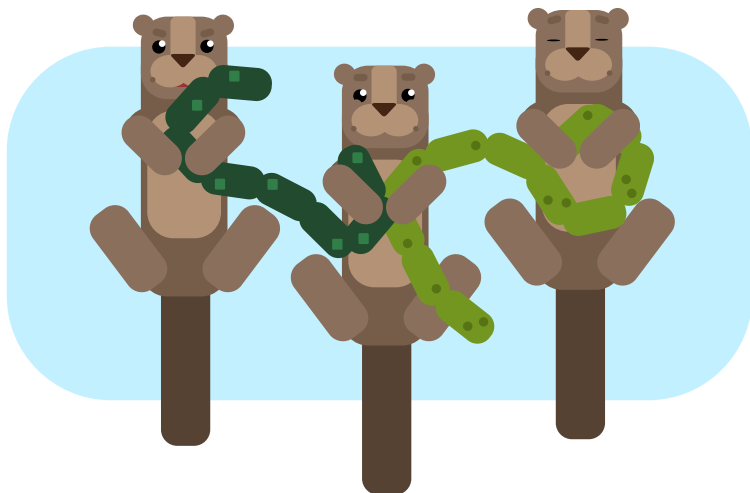
When we repeatedly need to ask the computer to do the same set of tasks over and over again, we will define this set of tasks as a *function* for easier reference in our program. This way, when we need to do this set of tasks, we can call the function instead of writing out the set of instructions over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

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



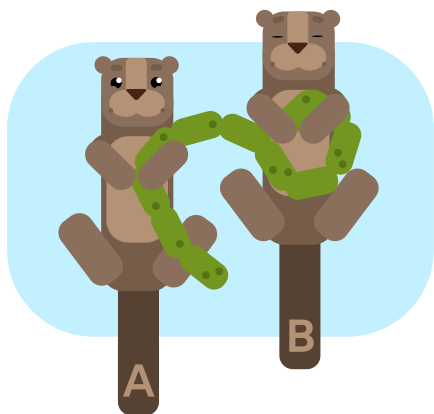
Napping Together

When two otters in Otter Kingdom meet each other, they will wrap seaweed around themselves so that they can stay together during nap time. However, to avoid knots, if two otters are already connected through a seaweed chain, they won't wrap another seaweed.

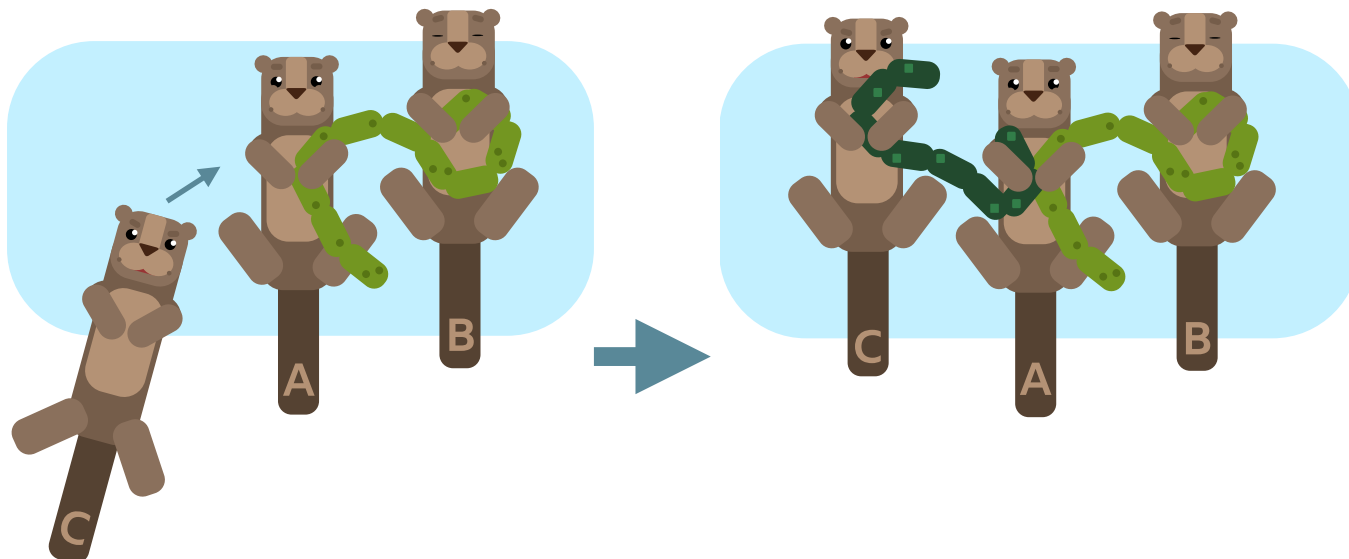


For instance, if otters meet each other in the following order: A - B, A - C, B - C:

1. Otter B meets Otter A. They wrap seaweed around themselves.



2. Otter C meets Otter A. They wrap seaweed around themselves.

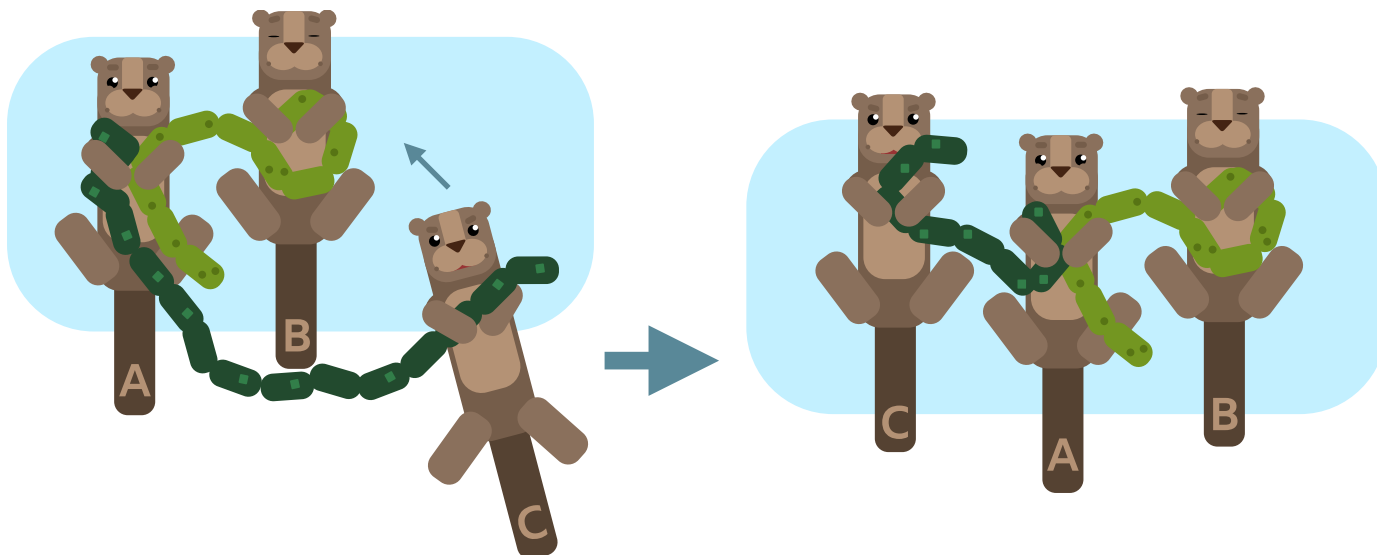


Continued on next page



Napping Together – continued

3. Otter C meets Otter B. Since they are already connected through Otter A, they *won't* wrap seaweed around themselves.



Question

Otters meet each other in the following order:

A - B, A - C, B - C, D - E, A - E, D - F, A - F

How many seaweeds are wrapped around otter A?

(Type your answer as an integer in the text box. Remember to press 'Save' when you have finished.)

Save

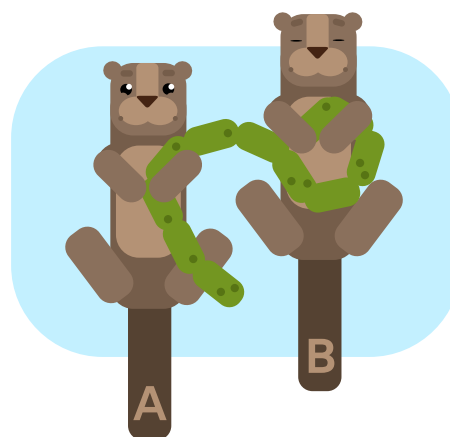
EXPLANATION

Answer

3.

