Bebras PT

Edição 2024

Categoria
Seniores (11º e 12º ano de escolaridade)

Tempo
45 minutos

Resolve tantos problemas quanto possível em 45 minutos.

Não é esperado que consigas resolver todos!

Responde apenas na folha de respostas.

É uma folha única, à parte, que deverás identificar com o teu nome.

Os enunciados e folhas de rascunho devem ser obrigatoriamente recolhidos no final da prova.

O **Bebras** é uma iniciativa internacional destinada a promover o pensamento computacional e a Informática (Ciência de Computadores). Foi desenhado para motivar alunos de todas as idades mesmo que não tenham experiência prévia.

Esta iniciativa começou em 2004 na Lituânia e todos os anos participam mais de 3 milhões de alunos de todo o mundo. O seu nome original vem dessa origem - "bebras" significa "castor" em lituano. A comunidade internacional adotou esse nome, porque os castores buscam a perfeição no seu dia-a-dia e são conhecidos por serem muito trabalhadores e inteligentes.

O que é o Pensamento Computacional?

O pensamento computacional é um conjunto de técnicas de resolução de problemas que envolve a maneira de expressar um problema e a sua solução de modo a que um computador (seja um humano ou máquina) a possa executar. É muito mais do que simplesmente saber programar. O desafio do Bebras promove precisamente este tipo de habilidades e conceitos como a capacidade de partir um problema complexo em problemas mais simples, o desenho de algoritmos, o reconhecimento de padrões ou a capacidade de generalizar e abstrair.

Organização Portuguesa

O Bebras começou em **Portugal** em 2019 e ano passado contou com a participação de 105 150 alunos, de 683 escolas de Portugal, Angola, Moçambique, São Tomé e Príncipe e Timor-Leste.

É organizado por uma equipa de pessoas ligadas à Educação e à Ciência de Computadores da **TreeTree2** e do Departamento de Ciência de Computadores da Faculdade de Ciências da Universidade do Porto (**DCC/FCUP**)

Estrutura da Prova

Existe apenas uma fase a nível nacional, a qual é constituída por uma prova individual com 12 questões de três níveis de dificuldade diferentes, cuja pontuação é da seguinte forma:

Dificuldade	Correto	Incorreto	Não respondido
fácil	+6 pontos	-2 pontos	0 pontos
média	+9 pontos	-3 pontos	0 pontos
difícil	+12 pontos	-4 pontos	0 pontos

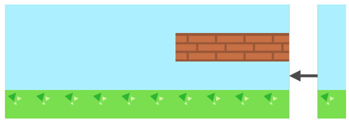
Sobre os Problemas

Os problemas da edição portuguesa foram escolhidos, traduzidos e adaptados pela organização portuguesa. Para a deste ano foram usados problemas com autores originários dos seguintes países (além de Portugal):

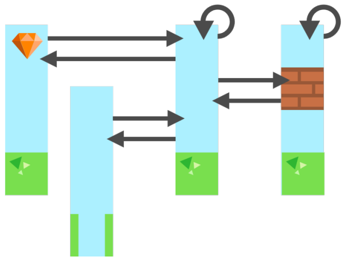
Alemanha	Austrália	Bélgica	Canadá	Chéquia
Coreia do Sul	Eslováquia	Estónia	Hungria	Índia
Irlanda	Japão	Lituânia	Macedónia	Malásia
Paquistão	Reino Unido	Suiça	Taiwan	Ucrânia

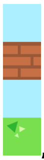
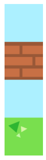
1. Superbebras

Num jogo de computador, o fundo é constituído por uma sequência de peças. O computador acrescenta constantemente uma nova peça à direita da sequência e, simultaneamente, retira uma peça da esquerda. Desta forma, o computador cria a ilusão de movimento.



O computador seleciona uma nova peça a adicionar ao fundo utilizando o diagrama à direita. Olha para a peça anterior e verifica as setas que saem dessa peça. Em seguida, seleciona aleatoriamente (*"à sorte"*) a peça para a qual uma das setas está a apontar.

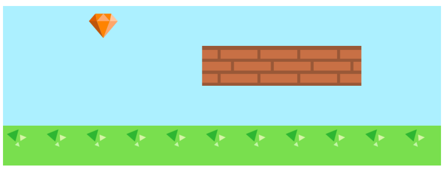


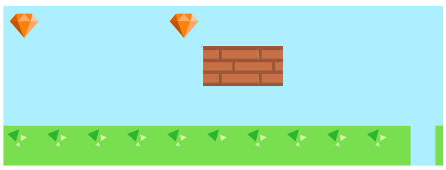
Por exemplo, depois da peça , o computador pode escolher a peça  ou a peça .

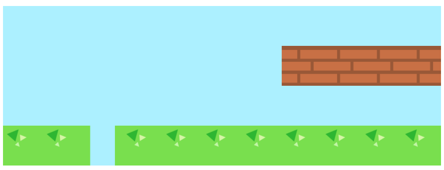
Pergunta

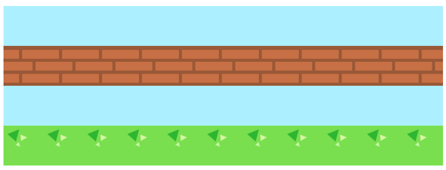
Qual das imagens **não** *é um fundo válido neste jogo?*

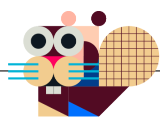
Respostas possíveis

(A) 

(C) 

(B) 

(D) 



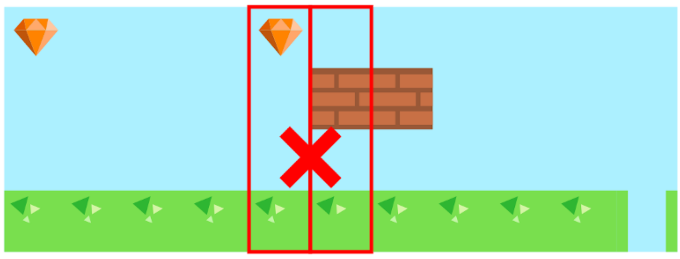
1. Superbebras (Resolução)

Solução

(C)

Resolução

O fundo apresentado na resposta (C) não está completamente construído de acordo com o diagrama.



Uma peça com um diamante, só pode ser seguida por , o que não acontece aqui.

Há várias formas de encontrar a solução para este desafio. Uma estratégia simples é experimentar cada imagem de fundo dada e verificar se cada peça é válida em relação ao diagrama. A forma mais rápida é esta: Verificam-se os mosaicos no diagrama, começando pelos mosaicos dos quais sai apenas uma seta. Pensa nestas setas como restrições. A → B significa que "a peça A deve ser seguida pela peça B". De seguida, verifica se esta restrição é válida para cada imagem de fundo.

Isto é Pensamento Computacional!

Alguns jogos de computador têm um fundo que se desloca horizontalmente, criando a ilusão de que o jogador se move num mundo de fantasia. Por vezes, o fundo não é uma imagem constante pré-determinada, mas é antes criado de forma dinâmica e automática pelo computador. A isto chama-se geração processual. Normalmente, os elementos mais pequenos são combinados usando aleatoriedade para criar uma enorme variedade de fundos. No entanto, a combinação de elementos não pode ser completamente aleatória, devendo antes obedecer a determinadas regras para criar mapas que sejam realistas e evitar situações que o jogador não consiga resolver.

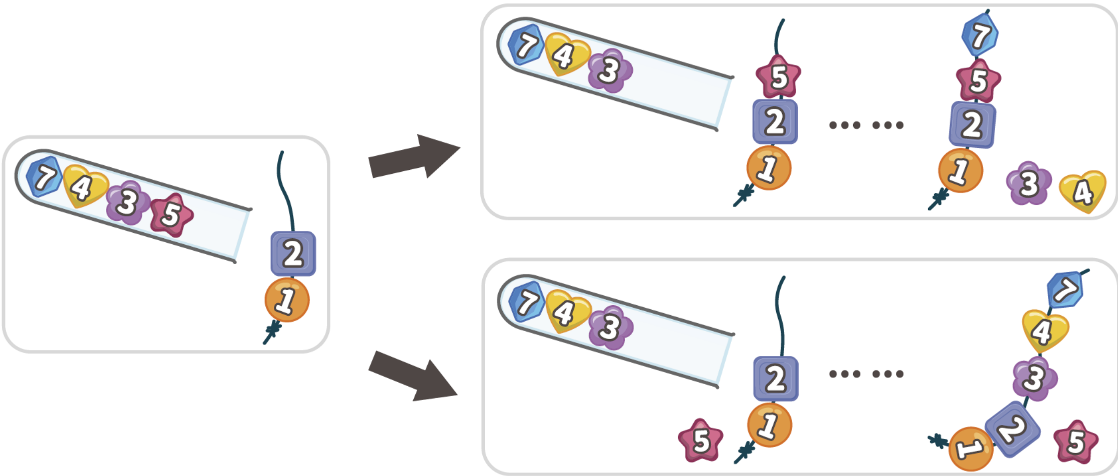
Neste problema, essas regras são representadas por um diagrama constituído por peças e setas. Este diagrama é designado por grafo dirigido. Os grafos são utilizados para todos os tipos de modelação de problemas.

