CONCURSO BEBRAS · ESPAÑA

# Bebras 2024

## 1.º y 2.º de ESO · Grupo IV (12–14 años)

Cuadernillo con las 15 tareas del concurso, sus soluciones explicadas y las secciones «Esto es informática» y «Esto es pensamiento computacional».

bebras.ehu.es

## Presentación

Bebras es un desafío internacional de informática y pensamiento computacional que se celebra cada curso en más de 80 países. Sus tareas son problemas breves, resolubles sin conocimientos previos de programación, que acercan al alumnado a ideas fundamentales de la informática: algoritmos, estructuras de datos, lógica, grafos o representación de la información.

Este cuadernillo recoge las 15 tareas que resolvió el alumnado de 1.º y 2.º de ESO en la edición de 2024, realizada en la plataforma ViLLE (curso «2024 Spanish IV»). Cada tarea valía 10 puntos, con un máximo de 150. Para cada tarea se incluye:

- **Enunciado**, en una página que se puede fotocopiar para trabajar en clase.
- **Solución** con la respuesta correcta y su explicación.
- **Esto es informática**: el concepto informático que hay detrás de la tarea.
- **Esto es pensamiento computacional**: las destrezas que se ponen en juego al resolverla.

La dificultad indicada es la que asignó el taller internacional Bebras para la franja de 12 a 14 años; cuando una tarea no tenía asignada esa franja, se indica entre paréntesis la dificultad en la franja más próxima. Las opciones de respuesta aparecen en el orden de los materiales originales; en ViLLE el orden pudo ser distinto.

## Resumen de tareas y respuestas

| N.º | Tarea | Código | Dificultad | Respuesta |
|---|---|---|---|---|
| 1 | La flor milagrosa | 2024-DE-03 | — (media en 10–12) | A) 5 días. |
| 2 | Cartas dibujadas | 2024-IN-04 | — (fácil en 10–12) | C) Si el hombre se come una pizza, entonces no lleva corona. |
| 3 | Préstamos de canicas | 2024-AT-03 | Media | C) Tim. |
| 4 | Rutas de árboles | 2024-DE-08 | Media |  (roble, abedul, abeto, frutal, árbol claro). |
| 5 | Alfred en el baile | 2024-HU-02 | — (fácil en 10–12) | B) Alfred puede que llevara corbata o puede que llevara bastón. |
| 6 | Superbebras | 2024-DE-07 | Media | C. |
| 7 | En busca del tesoro | 2024-HU-03 | Media | 4. |
| 8 | La máquina de globos | 2024-DE-06a | — (difícil en 14–16) | Por ejemplo, B E B C A C B D B (hay 4 respuestas válidas). |
| 9 | Un camino para el robot | 2024-PL-04 | Fácil | ↑ en la 2.ª columna (3.ª fila), → en la 2.ª columna (2.ª fila) y → en la 4.ª columna (1.ª fila). |
| 10 | Días soleados | 2024-DE-04a | Fácil | B) ningún castor nadó en el estanque de la cascada helada. |
| 11 | Clase en línea | 2024-MY-03 | — (fácil en 10–12) | E) Hannah. |
| 12 | Cadenas de palabras | 2024-CA-02 | Difícil | C) BAT → CAT → COT. |
| 13 | Las cartas Ricca | 2024-CH-03a | Fácil | nombre: ABC... · ojos: 123... · patas: 123... · alas: no/sí · dientes: no/sí. |
| 14 | Plazas libres en el puerto | 2024-MT-02 | — (fácil en 10–12) | C) 6. |
| 15 | La pulsera Bebras | 2024-TW-03 | Media | D) 6 cuentas. |

# 1 La flor milagrosa

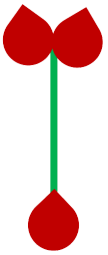
2024-DE-03 · Alemania
Dificultad: — (media en 10–12)

Al amanecer, de cada capullo nuevo de una flor milagrosa brota un tallo. El tallo crece durante todo el día. Al atardecer, el tallo se divide en dos capullos nuevos de flor milagrosa y deja de crecer. Esto se repite día tras día, y la flor milagrosa es cada vez más espléndida.