Explanation

A meets B. They wrap seaweed around themselves.
One seaweed is wrapped around otter A at this point.

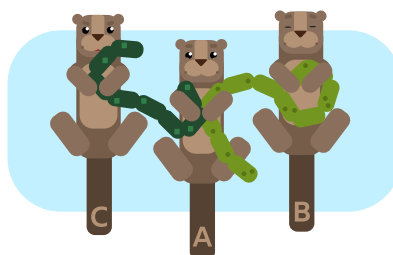


Continued on next page

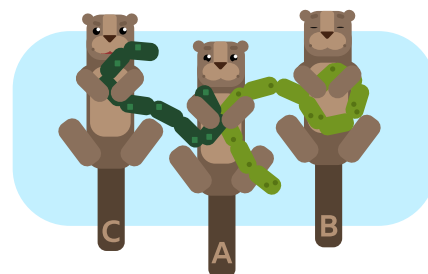
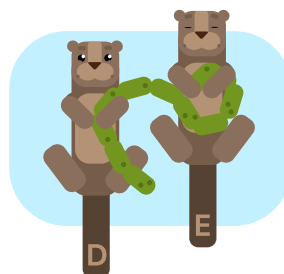


Napping Together – continued

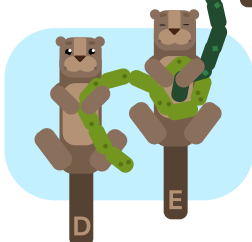
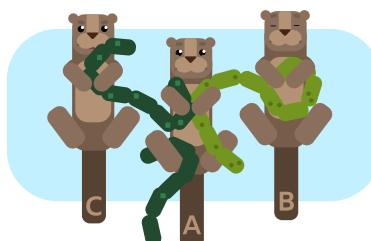
A meets C. They wrap seaweed around themselves.
Two seaweeds is wrapped around otter A at this point.



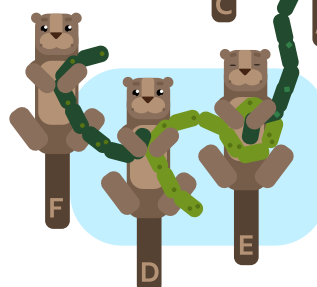
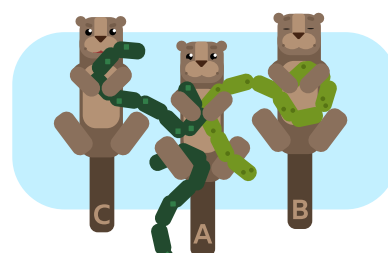
B meets C. Since they are already connected, they won't wrap any seaweed around themselves. D meets E. They wrap seaweed around themselves.



A meets E. They wrap seaweed around themselves.
Three seaweeds is wrapped around otter A at this point.



D meets F. They wrap seaweed around themselves. A meets F. Since they are already connected, they won't wrap any seaweed around themselves.



Thus, only three seaweeds are wrapped around otter A.

BACKGROUND INFORMATION

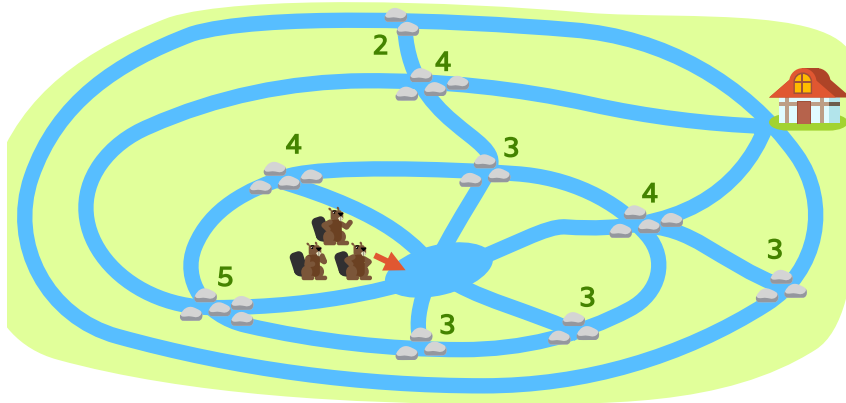
“Disjoint-set” or “union-find” is a data structure that stores a collection of *disjoint* (non-overlapping) sets. It provides operations such as adding new sets and merging sets. In this task, we merge two sets by wrapping seaweed around two otters.

A disjoint-set is an important data structure when implementing *Kruskal's algorithm* to find a minimum spanning tree in a graph. A disjoint-set is used for preventing cycles in the selected edges, which is exactly how we avoid knots in this task.



Collecting Stones

The Barker Beaver's family lodge consists of 21 canals. 31 stones need to be removed. The beavers start in the central pool and go for a swim to collect the stones, bringing them to the storage deposit on the right.



The stones are heavy. A beaver can only carry one or two of them at the same time, but no more. To get from one intersection to the following, it takes each beaver exactly one hour to swim. After starting simultaneously the beavers have to collect all stones within four hours. They also only want to recruit the minimum number of beavers they need to do the task.

Question

What is the minimum number of beavers the Barkers need to recruit to clear the canal within 4 hours?

12 beavers

14 beavers

16 beavers

18 beavers

20 beavers

22 beavers

24 beavers

EXPLANATION

Answer

The correct answer is 14 beavers.

Explanation

There is only one path where the beavers can reach the deposit from the central pool within 2 hours. On this path are 4 stones. 2 beavers can carry these 4 stones to the deposit and may continue to carry 4 additional stones to the deposit.

Then there are $23 = 31 - 8$ stones left, and at least 12 more beavers are necessary. The remaining stones lie on paths which consist of 3-4 intersections. Within 4 hours, no time is left after depositing stones to continue to collect more.

We can observe by inspecting the beaver lodge that the remaining 23 stones can be brought to the deposit by 12 beavers. 11 of them carry 2 stones and 1 of them carries just one stone.

Another way of thinking about this problem is that there are 31 stones, and only two beavers can bring 2 stones each and then return for more. The other beavers can bring at most 2. So at least 13.5 beavers are necessary. Therefore the best solution requires 14 beavers.

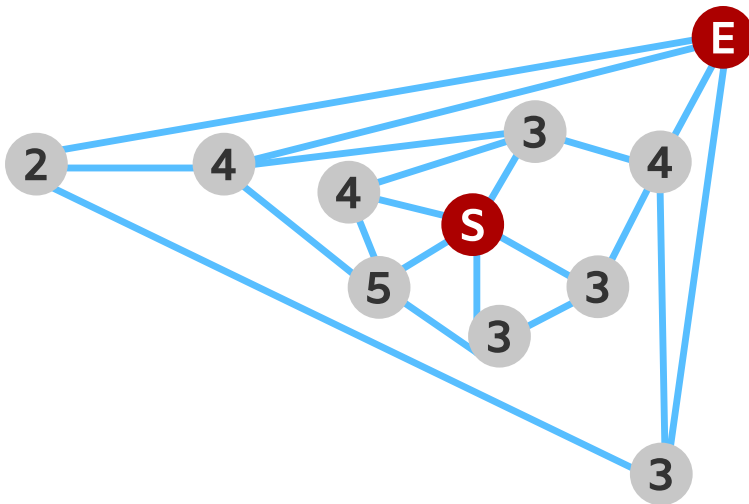
Continued on next page



Collecting Stones - continued

A more detailed explanation is shown below:

At the start, all the stones and the deposit of the beavers lie in canal intersections. In the following picture, we represent canals using straight lines. The numbers in the pictures represent the number of stones on the respective intersection.