2. A Pulseira do Frederico

O Frederico está a fazer uma pulseira. Ele tira missangas numeradas de um tubo, uma a uma. Por cada missanga, ele escolhe se quer pô-la no fio ou se quer pô-la de parte e não a usar. Mas ele pode pôr a missanga no fio apenas se:

- o fio está vazio, ou
- o número na missanga é maior do que o número da última missanga no fio.

Neste exemplo, a última missanga no fio tem o número 2. Assim, o Frederico pode pôr a missanga com o número 5 no fio ou então pô-la de parte (sem a usar).



O Frederico está a fazer uma nova pulseira com as missangas deste tubo:

Pergunta

Qual é o número máximo de missangas que o Frederico pode pôr no fio?

Respostas possíveis

(A) 3 missangas

(B) 4 missangas

(C) 5 missangas

(D) 6 missangas

(E) 7 missangas

2. A Pulseira do Frederico (Resolução)

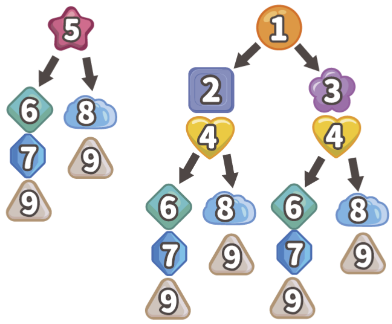
Solução

(D)

Resolução

A resposta correta é a (D) 6 missangas.

Podemos simular todas as pulseiras que podem ser feitas neste caso e ver quais têm o maior número de missangas.



A primeira missanga do tubo tem o número 5. Se for colocada no fio, apenas as missangas 6, 7, 8 e 9 podem ser colocadas no fio a seguir, com duas sequências possíveis de missangas que podem estar no fio no final: 5679 ou 589. Se a missanga 5 for posta de lado e a missanga seguinte (1) for colocada primeiro no fio, existem mais sequências possíveis, cada vez mais longas, como se mostra na figura de cima do lado direito.

Se qualquer outra missanga for colocada no fio primeiro, a sequência final mais longa possível já está incluída numa das anteriores, começando com a número 1 ou a número 5. Por exemplo, se colocarmos primeiro a missanga nº 2 no fio, obtemos sequências como 24679 ou 2489, ambas incluídas numa das sequências mais longas acima (124679 e 12489, respetivamente).

A partir daqui, descobrimos que as pulseiras com o maior número de missangas do tubo dado começam com a missanga 1 e consistem em 6 missangas cada: 124679 e 134679.

Isto é Pensamento Computacional!

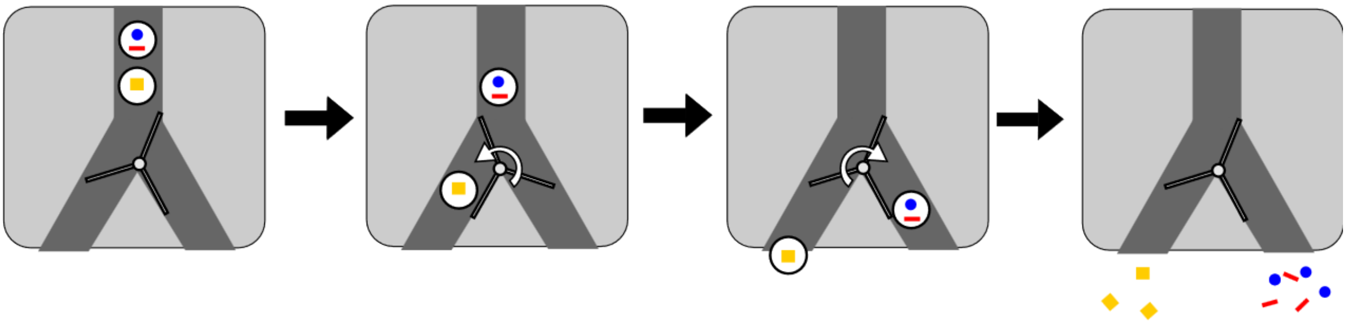
Em informática, uma "sequência"refere-se à organização de dados numa série, como uma sequência de caracteres (também conhecida como *string*) ou uma sequência numérica. Cada membro da sequência é chamado um elemento, e uma subsequência é formada pela seleção de elementos da sequência original original sem alterar a ordem desses elementos.

Tanto as missangas no tubo (na ordem em que estão a sair do tubo) podem ser consideradas sequências dos números nas missangas, se outras propriedades como a forma e a cor forem ignoradas. Esta tarefa pede para encontrar a maior sequência crescente da sequência do tubo, ou seja, a maior sequência com números que vão sempre aumentando.

Para sequências que são (muito) mais longas do que a sequência do tubo neste problema, tentar passar por todas as subsequências crescentes possíveis para encontrar as mais longas demora normalmente demasiado tempo para ser viável. Felizmente, os cientistas informáticos desenvolveram algoritmos eficientes para encontrar rapidamente as subsequências crescentes mais longas. Para isso, usam uma técnica conhecida como *programação dinâmica*, que na sua essência passa por reduzir um problema a problemas semelhantes mais pequenos e depois guardar as soluções desses problemas para evitar repetir cálculos.

3. Padrões Artísticos

Foi criada uma máquina para produzir padrões artísticos num chão quando visto de cima. Cada bola contém uma forma de padrão diferente e segue a direção permitida pelos portões. Quando a bola passa por um portão, este muda automaticamente e envia a bola seguinte na direção oposta.



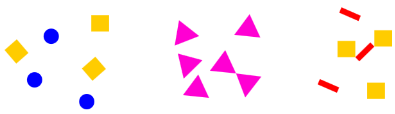
O exemplo mostra o portão aberto à esquerda, a primeira bola a ir para a esquerda e o portão a mudar para enviar a bola seguinte para a direita. Esta segunda bola faz com que o portão volte a mudar.

Cada bola está identificada com uma imagem que representa a forma que irá criar.

Se bolas diferentes saírem da máquina no mesmo espaço, as formas serão distribuídas no chão. Se duas bolas iguais caírem no mesmo espaço, o resultado será o equivalente a se houvesse apenas uma.

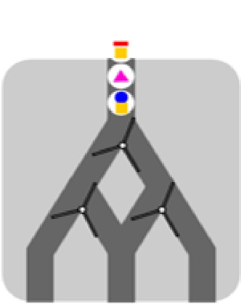
Pergunta

Que sequência de bolas vai criar o seguinte padrão no chão?

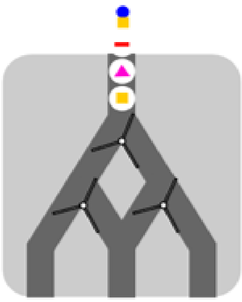


Respostas possíveis

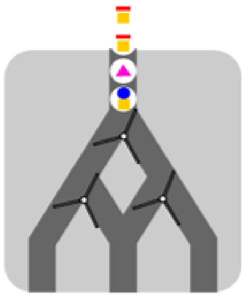
(A)



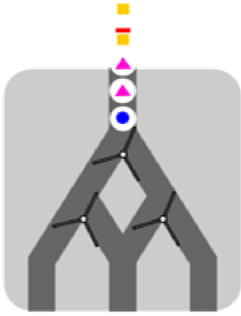
(B)



(C)



(D)



3. Padrões Artísticos (Resolução)

Solução

(D)

Resolução

A resposta correta é a (D).