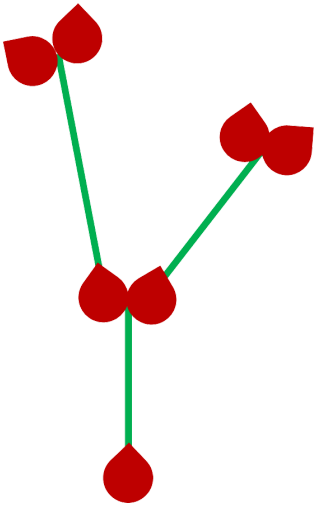
**Capullo nuevo antes del amanecer**



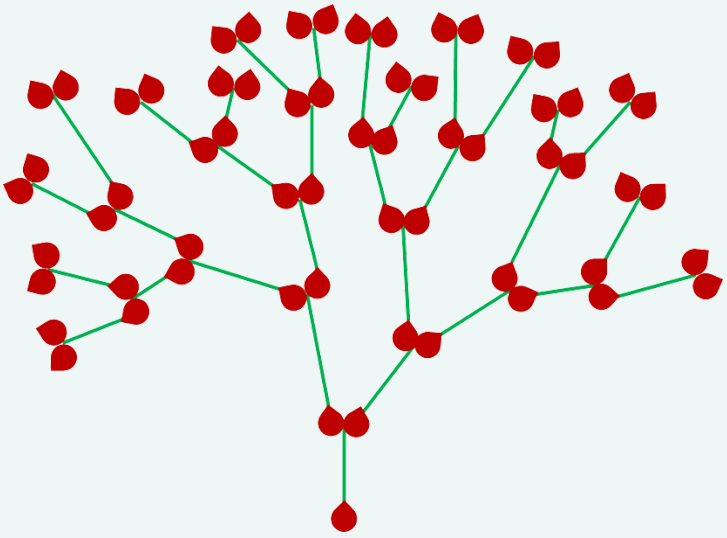
**La flor después del atardecer (primer día)**



**La flor después del atardecer (segundo día)**



**¿Cuántos días lleva creciendo esta flor milagrosa?**



**A)** 5 días
**B)** 11 días
**C)** 16 días
**D)** 32 días

**Respuesta correcta:** A) 5 días.

## Explicación

La respuesta correcta es **5 días**.

Al final de cada día, de cada capullo de la flor ha salido un tallo con dos capullos nuevos. Por tanto, basta con seguir una sola rama (por ejemplo, la de más a la derecha) desde la base hasta un capullo exterior y contar los tallos: hay 5, uno por día. Es la **altura** del árbol binario que forma la flor.

- **11 días** sería el número total de capullos y tallos de una sola rama.
- **16 días** sería el número de tallos exteriores.
- **32 días** sería el número de capullos exteriores ($2^5 = 32$).

## Esto es informática

Las flores milagrosas crecen según una única regla: un capullo forma un tallo y dos capullos nuevos. La regla se aplica a sí misma, y así la flor podría, en teoría, crecer indefinidamente. Este tipo de reglas se llaman **recursivas**.

En programación, una **función recursiva** es una función que se llama a sí misma; la nueva copia vuelve a llamarse, y así sucesivamente. Es imprescindible una **condición de parada**: la función solo se llama a sí misma mientras se cumpla una condición. Un ejemplo matemático es el factorial: $n! = n \cdot (n-1)!$ si $n > 1$, y $1! = 1$. Por ejemplo, $4! = 4 \cdot 3! = 4 \cdot 3 \cdot 2! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$.

Los programas recursivos suelen ser claros y fáciles de leer, aunque, por la cantidad de llamadas, pueden ser más lentos y gastar más memoria que los iterativos. Con recursión se generan también gráficos de aspecto muy natural llamados **fractales**, como los árboles de Pitágoras. Y la flor tiene forma de **árbol binario**, una estructura de datos muy usada en informática: con n niveles tiene $2^n$ hojas.