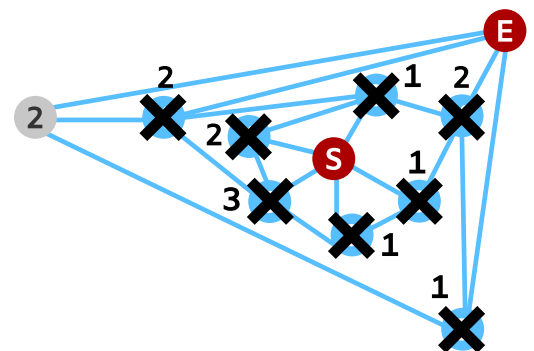


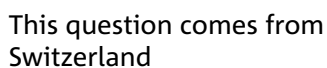
We measure the path length between the central pool and the deposit by counting the number of canals the beavers have to swim through:

- There are no canals that directly connect the starting point and the deposit (i.e., there are no paths of length 1 hour).
- There are paths of 2 hours, of 3 hours, and of 4 hours in length.
- The beavers are required to collect stones at most for four hours. Paths of more than four hours in length do not need to be further considered.

First, we consider all the intersections with at least three stones. In each such intersection, a beaver collects two stones and brings them to the deposit. To this end, we need 8 beavers. They collect 16 stones. $31-16=15$ stones are still left. All the beavers except one were at least three hours on the way. They are not able to go back to the canals and to collect further stones. We can't "reuse" them.

One of the beavers, however, collected two stones within two hours. They would have enough time to get back to an intersection, to collect two additional stones, and to bring them to the deposit. We can therefore "reuse" this beaver at a later point.





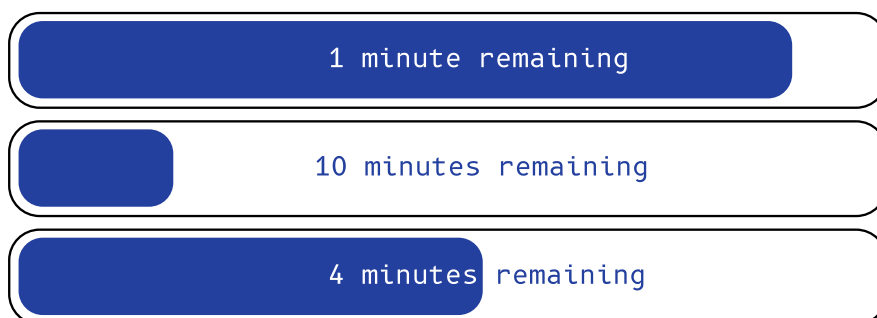
Years 11+12 Easy



Downloads

When downloading files from a server, the download speed is limited. For example, when ten files are downloaded simultaneously, the download speed for each file is ten times slower than it would be for only one file.

A user simultaneously downloads three files from a server. The picture below shows the current download state.



Note that the time remaining for each file is computed based only on the current speed and does not depend on any history.

Question

How many minutes will it take to download all the files?

(Enter your answer as an integer. Remember to press 'Save' when you have finished.)

Save

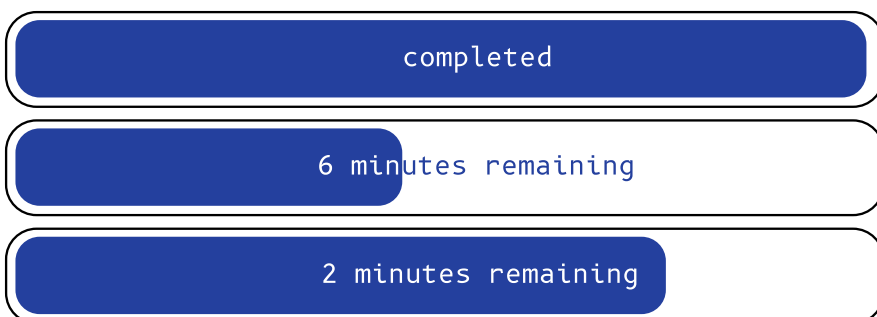
EXPLANATION

Answer

The correct answer is 5.

Explanation

After one minute the first file will be downloaded, the speed will increase by a factor of $3/2$ (that is, the 3 downloading files became 2 downloading files), and the progress will look like:

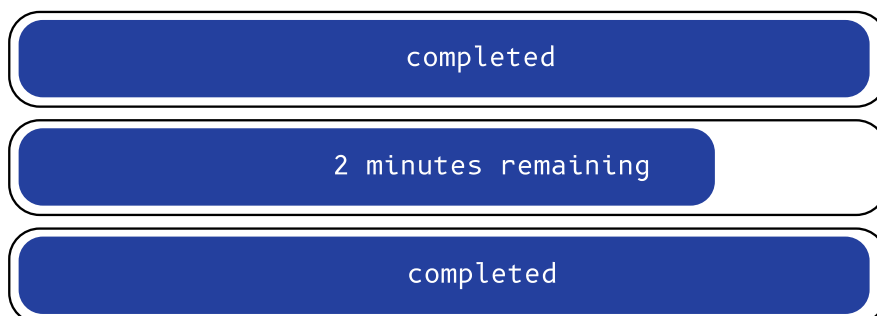


Continued on next page



Downloads – continued

After two minutes more the third file will be downloaded, and the progress will look like:



There are two more minutes needed to download the last file. So, after $1 + 2 + 2 = 5$ minutes all the files will be downloaded.

BACKGROUND INFORMATION

A progress bar is an element of a *visual user interface* (sometimes called a *widget*), that is usually represented as a rectangle or any other area partly filled by a colour or a pattern. The size of a filled part represents a fraction of the time passed from the start of some operation to the total time it would take. Progress bars are used when the remaining time to finish an operation is known at least approximately. This is usually a tricky task to make a plausible estimation of the remaining time.

Many aspects of user interfaces require calculations: as another example, when you resize a window, the contents contained in the window, as well as other elements on the screen, may need to be adjusted, and these adjustments can involve detailed computation. These issues are part of the fields of human-computer interaction or user interfaces.

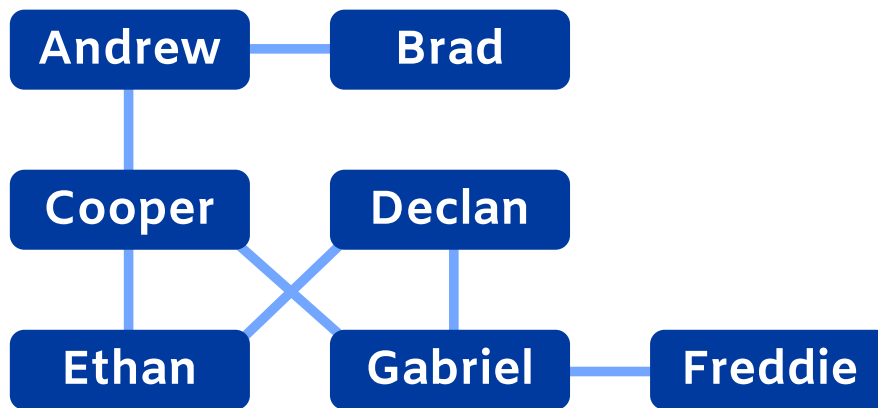


Popularity

Seven beavers are in an online social network called Instadam.

Instadam only allows them to see the photos on their own and their friends' pages.

In this diagram, if two beavers are friends they are joined by a line.



After the summer holidays everybody posts a picture of themselves on all of their friends' pages.

Question

Which beavers' picture will be seen the most?

Andrew	Brad	Cooper	Declan
Ethan	Gabriel	Freddie	

EXPLANATION

Answer

The correct answer is Cooper.

Explanation

In order to find the beaver whose picture gets seen by most beavers, you have to count the beavers that are at most two steps away. The beavers one step away are those on whose page the pictures will be posted and the beavers two steps away are those who can see these pages. Each beaver can only be counted once. The following table shows the names of these beavers starting from each beaver.



Popularity – continued

Beaver	Direct Friends	Friends' Friends (that have not yet been named)	Total number of beavers reached
Andrew	Brad, Cooper	Ethan Gabriel	4
Brad	Andrew	Cooper	2
Cooper	Andrew, Ethan, Gabriel	Brad, Declan, Freddie	6
Declan	Ethan, Gabriel	Cooper, Freddie	4
Ethan	Cooper, Dmitri	Andrew, Gabriel, Freddie	5
Freddie	Gabriel	Cooper, Declan	3
Gabriel	Cooper, Declan, Freddie	Andrew, Ethan	5

BACKGROUND INFORMATION

Many social networks use larger and more complicated versions of this concept, while it is not always obvious that by posting something on someone else's page it might be available to people the original poster does not know or is not a friend of.