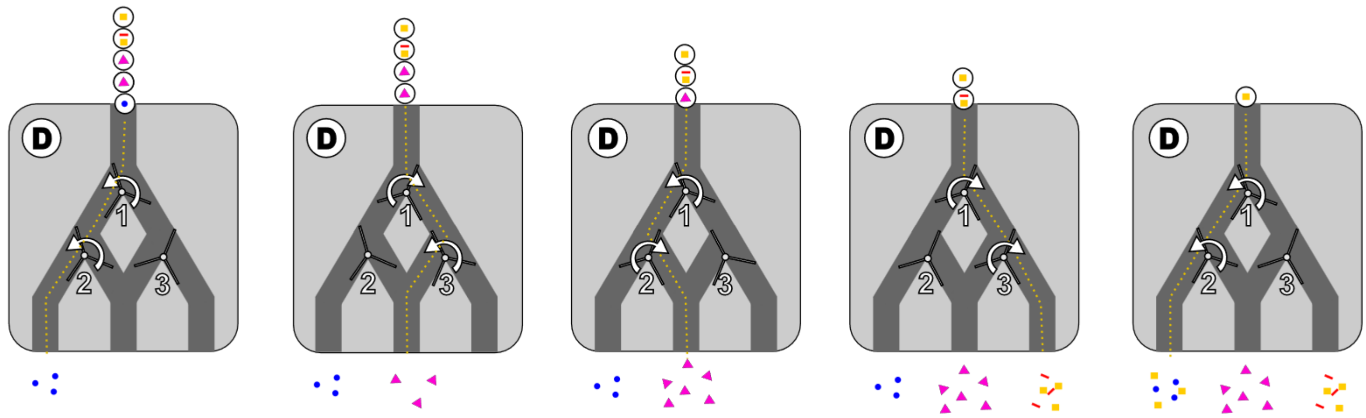
A 1ª bola vai descer pelo caminho mais à esquerda. Isto irá mudar o portão 1 e o portão 2 para a direita.

Como o portão 1 foi alterado, a 2ª bola vai para a direita, mas depois para a esquerda no portão 3. Isto irá mudar o portão 1 novamente para a esquerda e o portão 3 para a direita.

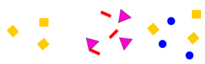
A 3ª bola será enviada para a esquerda no portão 1 e para a direita no portão 2. Isto irá mudar o portão 1 para a direita e o portão 2 para a esquerda.

A 4ª bola será enviada para a direita no portão 1 e continuará à direita no portão 3. Isto irá mudar o portão 1 e o portão 3 para a esquerda.

A 5ª bola será enviada pelo caminho da esquerda.



A resposta (A) está incorreta porque vai criar o seguinte padrão incorreto: 

A resposta (B) está incorreta porque vai criar o seguinte padrão incorreto: 

A resposta (C) está incorreta porque vai criar o seguinte padrão incorreto: 

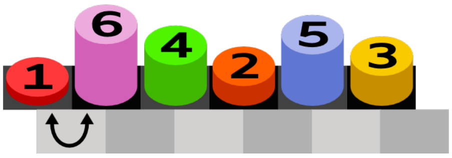
Isto é Pensamento Computacional!

Os computadores utilizam peças muito pequenas que funcionam como estas portas. Chamam-se transístores e um computador moderno tem milhares de milhões deles para controlar o destino da eletricidade. Nesta tarefa, todas as portas foram colocadas inicialmente a enviar a bola para a esquerda esquerda, mas podemos ver como uma porta afecta as portas que se seguem. Os transístores de um computador podem ser ligados ou desligados, tal como estas portas são colocadas à esquerda ou à direita. Imagina se pudesses controlar todas as portas de forma independente, não seria muito mais fácil?

É assim que os transístores de um computador funcionam. Com ou sem carga eléctrica, podem funcionar como um interrutor para controlar um circuito. Os transístores também podem ser utilizados como uma memória para armazenar informação num computador, em que, se houver uma carga no transístor, este representa o dígito 1 e, se não houver carga, representa o dígito 0.

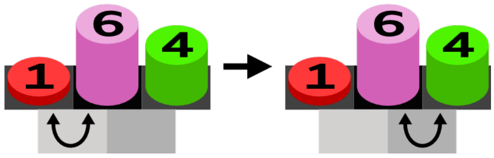
4. Máquina de Trocas

Uma máquina consegue movimentar seis blocos de diferentes alturas. A máquina usa um marcador que fica posicionado entre dois blocos. No início, a configuração da máquina e dos blocos é a seguinte:

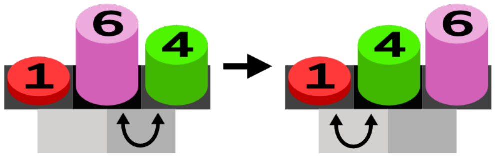


A máquina faz então repetidamente um dos seguintes seguintes passos:

Se o bloco à esquerda do marcador é mais pequeno que o bloco à sua direita, o marcador move-se para a direita.



Se o bloco à esquerda do marcador é mais alto que o bloco à sua direita, os blocos são trocados de posição pela máquina. Além disso, o marcador move-se para a esquerda.



A máquina para quando o marcador atinge a posição mais à direita dos blocos.

Pergunta

Em que posições ficam os blocos quando a máquina para?

Respostas possíveis

(A) 

(C) 

(B) 

(D) 

4. Máquina de Trocas (Resolução)

Solução

(C)

Resolução

A resposta correta é a (C).

Podes obter esta resposta fazendo cuidadosamente cada movimento como descrito e vendo onde acaba, mas uma forma mais rápida é perceber o que esta máquina faz aos blocos.

À medida que movemos o marcador, em cada ponto os blocos à esquerda do marcador são ordenados de baixo para cima. Utilizando o 1º movimento, o marcado desloca-se para a direita, desde que esta ordem se mantenha correta. Se for encontrado um bloco que não esteja na posição correta, uma série de movimentos de troca irá "transportar" esse bloco para a posição correta e depois o marcador pode começar a mover-se novamente para a direita.

C é a resposta em que todos marcadores estão ordenados corretamente.

Isto é Pensamento Computacional!

Em Pensamento Computacional, muitas vezes é necessário reorganizar os dados de modo a que fiquem numa determinada ordem, de pequeno para grande, de barato para caro, etc. Uma razão para isto é que encontrar dados numa sequência ordenada é consideravelmente mais rápido do que encontrar dados numa sequência não ordenada.

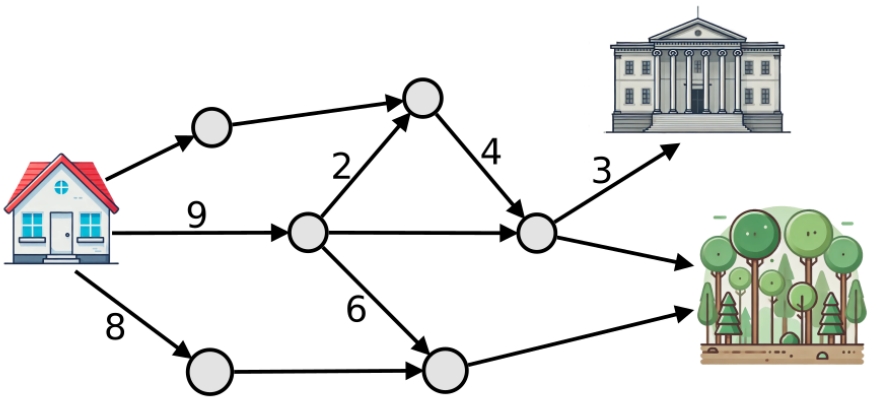
O método aqui descrito chama-se *Gnome Sort*. É fácil de compreender e executar, e também fácil de programar, mas não é utilizado na prática porque é muito mais lento do que os algoritmos mais sofisticados que são tradicionalmente utilizados (por exemplo, *Quick Sort* ou *Merge Sort*).

5. Caminhos para a Floresta

Um grupo de castores saiu da sua casa  para passear.

Alguns dos castores foram para o museu  e o resto deles foi para a floresta .

Os castores documentaram o seu passeio no mapa da figura abaixo. Cada círculo é um sítio onde os castores podiam ir e as setas mostram os *caminhos* e direção que podiam tomar. Os números nos caminhos indicam quantos castores foram por esse caminho. Por exemplo, 3 castores foram pelo caminho que termina no museu.



Infelizmente, alguns castores esqueceram-se de escrever o número, e por isso alguns caminhos ficaram sem indicação de quantos castores o percorreram...

Pergunta

Quantos castores foram para a floresta?

Resposta

Escreve um número inteiro de 0 a 99 na tua folha de respostas.