## Esto es pensamiento computacional

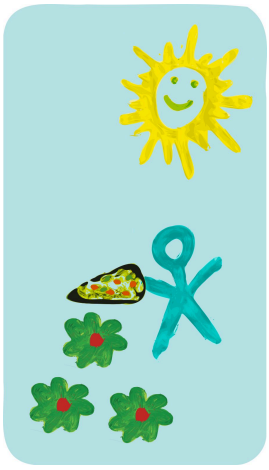
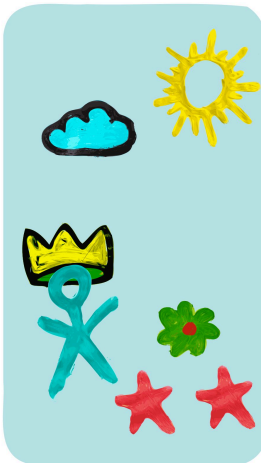
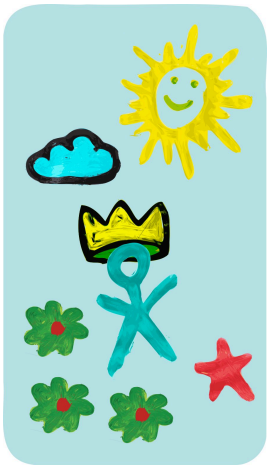
La **abstracción** exige centrarse solo en la información importante del problema e ignorar los detalles irrelevantes. Aquí basta con darse cuenta de que es suficiente mirar la longitud de una sola rama (cualquiera), sin necesidad de contar todos los capullos de la flor.

**Palabras clave y enlaces:** Recursión, árbol binario, fractales · es.wikipedia.org/wiki/Recursión_(ciencias_de_computación) · es.wikipedia.org/wiki/Árbol_binario

# 2 Cartas dibujadas

2024-IN-04 · India
Dificultad: — (fácil en 10–12)

Manas hace dibujos en sus cartas siguiendo una única regla secreta. Ha creado cuatro cartas que cumplen la regla:


Anil observa las cartas y crea una carta nueva, pero Manas le dice que no cumple la regla.



**¿Cuál de las siguientes podría ser la regla secreta que sigue Manas para dibujar las cartas?**

**A)** Si está nublado, entonces no hay flores.
**B)** Tiene que haber una estrella roja o el hombre se come una pizza.
**C)** Si el hombre se come una pizza, entonces no lleva corona.
**D)** Tiene que haber un sol sonriente o una nube.

**Respuesta correcta:** C) Si el hombre se come una pizza, entonces no lleva corona.

## Explicación

La respuesta correcta es la **C**.

La regla secreta tiene que cumplirse en las cuatro cartas de Manas y no cumplirse en la carta de Anil.

- **A** no puede ser: en la 2.ª y la 3.ª carta hay una nube y también flores.
- **D** no puede ser: la 4.ª carta no tiene ni sol sonriente ni nube.
- **B** no puede ser: las cuatro cartas la cumplen, pero la carta de Anil también (en ella el hombre se come una pizza), y Manas dice que esa carta no cumple la regla.
- **C** sí puede ser: ninguna de las cuatro cartas tiene a la vez pizza y corona, y la carta de Anil tiene las dos cosas, así que no cumple la regla.

## Esto es informática

La tarea muestra la **lógica** básica que usa la informática para representar y procesar decisiones y observaciones. Los ordenadores siguen reglas claras para decidir qué hacer, de forma parecida a como observamos patrones y tomamos decisiones en la vida diaria.

Las reglas de las opciones son **condiciones** como las de los programas: «si... entonces...» (implicación) y «... o ...» (disyunción). Un detalle importante: una regla «si A, entonces B» solo se incumple cuando A es verdadero y B es falso. Por eso, las cartas en las que el hombre no come pizza cumplen siempre la regla C, lleve o no corona. En programación, este tipo de reglas sirve, por ejemplo, para validar datos de entrada: un formulario puede rechazar un dato que no cumpla una condición.

## Esto es pensamiento computacional

**Descomposición**: cada opción aísla una condición concreta y su consecuencia, y se puede comprobar carta a carta. **Abstracción**: hay que fijarse solo en los elementos que menciona cada regla (nube, flores, pizza, corona...) e ignorar el resto del dibujo. **Evaluación**: para descartar una regla basta con encontrar un contraejemplo, es decir, una carta de Manas que la incumpla o la carta de Anil si la cumple.