Social networks themselves are incredibly powerful tools in today's world. Computing statistics on their users and pages is useful to marketers and anyone else trying to understand a person or group of people.

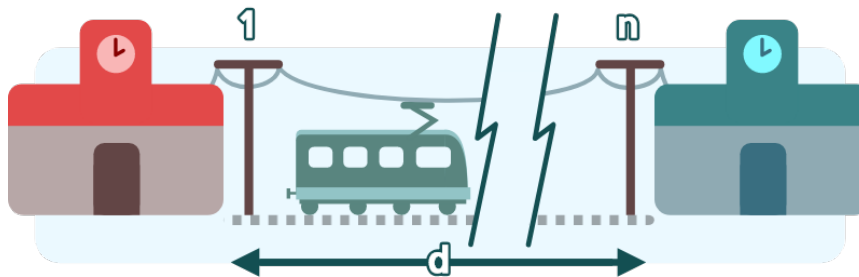
This social network could also be interpreted as a model of a miniature internet, with the beavers being websites and being friend as "linked to". Search engines typically rank these websites by some measure of popularity or importance, at least by the number of links to and from the website.

A widely used way to find the result by using a computer is to use the flood fill algorithm, in this case restricted by the fact that only two iterations have been shown.



Railway Electrification

The railway line between two stations is to be electrified. The mast poles must be placed at regular intervals, the first one at the 1st station and the last one at the 2nd station.



The pole-placing robot works according to the program below.
The distance between stations is d metres; the robot must place n mast.

Question

Drag in the correct bits of code, on the right, into the spaces in the program below so that the pole-placing robot can complete its task correctly.

(Remember to press 'Save' when you have finished.)

define

deploy the masts

drive to the warehouse

load mas

arrive at the 1-st station

place one mast

repeat

move by r

place one mast

n

d

1

2

n+1

d+1

n-1

d-1

n/2

d/2

n+d

n-d

d-n

d*n

n/d

d/n

n/(d+1)

d/(n+1)

n/(d-1)

d/(n-1)

Save

Erase

Continued on next page

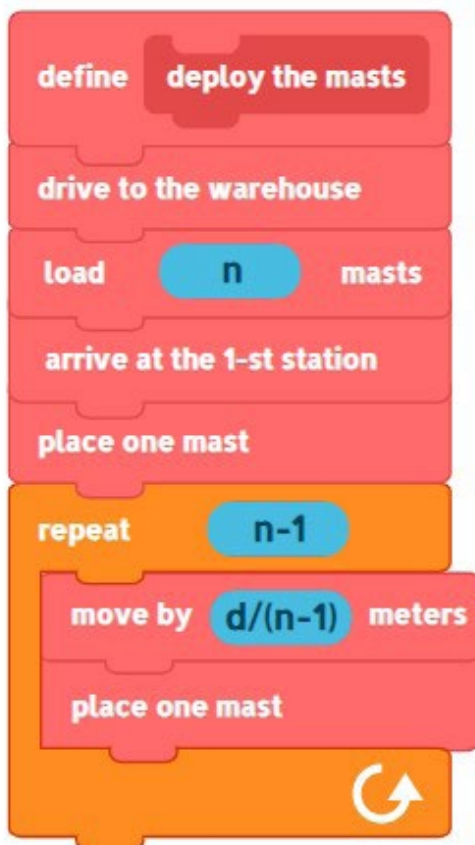


Railway Electrification – cont'd

EXPLANATION

Answer

The correct answer is (from top to bottom) n , $n-1$, $d/(n-1)$



Explanation

If robot must place n masts, it must load n masts.

After placing one mast, $n-1$ masts are left. So the robot must repeat $n-1$ times moving and placing one mast.

Then we must divide the distance d by $(n-1)$ to know how much the robot has to move to place the next mast. Thus $(n-1) * [d/(n-1)] = d$.

BACKGROUND INFORMATION

The piece of code in this task is a procedure.

In this procedure we use variables d and n . A variable is considered as a name for a value, eg. some number. These variables d and n are defined previously in the main program and used in this procedure. These are called global variables because they can be accessed in different procedures of the same program.

The task is about loops too. A loop is a sequence of instructions executed a number of times. In this task it is very important to know how many times the loop must be executed. In our task the loop must be executed $n-1$ times to complete the electrification of the railway line.

Procedures, sequences of program instructions, are powerful programming tools that perform a specific task. Values are passed to and from procedures through variables. The same procedure can work on different tasks by passing different values in the variables.



Tree Farming

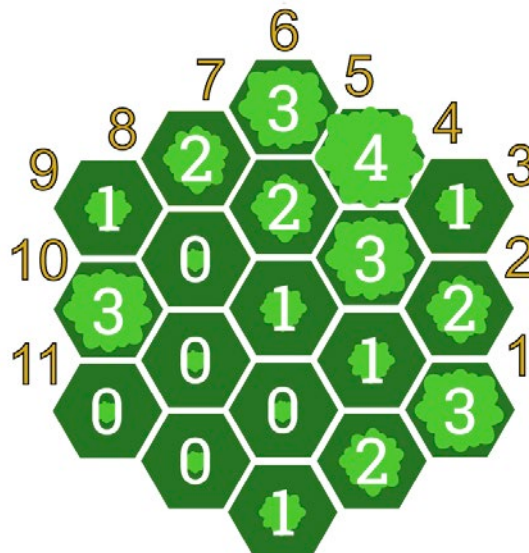
Beaver Dana is surveying the local forests. She has created a map of the trees in the forest on a hexagonal grid so she can work out how much sunlight a given tree will get during the day. The sun moves in the following way:

- The sun starts every morning in the position shown in the left image below.
- Every hour, it moves counter-clockwise around the hex grid by one side. One hour after sunrise it is located in the position shown in the middle image.
- It continues moving one side every hour until sunset, when it is located in the position shown in the right image.



The sun shines in a straight line along any sides it is touching, as shown above.

The map Dana creates looks like this:



The following rules are true:

- The height of a tree is given by a number.
- A tree will block the sun from reaching a number of hexes behind it equal to its height. (A tree of height 2 will block the sun from trees in the 2 hexes behind it.)
- However, a tree will not block the sun from reaching a taller tree. (A tree of height 2 will *not* block the sun of a tree of height 3, even if it is within 2 hexes.)
- There are 11 hours of sunlight every day, including sunrise and sunset.
- The position of the sun at each hour is shown by the numbers on the hexagons' borders.

A tree needs to get *at least* 3 hours worth of sunlight every day so that it can grow.

Continued on next page

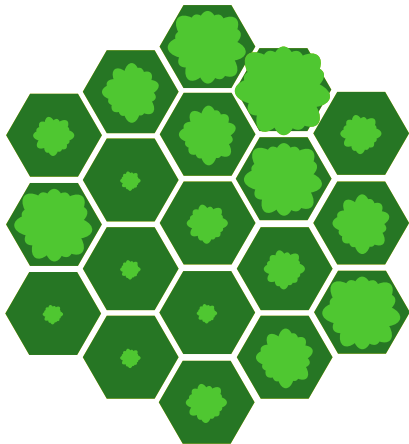




Tree Farming – continued

Question

Which trees will get enough sunlight to grow in a given day?
(Select all of the trees that will get 3 or more hours of sunlight in a given day)

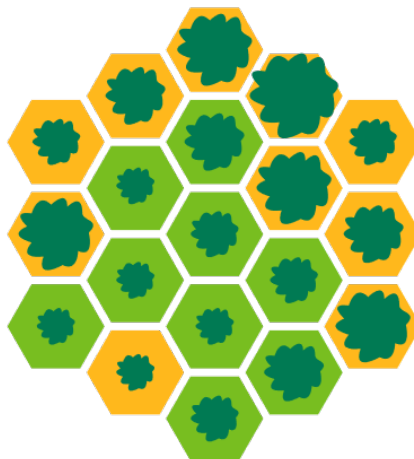


Save

Erase

EXPLANATION

Answer



Explanation

For this question, any given tree on the map will need to get 3 hours of sunlight to grow. To determine whether a tree will get 3 hours of light, we need to establish which angles the tree will be able to get sunlight from.