5. Caminhos para a Floresta (Resolução)

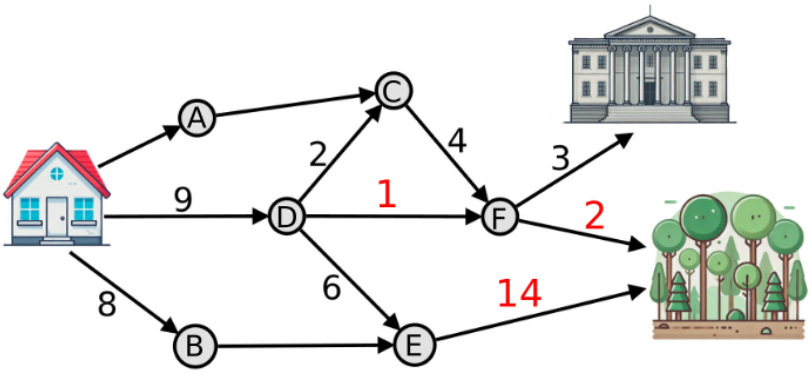
Solução

16

Resolução

16 castores foram para a floresta.

Para descobrir quantos castores percorreram cada caminho, marcámos as interseções com as letras A, B, C, D, E e F. Como um castor só se pode deslocar em cada caminho numa direção, indicada pela seta, cada castor só pode visitar cada interseção uma vez.



A figura seguinte ilustra como podemos deduzir o número de castores que percorreram certas partes do caminho (os números a vermelho):

- D→F (1 castor): 9 castores chegam a D (vindos de casa); destes 9 castores, 2 saem de D para C e 6 saem de D para E; deste modo o único castor restante (9-6-2=1) faz o caminho para F.
- F→floresta (2 castores): 5 castores chegam a F (4 vindos de C e 1 vindo de D); destes 5, 3 vão para o museu pelo que os outros 2 vão para a floresta.
- E→floresta (14 castores): 14 castores chegam a E (6 vindos de D e 8 vindos de B); E só tem saída para a floresta, pelo que os 14 castores seguem essa direção.

Somando os números de castores que foram de F para a floresta e de E para a floresta, temos 2 + 14 = 16 castores que foram para a floresta. A resposta final é **16**.

Isto é Pensamento Computacional!

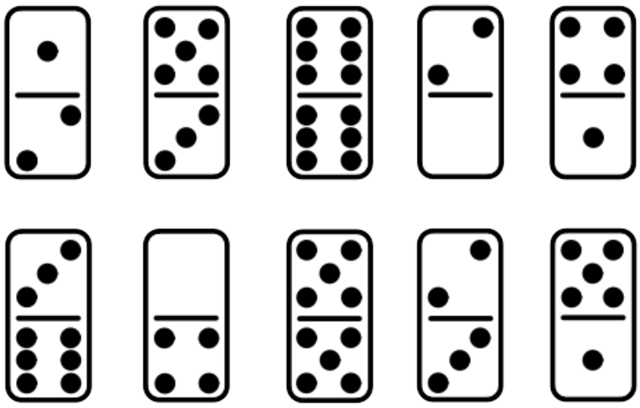
Este desafio envolve um conceito da teoria dos grafos chamado grafo dirigido. As setas indicam a direção permitida no grafo e cada ligação tem um número associado, que representa o seu peso. Este peso pode significar várias coisas, como o comprimento de uma estrada ou a frequência de travessia ao longo desse caminho. Por exemplo, neste problema, os números indicam quantos castores foram por esse caminho.

Este problema utiliza também os conceitos de grau de entrada e grau de saída. O grau de entrada refere-se à soma dos pesos das ligações de entrada num nó, enquanto que o grau de saída se refere à soma dos pesos das ligações de saída de um nó.

Para resolver este problema, os participantes precisam de determinar os caminhos não explicitamente indicados a partir da casa do castor. Isto implica analisar o grau de entrada e o grau de saída dos caminhos para inferir os restantes percursos efectuados pelos castores.

6. Adivinha o Dominó

A Alice e o Bernardo estão a jogar um jogo. Eles têm 10 dominós na mesa:



O Bernardo escolhe um dominó secreto, que apenas ele conhece. A Alice pode então fazer uma série de perguntas para tentar adivinhar qual é o dominó secreto. Cada pergunta tem de ter como resposta **sim** ou **não**.

A Alice quer escolher as questões de tal modo que, independentemente da resposta do Bernardo, ela tenha o menor número de possibilidades restantes para o dominó secreto.

Pergunta

Qual é a primeira pergunta que a Alice deve fazer?

Respostas possíveis

(A) A soma das pintas do dominó secreto é maior ou igual a 7?

(B) O número de pintas do menor lado do dominó secreto é maior ou igual a 2?

(C) O número de pintas do maior lado do dominó secreto é maior ou igual a 4?

(D) Ambos os lados do dominó secreto têm o mesmo número de pintas?

6. Adivinha o Dominó (Resolução)

Solução

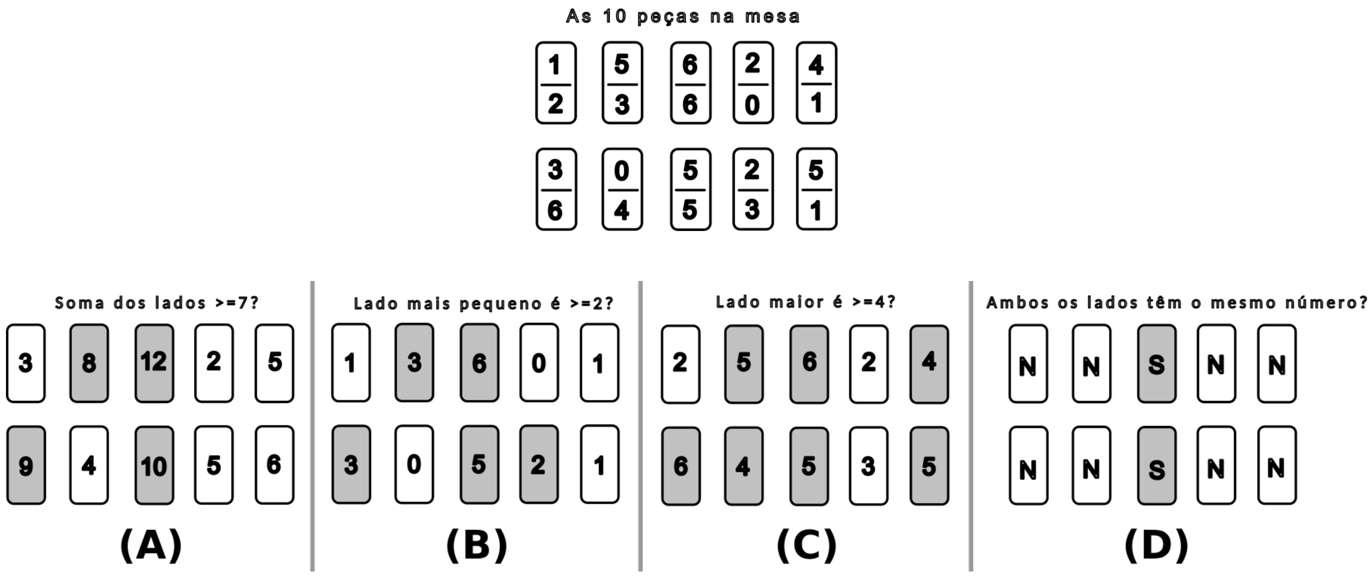
(B)

Resolução

A resposta correta é a (B), porque esta pergunta divide o conjunto em exatamente dois grupos de cinco dominós (cinco dominós têm uma extremidade menor maior ou igual a 2, pelo que os outros cinco têm uma extremidade menor com menos do que duas pintas), pelo que, independentemente da resposta do Bernardo, ela ficará com cinco dominós candidatos ainda válidos. Desta forma, a Alice elimina garantidamente metade das possibilidades, independentemente da resposta de Bernardo.

Nas outras opções, os grupos não estão divididos de forma tão uniforme e uma das respostas possíveis deixaria mais de cinco dominós como candidatos restantes.

A imagem a seguir mostra, para cada pergunta possível, qual conjunto de dominós seria indicado por uma resposta sim (a cinzento) ou não.



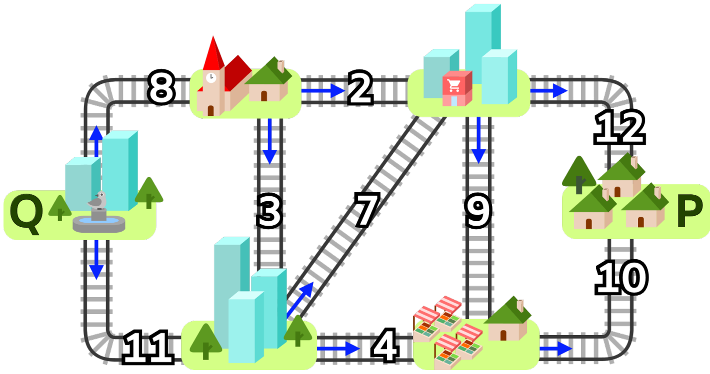
Isto é Pensamento Computacional!