**Palabras clave y enlaces:** Lógica condicional, lógica booleana, implicación · es.wikipedia.org/wiki/Condicional_material

# 3 Préstamos de canicas

2024-AT-03 · Austria
Dificultad: Media

Tres castores amigos juegan a menudo con canicas y se las prestan unos a otros. Para saber quién debe cuántas canicas a quién, decidieron apuntar todos los préstamos en una única hoja de papel:

Ejemplo de lista compartida (no es relevante para la pregunta):

| Presta | Recibe | Cantidad |
|---|---|---|
| Anna | Tim | 5 |
| Tim | Bernd | 3 |
| ... | ... | ... |

Con el tiempo empezaron a desconfiar. Se acusaban unos a otros de hacer trampa y pensaban que algunos préstamos nunca habían ocurrido.

Así que decidieron no depender de una sola lista a la que todos tuvieran acceso: cada castor llevaría su propia lista para detectar intentos de trampa. Al cabo de una semana comparan sus listas. ¿Ha vuelto a hacer trampa alguien?

**Lista de Anna**

| Presta | Recibe | Cant. |
|---|---|---|
| Tim | Anna | 2 |
| Bernd | Tim | 1 |
| Bernd | Anna | 3 |
| Anna | Tim | 2 |

**Lista de Tim**

| Presta | Recibe | Cant. |
|---|---|---|
| Tim | Anna | 2 |
| Bernd | Tim | 1 |
| Tim | Anna | 3 |
| Anna | Tim | 2 |

**Lista de Bernd**

| Presta | Recibe | Cant. |
|---|---|---|
| Tim | Anna | 2 |
| Bernd | Tim | 1 |
| Anna | Tim | 2 |
| Bernd | Anna | 3 |

**¿Quién ha hecho trampa?**

**A)** Anna
**B)** Bernd
**C)** Tim
**D)** Nadie

**Respuesta correcta:** C) Tim.

## Explicación

La respuesta correcta es **Tim**.

Si solo nos fijamos en la columna de cantidades, (2, 1, 3, 2), parece que el tramposo es Bernd, porque es el único cuya lista tiene otra secuencia (2, 1, 2, 3). Pero no es así: la lista de Bernd solo tiene los préstamos en otro orden, y el resultado total es el mismo.

Quien hizo trampa fue Tim: cambió el préstamo (Bernd → Anna, 3) por (Tim → Anna, 3). Si nadie lo hubiera notado, habría ganado 3 canicas a costa de Bernd. Su lista es la única que no coincide con las otras dos en el conjunto de préstamos.

## Esto es informática

La idea de no guardar todas las transacciones en un único servidor central de una entidad de confianza, sino distribuirlas por una red entre iguales (*peer-to-peer*), es una de las ideas básicas de la tecnología **blockchain** (cadena de bloques) o de los **registros distribuidos**.

El método de la tarea solo funciona si la mayoría de los participantes dice la verdad. Blockchain va un paso más allá con un **algoritmo de consenso** que, por ejemplo, exigiría que quien quiera hacer trampa controle más capacidad de cálculo que todos los demás participantes juntos, algo imposible si la red es lo bastante grande. Aun así, la tarea ya muestra cómo se puede reforzar la confianza en un entorno sin confianza repartiendo el poder entre todos, en lugar de depender de una única fuente de verdad.

## Esto es pensamiento computacional

Hay que **abstraer** lo esencial de cada lista: lo que importa es el conjunto de préstamos y su efecto, no el orden en que aparecen. Después se **evalúan** las tres listas comparándolas entre sí para localizar la que discrepa de la mayoría, evitando la conclusión precipitada a la que lleva fijarse solo en un patrón superficial (la columna de cantidades).