One way to do this would be to count each tree every hour, but this can be time consuming.

Given that the sun only travels in an arc along the right, top, and left sides of the map, we only need to consider the sides of the hexagons that will receive sunlight from those directions.

By looking at each tree, if we can establish for each of those lines that at least 3 have unbroken lines to the sun, we can be certain that this tree will get the 3 hours needed to grow.

Further, for all of the hexagons below the centre hex, there are only three possible directions that they have a line to the sun at any point during the day. If any of these three lines are broken at all, we know that the tree will not grow.

Continued on next page



Tree Farming – continued

BACKGROUND INFORMATION

This task shows a basic method of *modelling* a complex system, specifically the amount of light shining into a forest. Computer scientists often use *algorithms* to create models of real-world situations, so that we can better understand the world around us. These models often cannot be exactly perfect, but are intended to be as close an approximation as possible.

We can then use these models to test hypothetical scenarios, which is called *simulation*. Examples of this might include modelling weather, or creating a model of the terrain of a city, then using that to examine how large amounts of rainfall will impact the waterways in that city. This helps to answer questions like “will it flood?”, and “will it drain away through safe channels?”

When creating models, computer scientists often have to determine which aspects of the real-world situation they can simplify, and which need to be as detailed as possible. In this example, the sun operates in a very simple way, compared to the way the sun functions in real life. This has been simplified in order to make the model easier to create and use.



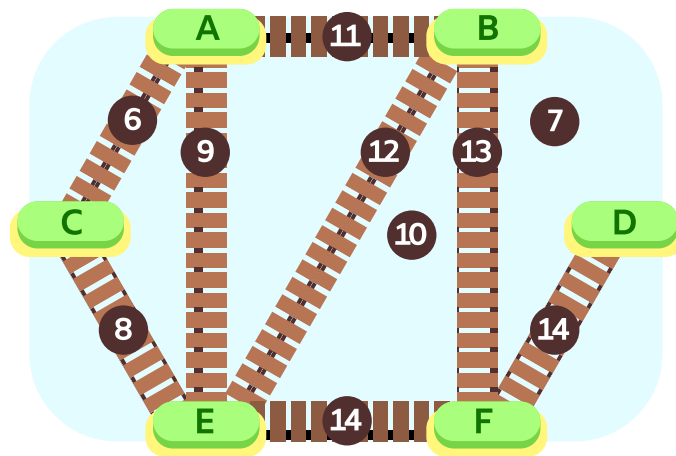
Connection of Islands

A jungle community, living on six islands, wants to connect these islands by building a network of canopy bridges.

A plan of all possible bridges was created. The numbers show the cost to build each bridge, for each possible connection. The bridges do not intersect with each other.

The community wants to link all the islands so that it is possible to travel from any island to any other island. Travel between islands can either be done directly, or by going indirectly through one or more islands.

At the same time, the community wants to build the bridges as cheap as possible.



Question

What is the cheapest whole number that can be spent to link up all six islands?

(Type a whole number below. Remember to press 'Save' when you have finished.)

Save

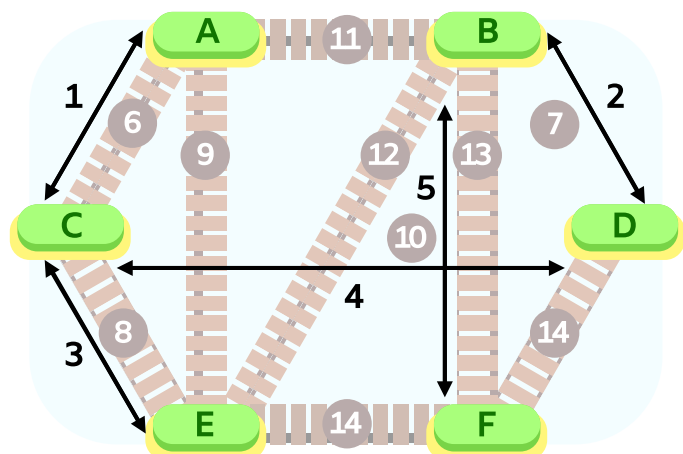
EXPLANATION

Answer

44

Explanation

Counting: $6 + 7 + 8 + 10 + 13 = 44$



Continued on next page



Connection of Islands – cont'd

The algorithm used to solve this problem is called Kruskal's algorithm:

1. Start from the cheapest bridge AC – with a cost of 6.
2. Then we choose the second cheapest bridge – BD (7).
3. The third cheapest bridge is CE (8).
4. The next cheapest bridge has a cost of 9. However if we build it, we get a circuit consisting of AECA. It means that we do not need this bridge at all – we can reach islands A, C, and E without building this bridge.
5. The fourth bridge to build is CD (10).
6. The next bridge which costs (11) is a circuit again: ABDCA. Therefore we do not need to build this bridge.
7. In similar way, the bridge BE (12) also makes a circuit, and is unnecessary: ACDBEA.
8. The fifth bridge that we need to build is BF (13).

Now we have all six islands connected. At each step we have chosen the cheapest possible bridge.

BACKGROUND INFORMATION

The given problem requires us to find a minimum *spanning tree* for a given graph. A spanning tree is a subgraph of a given graph which has exactly the same vertices and some of the edges. In this problem, the subgraph is the selection of bridges from the total bridges possible - the islands are the vertices and the bridges are the edges.

Kruskal's algorithm always finds a minimum spanning tree (or shortest path) for a connected weighted graph. The algorithm is based on selecting edges one by one, with the smallest weight, avoiding circuits with already chosen edges.



Perfect Partners

Andy, Bert, Chris, David and Eric are professional male ballroom dancers that take part in a TV show.

Amy, Brenda, Carol, Dianna and Emma are female contestants that will learn to dance during this show.

Each professional will be assigned a single contestant to teach.

Before the show, the producer organises a party where everybody meets.

After the party the professionals and contestants fill out a questionnaire:

- Each professional ranks the contestants in the order that he thinks they can be successful.
- Each contestant ranks the professionals in the order of how fast she thinks she can learn from him.

(1 = 1st choice, 2 = 2nd choice, etc.) Here are the results of these choices:

Professional dancers' preferences

	Amy	Brenda	Carol	Dianna	Emma
Andy	1	3	2	5	4
Bert	1	2	3	4	5
Chris	2	1	4	5	3
David	5	4	3	2	1
Eric	4	5	2	3	1

Contestants' preferences

	Andy	Bert	Chris	David	Eric
Amy	4	3	5	2	1
Brenda	3	4	1	2	5
Carol	2	4	1	5	3
Dianna	5	2	3	4	1
Emma	5	2	3	1	4

The producers want to match the professionals with the contestants in such a way that there will not be any envy.

You are asked to match the professionals with the contestants so that every beaver has a partner. You must also ensure that all unmatched beaver pairs would not be happier if they matched together.

Question

When you have finished assigning partners, who is Eric's partner?

Amy

Brenda

Carol

Dianna

Emma



Perfect Partners - cont'd

EXPLANATION

Answer

The correct answer is:

	Amy	Brenda	Carol	Dianna	Emma
Andy	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bert	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chris	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Dave	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Eric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Explanation

The solution above is produced by Gale-Shapley algorithm:

Round 1:

Am asks Af

Bm asks Af

Cm asks Bf --> provisional Y

Dm asks Ef --> provisional Y

Em asks Ef

	Amy	Brenda	Carol	Dianna	Emma
Andy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bert	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chris	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Dave	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Eric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Round 2:

Am asks Cf --> provisional Y

Bm asks Bf

Em asks Cf

	Amy	Brenda	Carol	Dianna	Emma
Andy	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bert	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chris	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Dave	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Eric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Round 3:

Bm asks Cf

Em asks Df --> provisional Y

	Amy	Brenda	Carol	Dianna	Emma
Andy	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bert	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chris	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Dave	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Eric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Round 4:

Bm asks Df

Round 5:

Bm asks Ef

Round 6:

Bm asks Af --> provisional Y

	Amy	Brenda	Carol	Dianna	Emma
Andy	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bert	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chris	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Dave	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Eric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

BACKGROUND INFORMATION

Although there are not many problems that require this solution the few that do are important e.g. assigning trainee doctors to their preferred hospital practice. A very important example in Computer Science is assigning users to servers in a large distributed Internet service.