Os computadores armazenam muitos dados e necessitam de formas rápidas de encontrar informações específicas. Em vez de analisarmos os dados um a um, podemos utilizar métodos mais inteligentes se organizarmos bem os dados. Por exemplo, se os dados estiverem ordenados, podemos utilizar um método chamado *pesquisa binária*. Funciona da seguinte forma: começamos por verificar o item do meio. Se o que estamos à procura for maior ou menor, podemos ignorar metade dos dados e concentrarmo-nos apenas no resto. Continuamos a fazer isto, reduzindo a quantidade de dados para metade de cada vez. Assim, se tivermos 1 milhão de itens, podemos encontrar o que procuramos em apenas 20 passos!

Para esta tarefa, só temos perguntas de sim/não, pelo que a melhor primeira pergunta a fazer dividiria as possibilidades em duas, tal como o primeiro passo na pesquisa binária. Se precisássemos de continuar a fazer perguntas para encontrar um dominó secreto, poderíamos fazer um plano de todas as perguntas possíveis, o que é como fazer uma árvore de decisão.

7. Rede de Comboios

Na Bebralândia as cidades estão ligadas por uma rede de comboios. Para cada linha existe um número máximo de comboios que a podem percorrer em cada dia, como indicado na figura abaixo.



A cidade Q quer enviar materiais para a cidade P. Diferentes comboios podem usar a mesma linha, mas têm de seguir sempre na direção apontada pelas setas e respeitar o limite diário de comboios na linha.

Pergunta

Qual é o máximo número de comboios que pode ir da cidade Q para a cidade P em cada dia?

Respostas possíveis

(A) 13 comboios

(B) 15 comboios

(C) 19 comboios

(D) 22 comboios

7. Rede de Comboios (Resolução)

Solução

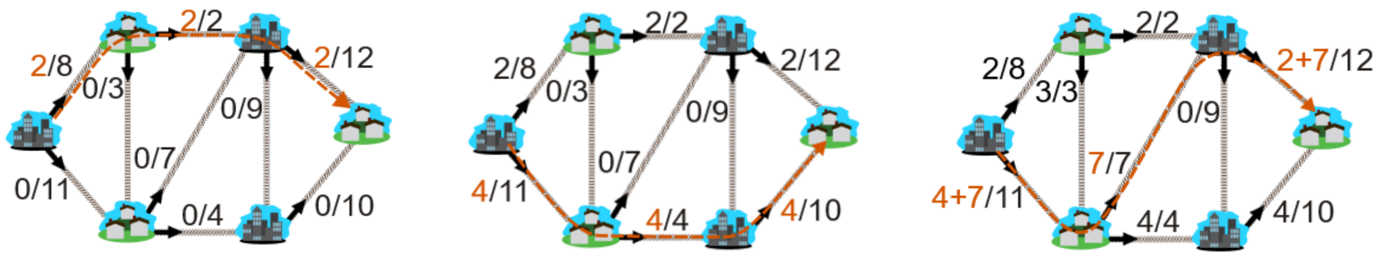
(A)

Resolução

A resposta correta é (A) 13 comboios.

A primeira observação é que a cidade Q não pode enviar para a cidade P mais do que 8 + 11 = 19 comboios, o número de comboios que pode partir da cidade Q. Assim, a resposta (D) 22 comboios, está incorreta.

Uma forma de resolver este desafio é encontrar o caminho mais curto possível de Q a P (neste caso, de comprimento 3) e determinar o maior número de comboios que pode acomodar. A seguir, repetimos este passo no resto da rede com o número remanescente de comboios disponíveis em cada linha. Os diagramas representados abaixo mostram 3 caminhos encontrados desta forma, com a capacidade combinada de 2 + 4 + 7 = 13 comboios. Há outras formas de levar 13 comboios de Q a P, incluindo algumas que requerem caminhos mais longos.



Para verificar que 13 é o máximo, repara nas três linhas no meio da rede com capacidades 2, 7 e 4: todos os caminhos passam por uma destas, por isso não conseguimos fazer melhor do que 2 + 7 + 4 = 13.

Isto é Pensamento Computacional!

O problema aqui apresentado é um exemplo de cálculo do fluxo máximo. Os problemas deste tipo assemelham-se a um jogo de puzzle com água a passar por canos: tem-se uma rede de canos com diferentes capacidades e quer-se descobrir quanta água pode passar de uma origem para um estino.

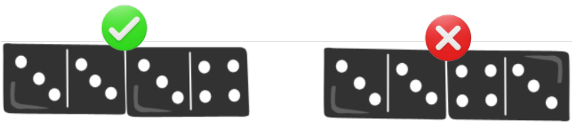
O problema do fluxo máximo pode ser resolvido através de vários algoritmos: Algoritmo de Ford-Fulkerson, Algoritmo de Edmonds-Karp, Algoritmo de Dinic e muitos outros. Os caminhos incontornáveis no meio da rede neste desafio formam o que é conhecido como "corte mínimo".

Cada um dos algoritmos acima tem suas vantagens e desvantagens em termos de eficiência de tempo de execução, facilidade de implementação e aplicabilidade a diferentes tipos de redes. A escolha do algoritmo depende de factores como a dimensão da rede, a natureza das capacidades e o nível de otimização pretendido.

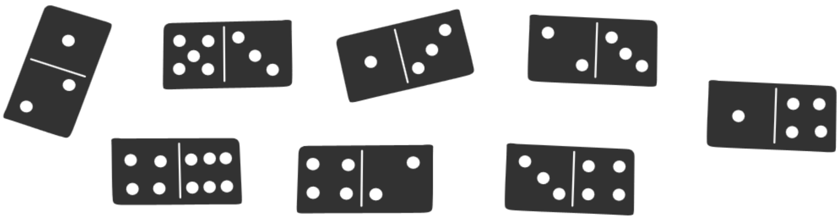
Os algoritmos de fluxo máximo são utilizados em otimização do tráfego, nos sistemas de distribuição (por exemplo, água e eletricidade), na atribuição de recursos, nas redes informáticas, etc.

8. Fila de Dominós

Dois dominós podem ser colocados lado a lado se os quadrados de ligação de cada dominó tiverem o mesmo número de pontos. Por exemplo, em baixo, à esquerda, os dois dominós podem ser colocados adjacentes porque os quadrados de ligação têm ambos 3 pontos. Pelo contrário, à direita, os dois dominós não podem ser colocados adjacentes porque os quadrados de ligação têm 3 e 4 pontos, respetivamente.

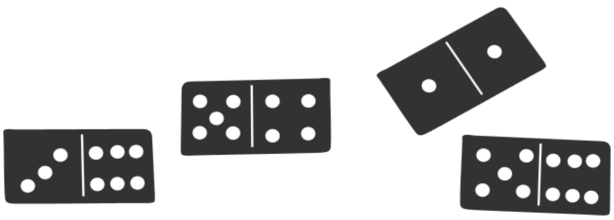


Já sabemos que os seguintes 8 dominós não podem ser colocados lado a lado numa fila de 8 dominós usando a regra acima:



Pergunta

Qual dos seguintes dominós deve ser escolhido, juntamente com os 8 dominós acima, para construir uma fila de 9 dominós?



Respostas possíveis

(A) 3-6

(B) 5-4

(C) 1-1

(D) 5-6

8. Fila de Dominós (Resolução)

Solução

(D)

Resolução

A única opção correta é a (D) 5-6

Cada peça de dominó é constituída por dois quadrados com 1 a 6 pontos brancos.

Aqui temos os seguintes quadrados:

- com 1 ponto - 3 quadrados
- com 2 pontos - 3 quadrados
- com 3 pontos - 4 quadrados
- com 4 pontos - 4 quadrados
- com 5 pontos - 1 quadrado
- com 6 pontos - 1 quadrado

Note-se que, para fazer uma linha contínua de peças de dominó, não pode haver mais do que dois tipos de quadrados com um número ímpar de ocorrências. Os dominós com estes dois tipos de quadrados podem ser colocados no início e no fim da fila.

Aqui temos 4 tipos de quadrados que ocorrem um número ímpar de vezes: **1, 2, 5 ou 6 pontos**.