As such there has been a lot of research into this and the Gale-Shapley algorithm is commonly used.



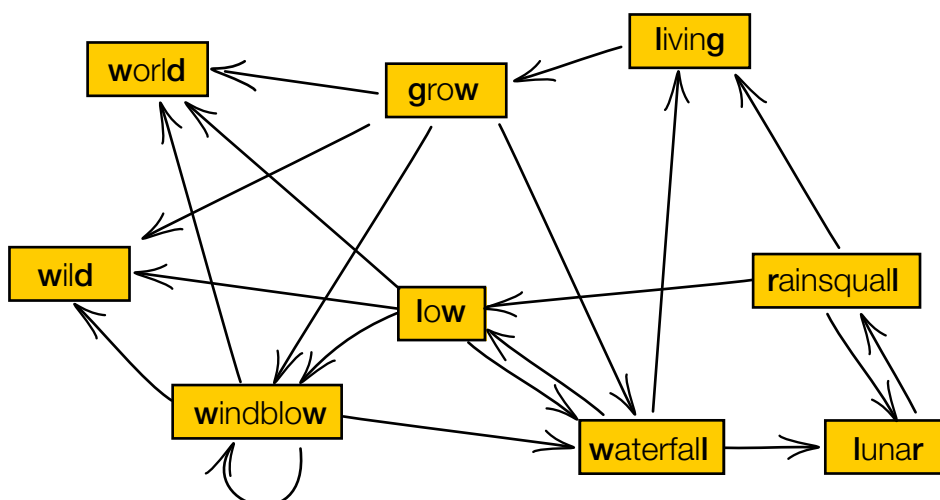
Longest Word Chain

Beavers often play a two-player word chain game.

Game Rules:

- Place nine word cards face up in front of both players.
- Player one starts by saying one of the words.
- Player two says another word that begins with the last letter of the previous word.
- Player one says another word that begins with the last letter of the previous word and which has not been used yet.
- Play continues in this way until there is no new word available.

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



Question

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

(Enter your answer in the form of an integer. Remember to press 'Save' when you have finished.)

Save

EXPLANATION

Answer

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

Continued on next page



Longest Word Chain – continued

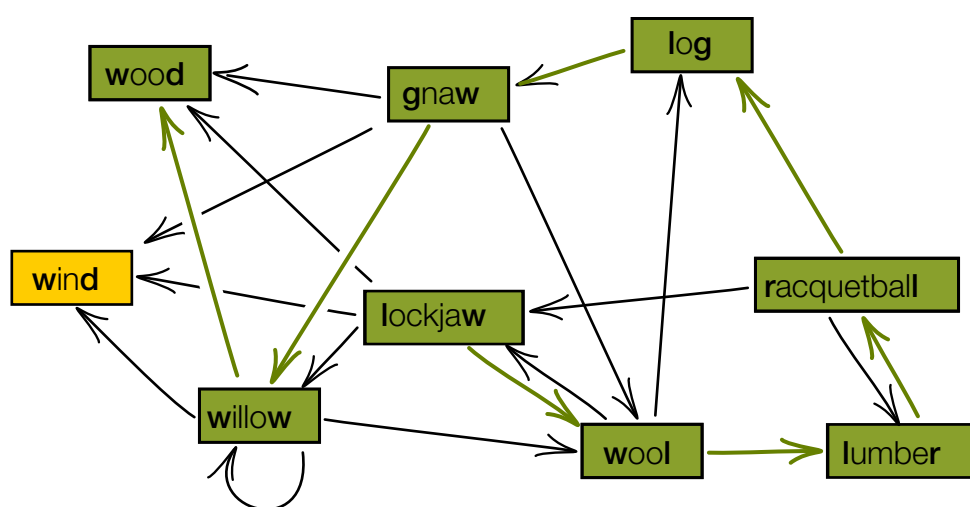
Explanation

One example is:

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

Can you find another word chain of the same length?

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



BACKGROUND INFORMATION

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

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



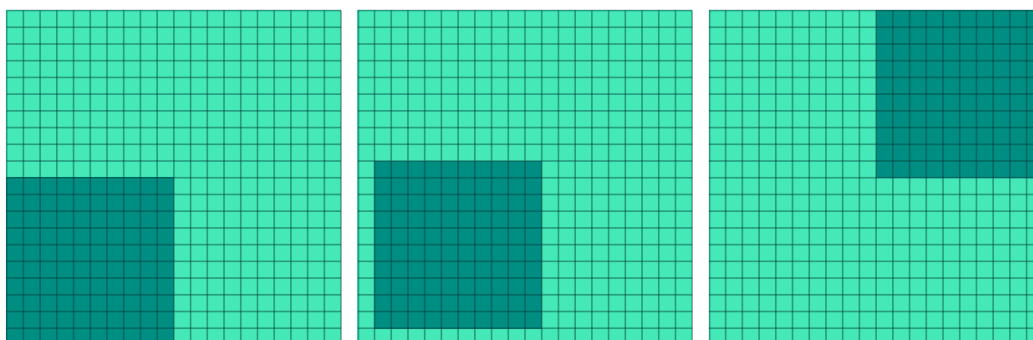
Video Compression

A computer image is a rectangular grid of coloured squares, called *pixels*. A video is a sequence of images, called *frames*, each slightly different from the previous one.

The simplest way to store a video is to store all the pixels in each frame.

A more efficient way is to store the entire first frame and then only store those pixels that change from the current frame to the next one.

In the picture below, the 10x10 dark coloured square moves from the lower-left corner to the upper right corner of the light coloured 20x20 field, moving one pixel horizontally and vertically in each frame. This takes 11 frames. If we store this video in the simple format, this will require $(20 \times 20) \times 11 = 4400$ pixels.



Question

How many pixels will you need to save these 11 frames in the more efficient format described above?

Save

EXPLANATION

Answer

780.

Explanation

We need to remember $20 \times 20 = 400$ pixels for the first frame and 38 changing pixels for each of the next frames.

BACKGROUND INFORMATION

This task is devoted to the idea of lossless compression, when new frames are considered as slices of a three-dimensional array and the change of the picture on the screen occurs only in those pixels that differ in adjacent slices-frames.

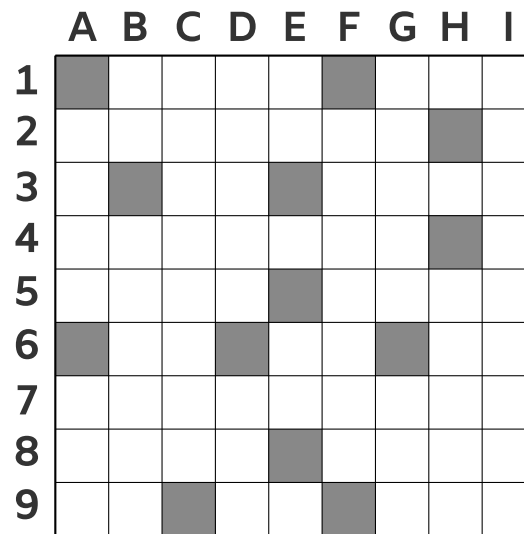
We should keep in mind that in the algorithm presented, we have to remember not only the colour of changed pixel, but its coordinates as well. If all those parameters are considered as one-byte information, we need three times as many bytes than in the answer for all frames, except the first one which is $400 + 380 \times 3 = 1540$ bytes. However, it is still better than remembering each frame.

Continued on next page



Robot's Path

A drone starts on a white cell of the grid shown. It faces one of four possible directions.



Then it visits exactly eight other white cells as follows:

1. Move two cells forward.
2. Turn 90 degrees left (in its current cell).
3. Move four cells forward.
4. Turn 90 degrees right (in its current cell).
5. Move two cells forward.

Question

How many possible starting cells are there?

(Enter your answer as an integer. Remember to press 'Save' when you have finished.)

Save

EXPLANATION

Answer

The answer is 8.