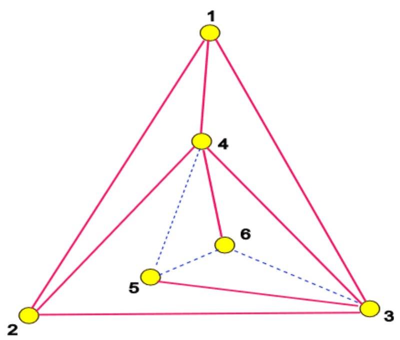
Dos quatro tipos propostos, apenas a adição do dominó **5-6** permite tornar par dois dos tipos que tinham número ocorrências ímpar. Aqui fica um exemplo de uma sequência de 9 dominós:

1 - 2 - 3 - 1 - 4 - 3 - **5 - 6** - 4 - 2

Isto é Pensamento Computacional!

Este desafio relaciona-se com o Pensamento Computacional de duas formas.

1. Se o espaço de pesquisa for algo limitado (neste caso, a disposição de 9 dominós = 362 880), uma pesquisa completa de todas as opções possíveis (força bruta ou pesquisa exaustiva) pode decidir se os dominós adicionados conduzem a uma solução. Enumerar todo o espaço de soluções é, por vezes, a única forma de encontrar a(s) solução(ões) pretendida(s).
2. No caso geral, esta tarefa está relacionada com o problema clássico do caminho Euleriano. Formando um grafo com um número de pontos como vértices (1 a 6). E o número de pontos nos dois quadrados de um dominó indica uma aresta entre os respectivos vértices. Ao encontrar o caminho Euleriano, pode facilmente ver-se que é o dominó 5-6 que precisa de ser adicionado.



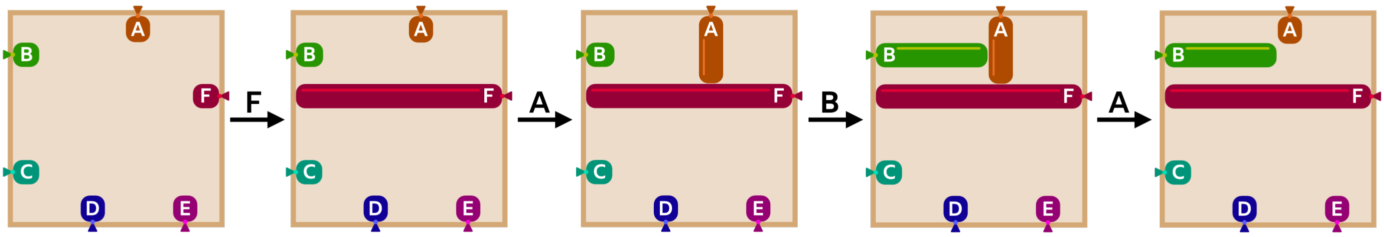
9. Máquina de Balões

Os castores têm uma máquina que pode criar imagens através do enchimento de balões numa moldura quadrada. Os balões estão etiquetados com as letras A, B, C, D, E e F.

A máquina lê as letras uma de cada vez. Quando lê uma letra:

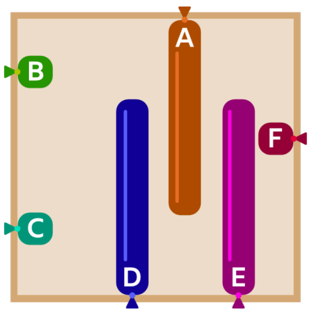
- Se o balão marcado com essa letra estiver vazio, é insuflado até tocar noutro balão ou na extremidade oposta da moldura
- Caso contrário, esvazia o balão marcado com essa letra.

Por exemplo, se todos os balões estiverem vazios no início e a máquina ler a sequência de letras **FABA**, fará o seguinte:



Pergunta

Supõe que no início todos os balões estão vazios.
Indica uma sequência de 9 letras que no final resulte na figura abaixo:



Resposta

Escreve uma sequência de 9 letras seguidas formada unicamente por letras A, B, C, D, E ou F.

9. Máquina de Balões (Resolução)

Solução

Existem 4 respostas corretas possíveis com 9 letras:

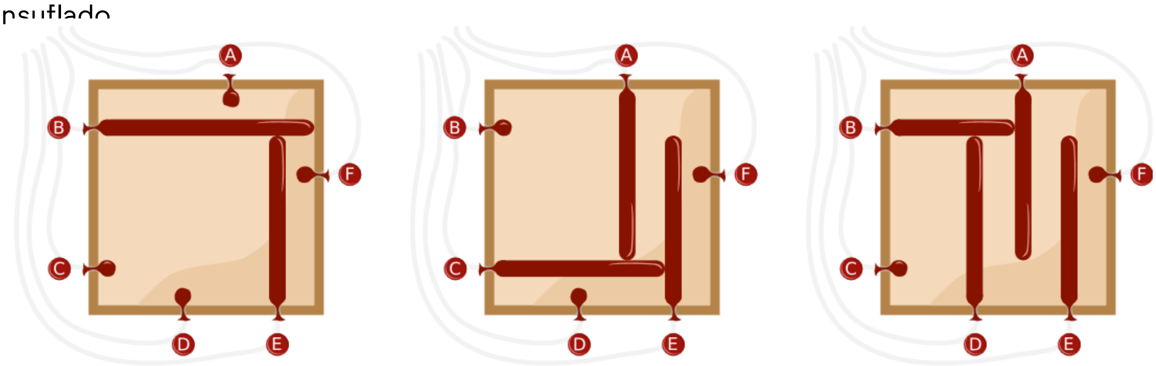
B E B C A C B D B B E C B A C B D B B E B C A B C D B B E C B A B C D B

Resolução

Vejamos uma explicação detalhada para uma das respostas corretas: B E B C A C B D B. As imagens ilustram a execução destas instruções, com os passos intermédios. A 1ª imagem mostra o estado depois da execução de B E. Podes ver que é necessário encher B antes de E para B servir de barreira para o enchimento de E.

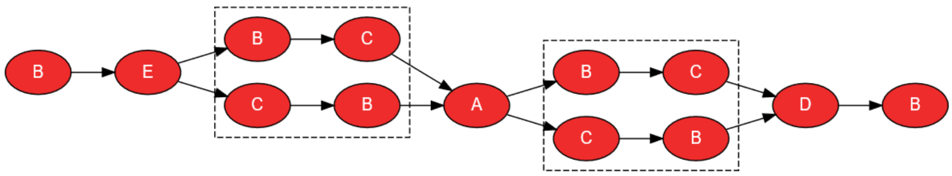
A 2ª imagem mostra o estado depois de B E B C A. O balão C serve de barreira para A e portanto deve ser insuflado antes de A. Além disso, o balão B deve ser esvaziado, antes de A poder ser insuflado, caso contrário, não atingiria o comprimento certo.

A 3ª imagem mostra o estado depois de B E B C A C B D. B serve de barreira para D e C deve ser esvaziado antes de D ser insuflado.



Para encontrar todas as respostas corretas, é útil representar o problema através de um *grafo dirigido*. As letras representam os balões. Uma seta de B para E indica que o balão B deve ser insuflado antes do balão E, etc. Há dois casos em que a ordem de B e C pode variar (indicados com um retângulo a tracejado).

Cada resposta correta (sequência de balões a encher) é uma ordenação topológica deste grafo. Neste caso, temos 4 ordenações topológicas, pelo que existem quatro soluções corretas possíveis.



Observação para avaliar a correção da sua solução: os balões insuflados têm de aparecer vezes ímpares → A, D e E têm de aparecer vezes ímpares. Os balões vazios têm de aparecer vezes pares (incluindo 0)→ B, C e F.

Isto é Pensamento Computacional!

A *sequência* de letras é um *programa de computador*, que controla uma máquina. Cada letra é uma *instrução* que faz a máquina encher ou esvaziar um balão. Como na maioria dos programas de computador deste tipo, a ordem das instruções é essencial. Por exemplo, a sequência B E faz com que a máquina crie uma imagem diferente da sequência E B.

O *grafo dirigido* utilizado na explicação da resposta não tem *ciclos*, pelo que é designado por grafo acíclico direcionado (*DAG*). Qualquer DAG tem pelo menos uma ordem topológica, e são conhecidos algoritmos rápidos para construir uma ordem topológica. Os DAGs têm várias aplicações em informática como, por exemplo, no planeamento de tarefas.

10. Figuras Escondidas

O Leonardo inventou um novo método para codificar imagens usando as operações H (horizontal) e V (vertical). Uma imagem é essencialmente um retângulo dividido em linhas e colunas de células quadradas chamadas píxéis, em que cada pixel armazena uma cor. Em cada aplicação da operação H:

- Todos os píxeis da 1ª linha permanecem no seu lugar (ou seja, não se movem).
- Todos os píxeis da 2ª linha deslocam-se 1 célula para a direita.
- Todos os píxeis da 3ª linha deslocam-se 2 células para a direita.

  ...
- Todos os píxeis da n-ésima linha deslocam-se $n-1$ células para a direita.

Quando os pixéis de qualquer linha são empurrados para além da margem direita da imagem, são mantidos em ordem e movidos como um grupo para o espaço disponível na extremidade esquerda da linha.
Da mesma forma, em cada aplicação da operação V:

- Cada pixel da n-ésima coluna move-se $n-1$ posições para baixo, e os pixéis empurrados para além da margem inferior são movidos para cima.

Aqui fica um exemplo de uma imagem 3×3 com cores marcadas de 1 a 9:

1	2	3
4	5	6
7	8	9

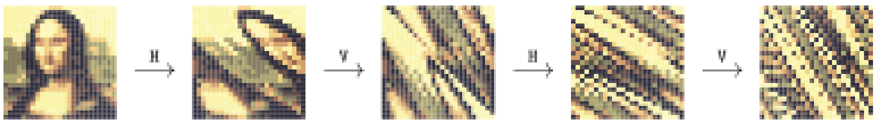
$\xrightarrow{H}$

1	2	3
6	4	5
8	9	7

$\xrightarrow{V}$

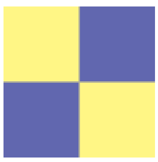
1	9	5
6	2	7
8	4	3

A imagem seguinte mostra como uma sequência $HVHV$ pode ser usada para codificar uma imagem de 25×25 píxeis da Mona Lisa:



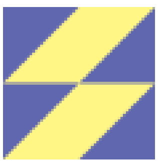
Pergunta

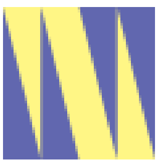
O Leonardo codificou a seguinte imagem de 1000×1000 píxeis aplicando V e depois H:

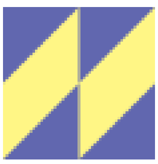


Qual das seguintes imagens corresponde ao resultado da codificação?

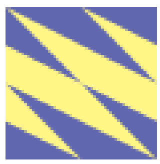
Respostas possíveis

(A) 

(B) 

(C) 

(D) 

(E) 

10. Figuras Escondidas (Resolução)

Solução

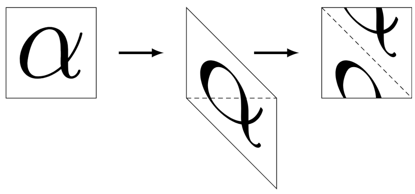
(E)

Resolução

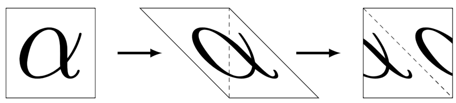
A resposta correta é a (E). A imagem tem uma resolução elevada, pelo que o movimento de pixels individuais pode ser razoavelmente aproximado pelas transformações geométricas das formas representadas na imagem.

No exemplo 3×3, vemos que a primeira coluna (1, 4, 7) se torna a diagonal de cima para baixo da esquerda para a direita, enquanto a diagonal de cima para baixo da direita para a esquerda (3, 5, 7) se torna a última coluna. Isto sugere que está a ocorrer um corte, o que é apoiado pela primeira imagem do exemplo da Mona Lisa. Analisaremos os efeitos geométricos das duas operações em mais pormenor, começando por V, uma vez que será aplicada em primeiro lugar.

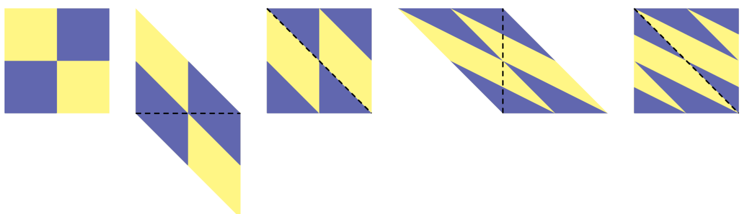
O efeito de V pode ser dividido em duas fases. Primeiro, o retângulo é cortado verticalmente em direção ao canto inferior direito para formar um paralelogramo com uma imagem distorcida. Depois, translada verticalmente o triângulo saliente na parte inferior para formar uma nova imagem retangular com o tamanho original:

V:


De forma semelhante, o efeito de H é um corte horizontal seguido de uma translação horizontal parcial:

H:


Aplicando estes passos à imagem dada, temos a seguinte sequência:



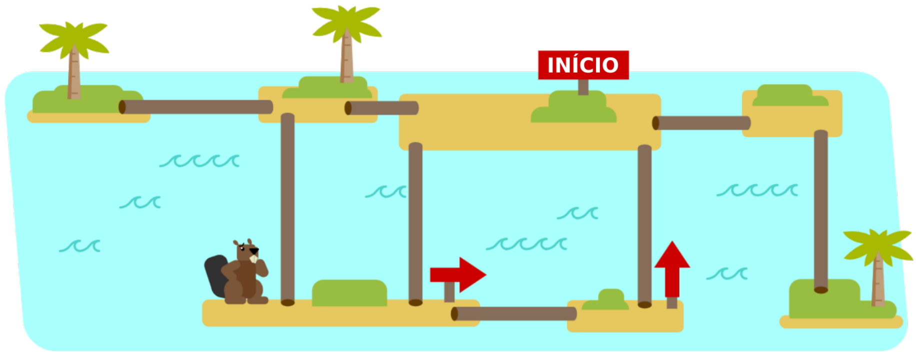
Isto é Pensamento Computacional!

Ao transferir informação, pode ser importante encriptar os dados para que quem intercetar a sua informação não consiga determinar facilmente os dados originais. Nesta tarefa, foi fornecido um algoritmo de encriptação que parece simples, é facilmente reversível, mas que gera uma nova imagem que não revela imediatamente a original. De facto, sem o conhecimento do mecanismo de encriptação, podem existir muitas potenciais imagens de origem. No entanto, é importante notar que não é necessária uma chave secreta; basta conhecer o algoritmo de encriptação e é possível desencriptar qualquer imagem encriptada desta forma.

Durante este processo, convertemos a imagem numa matriz, onde os valores das células representam um nível de cor. Isto permite-nos representar os dados de uma forma que pode ser facilmente operada.

11. Explorando as Ilhas

O castor Hugo está a visitar um grupo de ilhas ligadas por troncos de árvores. Ele quer visitar cada ilha pelo menos uma vez. Quando ele está numa ilha, pode ver todas as ilhas vizinhas e ver se têm um sinal nelas. Na imagem abaixo, ele já visitou três ilhas.



O Hugo desloca-se de acordo com as seguintes instruções:

Vai para a primeira ilha e coloca um sinal de INÍCIO.

De cada vez que estiveres numa nova ilha, faz o seguinte:

- **Se** não existe nenhum sinal na ilha, coloca uma seta que aponta para a ilha de onde vieste.
- **Se** esta ilha tem uma ilha vizinha que **(1)**, **então** vai para essa ilha.
  - **Senão:**
    - **Se** existe uma seta na ilha, **então** vai para a ilha **(2)**.
    - **Senão**, fica onde estás. Já visitaste todas as ilhas pelo menos uma vez.

(1):
- não tem nenhum sinal
- tem uma sinal de seta
- tem um sinal de INÍCIO

(2):
- de onde acabaste de vir
- para onde a seta aponta

Pergunta

Como devem ser preenchidos os espaços (1) e (2) para que as instruções fiquem corretas e o Hugo visite todas as ilhas pelo menos uma vez?

Respostas possíveis

(A) (1) não tem nenhum sinal + (2) de onde acabaste de vir

(B) (1) não tem nenhum sinal + (2) para onde a seta aponta

(C) (1) tem um sinal de seta + (2) de onde acabaste de vir

(D) (1) tem um sinal de seta + (2) para onde a seta aponta

(E) (1) tem um sinal de INÍCIO + (2) de onde acabaste de vir

(F) (1) tem um sinal de INÍCIO + (2) para onde a seta aponta

11. Explorando as Ilhas (Resolução)

Solução

(B)

Resolução

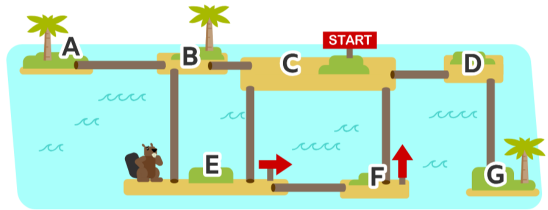
Espaço (1): *"não tem nenhum sinal"*

Repara que as ilhas onde o Hugo já esteve têm um sinal, enquanto que as ilhas onde ele ainda não esteve não têm sinal (porque ele coloca um sinal quando vai para uma ilha que ainda não tem sinal).