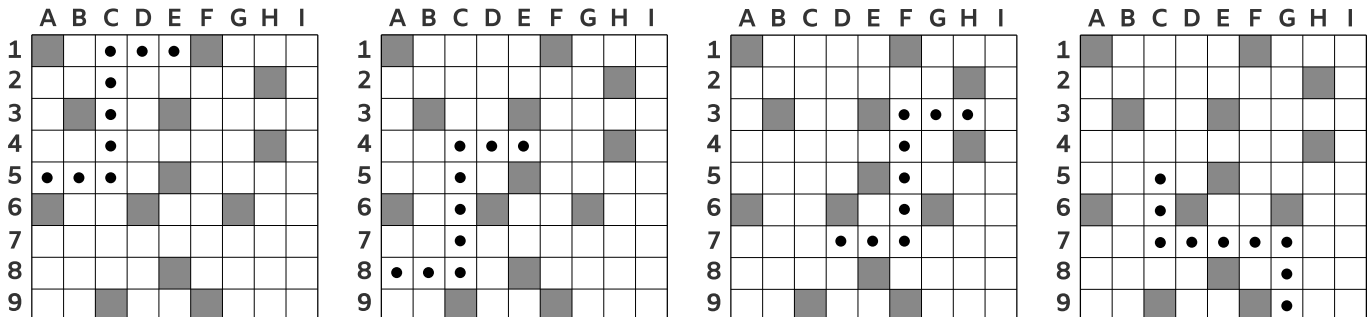
Continued on next page



Robot's Path – continued

Explanation

To see this, first note that there are four possibilities for the cells visited by the drone:



To find these possibilities, it is probably easiest to begin with a search for five consecutive white cells and then try to extend the endpoints in opposite perpendicular directions. To be certain that you are correct, you should do this systematically (e.g. from the top left to the bottom right of the grid).

The path taken by the drone is symmetric, so the drone could start from either endpoint of each of these four possibilities. Therefore, there are eight different possible starting cells.

BACKGROUND INFORMATION

The task is a simple example of rotation invariant image (or template) matching, which is one task within the wider computer science field of image processing. In this question the template is steps 1 through 5 and the test grid is the search image. Template matching has a number of applications and has been used in facial recognition and medical image processing. In some of the template matching methods, the search pattern is often described as a path.

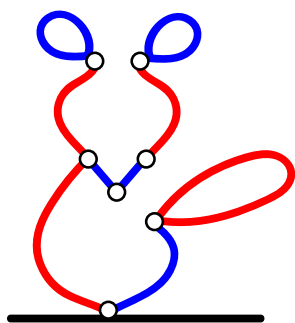


A Game of Cut and Mouse

Two friends, Bob and Ralph, are playing a game.

They start by drawing a thick black line at the bottom of the paper, calling it the *ground*.

Afterwards, they draw several blue and red segments, creating the mouse-shaped figure shown below:



The rules of the game are as follows:

- They take turns to cut any segment of their choice. However, Bob is only allowed to cut *blue* segments while Ralph is only allowed to cut *red* segments.
- Cutting a segment removes that segment and all other segments that are no longer connected to the ground.
- The first player who no longer has any segment to cut is considered the loser.

A possible sequence of moves is given in the table below. Two figures are displayed for each turn: the first figure marks the segment that the player intends to cut while the second figure displays the result of cutting this segment.

Bob's Turn	Ralph's Turn	Bob's Turn	Ralph's Turn

Since Bob no longer has any segment to cut, he loses the game, and Ralph is declared the winner.

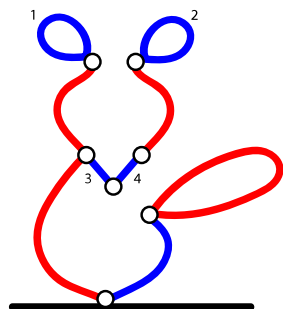


A Game of Cut and Mouse-cont'd

Question

If Bob is the first to move and he always plays the *best* possible move at each turn, which segment should he cut first to guarantee his victory — no matter what move Ralph makes?

Refer to the figure below for the numbering of the segments:



Segment 1

Segment 2

Segment 3

Segment 4

Bob has no chance of winning

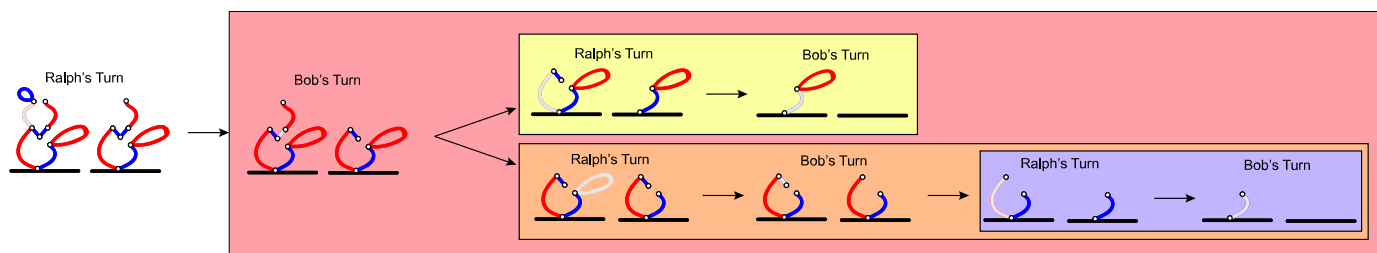
EXPLANATION

Answer

The correct answer is Segment 2.

Explanation

To prove that cutting Segment 2 guarantees Bob's victory (provided that he continues to play the best move at each turn), we start by selecting one of Ralph's possible responses to this opening move. From there, we systematically play through complete games until a winning sequence of moves (or a *strategy*) is found. The diagram below shows a winning strategy if Ralph cuts the left half of the "face":



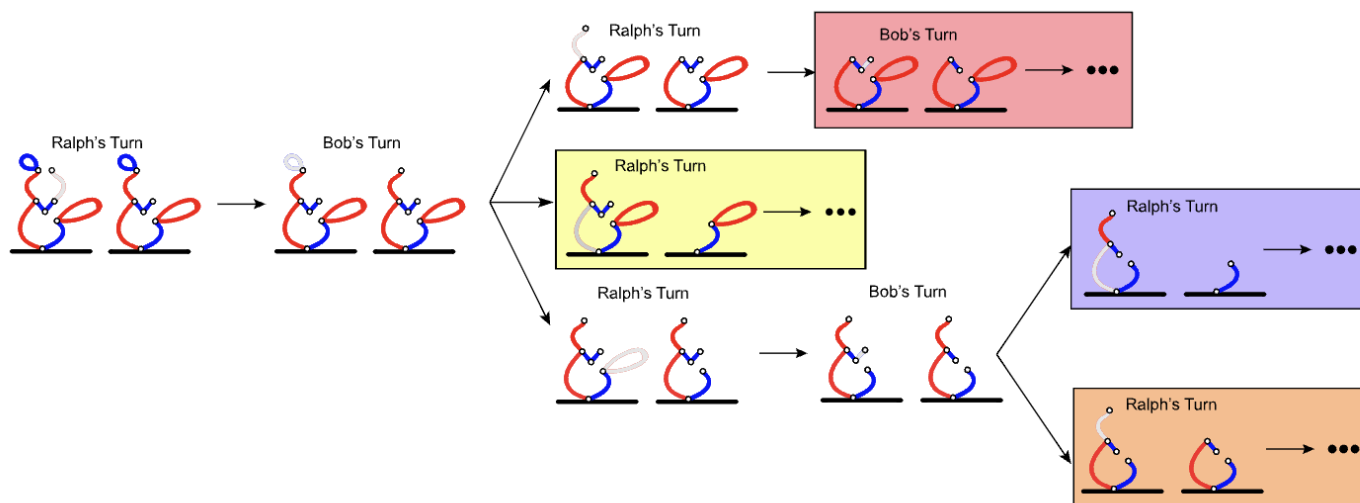
To avoid playing blindly, it is advantageous for Bob to *force* Ralph into a position that is present in the winning strategy above — since this will allow Bob to secure victory by employing the same sequence of moves as in this strategy. In the following diagrams, these positions are marked using color-coded



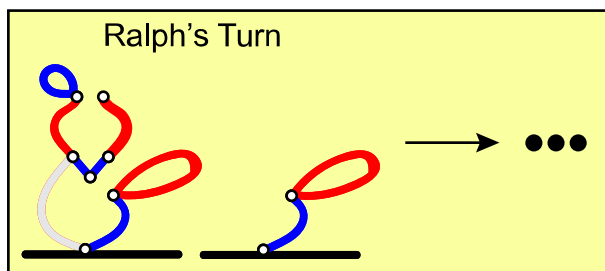
A Game of Cut and Mouse-cont'd

boxes, illustrating how the said idea can be applied to counter Ralph's remaining responses to Bob's opening move:

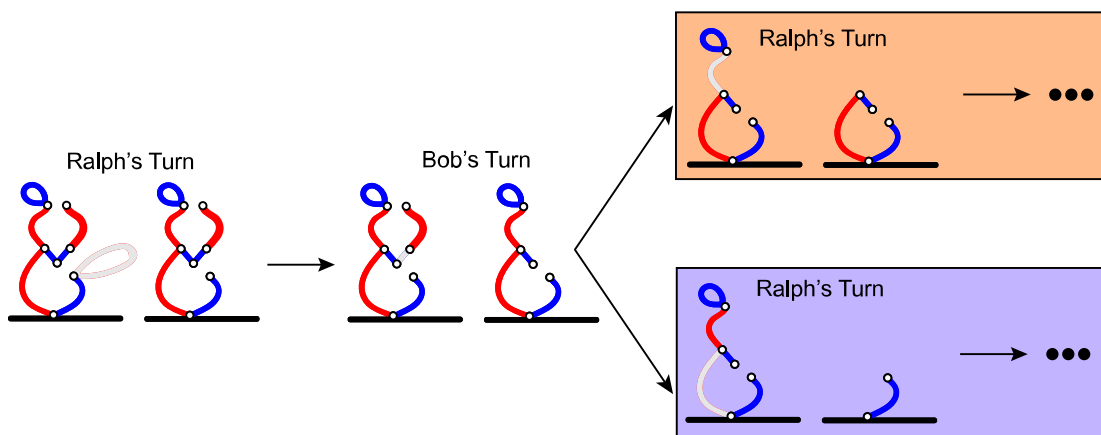
- Ralph cuts the right half of the "face"



- Ralph cuts the left half of the "body"



- Ralph cuts the "tail"

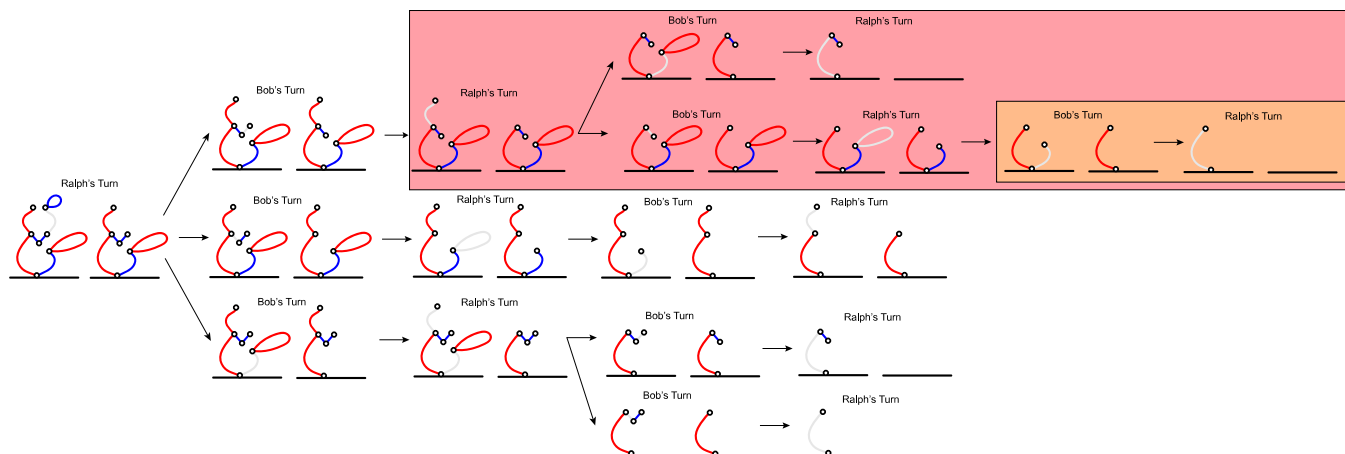




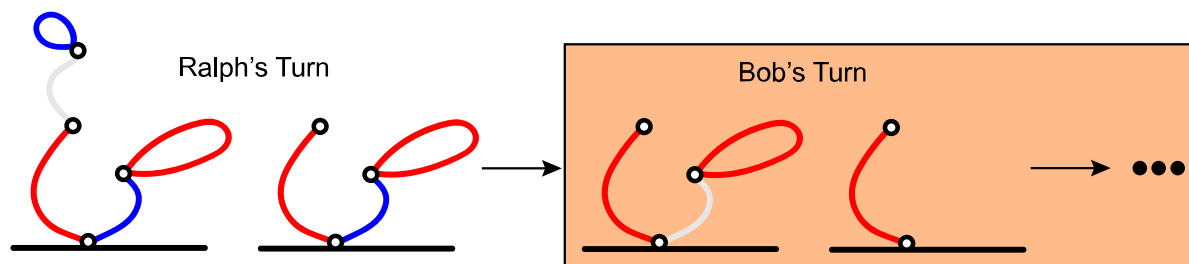
A Game of Cut and Mouse-cont'd

For completeness, it can also be shown that Bob has no chance of winning if he starts by cutting any other segment and Ralph finds the perfect counterplay at each turn:

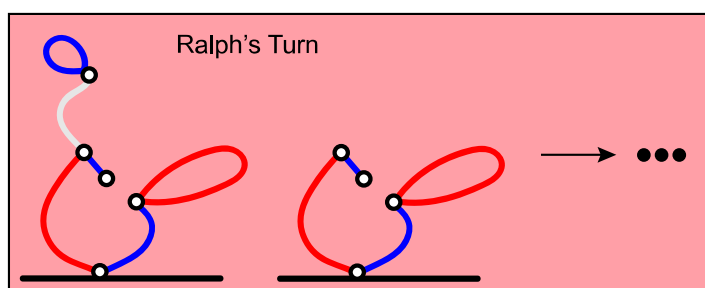
- Bob starts by cutting Segment 1



- Bob starts by cutting Segment 3



- Bob starts by cutting Segment 4





A Game of Cut and Mouse-cont'd

BACKGROUND INFORMATION

The game featured in this challenge, known as *Hackenbush*, was introduced by the English mathematician John H. Conway in the early 80s. It belongs to a special category of games referred to as *combinatorial games*; two-player combinatorial games satisfy three criteria: (1) they are played *sequentially* (the players take turns), (2) they are *deterministic* (they are not influenced by randomizing devices, such as roulettes or dices), and (3) players always have *perfect information* (no information is hidden or withheld from them). Popular examples include chess, checkers, tic-tac-toe (noughts and crosses), Nim, and Hackenbush.

Combinatorial games are of great value to informatics, especially to theoretical computer science and artificial intelligence. Areas of interest include determining the winner if both parties always play the best possible move (*perfect play*), formulating strategies to maximize the chances of victory, and creating *heuristics* (estimates) to assess which player has the advantage in a given position.

The simplicity of Hackenbush belies deep concepts that spring from its study; most notably, evaluating its positions is tied to a new class of numbers called *surreal numbers* (invented by Conway and popularized by computer scientist Donald Knuth). A deeper analysis shows that the mouse-shaped figure in this task actually favors Bob (blue) — regardless of who moves first, Bob is guaranteed to win assuming perfect play. Hackenbush is also described as a *cold game*, that is, making a move can only worsen a player's position.

Game trees are constructed to enumerate the possible positions at each turn. However, games can quickly grow complex, which makes generating the entire game tree inefficient or infeasible. To this end, algorithms have been devised to avoid exploring the same positions twice (as we did in our solution) or to “prune” (exclude) positions whose evaluations fall below a certain threshold. In modern artificial intelligence, techniques such as *reinforcement learning*, where a computer is trained by “learning from its mistakes,” are being explored to improve decision-making for complex games, like chess and Go.

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

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

Contact us

CSIRO Digital Careers

digitalcareers@csiro.au

csiro.au/education/Programs/Digital-Careers