Correto: não tem nenhum sinal - vai para as ilhas onde ainda não esteve.

Incorreto: tem um sinal de seta - deslocar-se para sempre entre ilhas onde já esteve.

Por exemplo, na situação apresentada ao lado, ele deve ir para a ilha vizinha sem sinal, B. Se ele escolhesse preferencialmente ilhas que já têm sinal, ele deslocar-se-ia entre as ilhas C, E e F, mas nunca iria para as ilhas onde ainda não esteve.



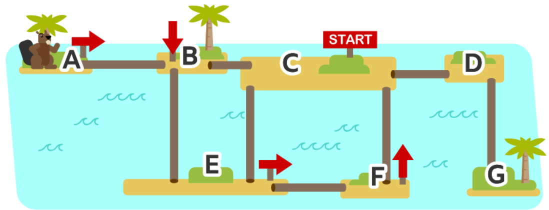
Incorreto: tem um sinal de INÍCIO - durante a segunda iteração do algoritmo, o Hugo estará novamente na ilha inicial e o algoritmo executaria incorretamente a última instrução "Senão".

Espaço (2): *"para onde a seta aponta"*

Correto: para onde a seta aponta - voltar para uma ilha que ainda tem vizinhos inexplorados

Incorreto: de onde acabaste de vir - deslocar-se para sempre entre duas ilhas que não têm vizinhos sem sinais

Por exemplo, na situação do lado, ele está na ilha A que já não tem vizinhos inexplorados. Ele não tem escolha a não ser ir para a ilha B. Na etapa seguinte, se ele voltasse para o lugar de onde acabou de sair, ele voltaria para A, depois B, depois A, e nunca sairia desse par de ilhas. Assim, ele deve seguir as setas nos sinais para ir para E, F, C, e daí para a ilha D, onde ainda não esteve.



Isto é Pensamento Computacional!

A rede de ilhas ligadas é designada por grafo. Um grafo é uma estrutura de dados constituída por vértices (por exemplo, ilhas) e arestas (ligações) entre eles (por exemplo, troncos). Pode, por exemplo, ser utilizado para modelar estações ligadas por linhas de comboio, pessoas ligadas por amizades ou possíveis estados de um jogo de xadrez ligados por movimentos válidos.

Muitas vezes, é importante procurar em todos os vértices de um grafo, por exemplo, para encontrar um que tenha determinadas caraterísticas. Especialmente se o grafo for grande, com muitos vértices e arestas, é importante ter uma estratégia para garantir que não se esquece de nenhum vértice. A estratégia utilizada nesta tarefa chama-se "pesquisa em profundidade". Pode, por exemplo, utilizá-la para resolver um labirinto (ou seja, procurar em todas as posições de um labirinto para encontrar a que está na saída).

O algoritmo de pesquisa de profundidade primeiro mostrado nesta tarefa foi descrito utilizando pseudocódigo, que é uma descrição dos passos de um algoritmo utilizando uma mistura de linguagem natural e convenções de linguagem de programação. É uma forma comum de definir um algoritmo sem utilizar uma notação de linguagem de programação real, mas uma notação informal e auto-explicativa destinada à leitura humana.

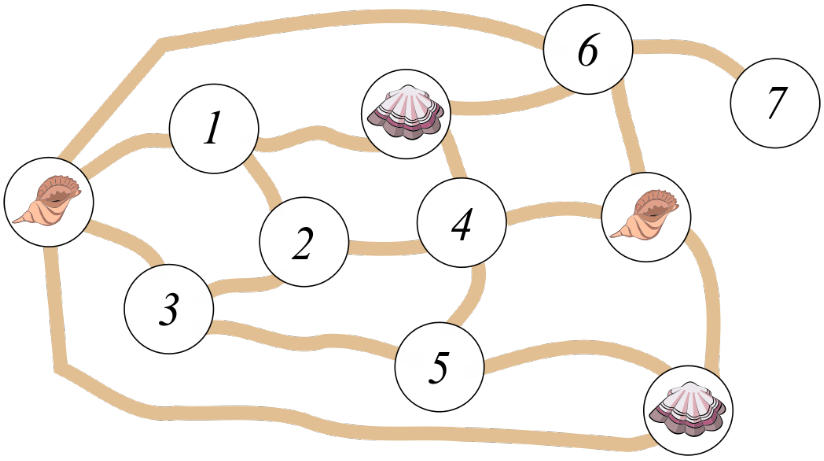
12. Um Jogo com Conchas

A Liliana e o Pedro estão a jogar um jogo na praia que envolve conchas, buracos e linhas na areia.

Durante o jogo, colocam à vez conchas nos buracos vazios, assegurando-se de que são adicionadas novas conchas que não tenham sido colocadas nos buracos anteriormente.

A Liliana joga com um tipo de conchas:  e o Pedro joga com outro tipo de conchas: .

O jogo já começou e cada jogador completou duas jogadas, colocando duas das suas conchas, como indicado na figura abaixo:



O perdedor do jogo será a primeira pessoa que tiver as suas conchas colocadas em dois buracos ligados por uma linha na areia.

É a vez da Liliana. Os números representam os buracos vazios.

Pergunta

Em que buraco vazio é que a Liliana deve colocar a sua próxima concha se quiser garantir uma vitória no jogo?

Respostas possíveis

(A) 1 **(B)** 2 **(C)** 3 **(D)** 4 **(E)** 5 **(F)** 6 **(G)** 7

12. Um Jogo de Conchas (Resolução)

Solução

(G)

Resolução

A resposta correta é (G), buraco 7.

A Liliana perderá imediatamente se colocar uma concha nos buracos 1, 3, 4 ou 6. Isto significa que ela tem de escolher o buraco 2, 5 ou 7 se quiser ter uma hipótese de ganhar.

Se a Liliana colocar a sua próxima concha no buraco 2, então o Pedro pode colocar uma concha no buraco 7. Nessa altura, a Liliana deve colocar uma concha no buraco 5 para evitar perder imediatamente. O Pedro pode colocar uma concha no buraco 3, e a Liliana perderá imediatamente na sua próxima jogada.

Da mesma forma, se a Liliana colocar a sua próxima concha no buraco 5, então o Pedro pode colocar uma concha no buraco 7. Nessa altura, a Liliana deve colocar uma concha no buraco 2 para evitar perder imediatamente. O Pedro pode colocar uma concha no buraco 3 e a Liliana perderá imediatamente na sua próxima jogada.

Em vez disso, supõe que a Liliana coloca uma concha no buraco 7. Então o Pedro deve colocar uma concha no buraco 1, 2, 3, 4, 5 ou 6. Ele perderá imediatamente se colocar uma concha no 1, 4, 5 ou 6. Se ele colocar uma concha no 2 ou no 3, então a Liliana pode colocar uma concha no buraco 5. Nessa altura, o Pedro perderá imediatamente quando colocar a sua próxima concha. Portanto, a Liliana tem a garantia de ganhar o jogo colocando uma concha no buraco 7.

Isto é Pensamento Computacional!

Jogos como este são estudados por *teóricos dos jogos*. Em particular, a descoberta e implementação de estratégias vencedoras, como a que descobrimos para a Liliana, é um elemento da *teoria dos jogos algorítmicos*.

A teoria dos jogos tem aplicações na economia, biologia, ciências sociais e, certamente, no Pensamento Computacional. Em particular, os atuais cientistas informáticos estão frequentemente a recorrer à teoria dos jogos para ajudar a resolver problemas importantes. Estes problemas podem surgir sempre que existe algum tipo de "concorrência". Em economia, pode tratar-se de empresas concorrentes. Na biologia, diferentes espécies de plantas competem pela luz solar e diferentes espécies de animais competem pela comida. Nas ciências sociais, as votações e as eleições são um exemplo natural de concorrência.

Mais tecnicamente, no domínio da *inteligência artificial*, os sistemas são frequentemente constituídos por *agentes* que se comportam como os jogadores de um jogo. Os algoritmos clássicos, como o *minimax* e o *negamax*, podem ser utilizados para encontrar estratégias vencedoras e os algoritmos mais modernos de *aprendizagem automática* podem resolver problemas mais complexos.

Outro exemplo de teoria dos jogos surge na *segurança de redes*. Pode imaginar-se a interação entre um administrador de rede e um *hacker* como uma competição ou um jogo.