**Palabras clave y enlaces:** Blockchain, registro distribuido, consenso · es.wikipedia.org/wiki/Cadena_de_bloques

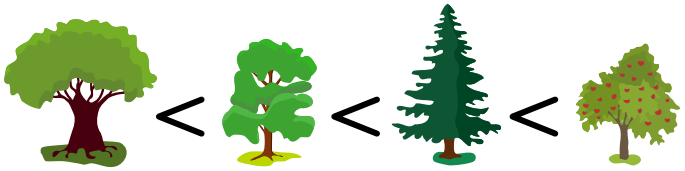
# 4 Rutas de árboles

2024-DE-08 · Alemania
Dificultad: Media

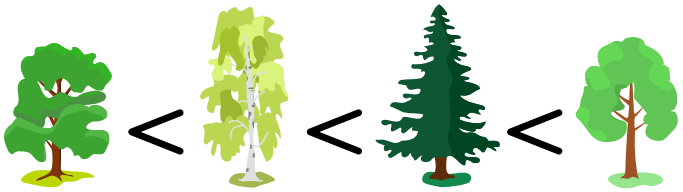
La bióloga Bea organiza rutas por el Bosque de los Castores. En cada ruta enseña algunos árboles especiales.

De tres rutas anteriores recuerda qué árboles gustaron más que otros a los visitantes (árbol 1 < árbol 2 significa que, en esa ruta, el árbol 2 gustó más que el árbol 1):

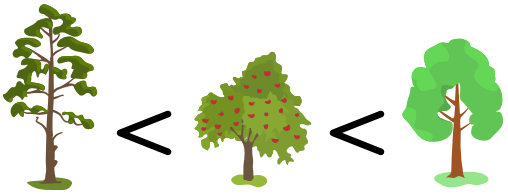
**Ruta 1**



**Ruta 2**



**Ruta 3**



En su próxima ruta, Bea quiere enseñar más tarde los árboles que más gustan. Quiere encontrar un **buen orden**, que no contradiga el orden de popularidad de ninguna de las rutas anteriores.

Por ejemplo, si Bea quiere enseñar los árboles  y  en un buen orden, debe enseñar  antes que . Enseñar  antes que  contradiría la ruta 1, en la que  gustó más que .

En la nueva ruta, Bea quiere enseñar estos árboles: 

**Ayuda a Bea: coloca estos árboles en un buen orden.**



Escribe el orden (1.º a 5.º):

**Respuesta correcta:** (roble, abedul, abeto, frutal, árbol claro).

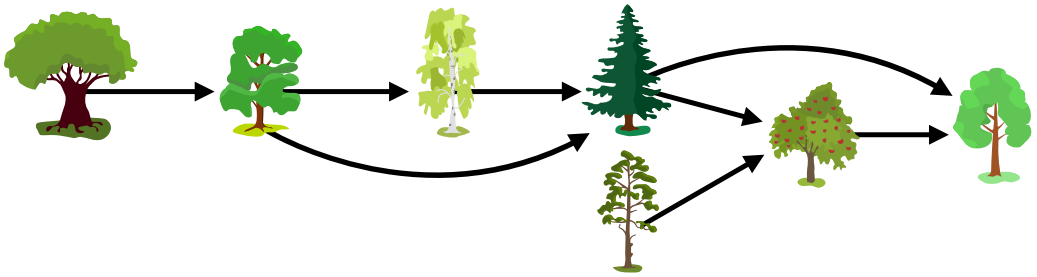
## Explicación

La respuesta es:



Un buen orden no debe contradecir ninguno de los órdenes de popularidad de las tres rutas anteriores. Para evitarlo, Bea tiene que deducir el orden de la nueva ruta a partir de esos órdenes.

Entre los cinco árboles, es el único que no aparece como más popular que ningún otro, así que debe ser el primero. En la ruta 1, < … < , así que debe ir después de . En la ruta 2, < < , así que va antes que y después. Y en la ruta 3, < . Todo ello se resume en este diagrama:



Por tanto, el orden es el único que respeta el diagrama.

## Esto es informática

En esta tarea hay que ordenar un conjunto de árboles por popularidad creciente. Se sabe cómo se comparan algunas parejas de árboles, pero no todas: por ejemplo, el roble y el árbol claro no se comparan directamente en ninguna ruta. ¿Seguro que aun así se pueden ordenar?

Hay dos propiedades importantes:

1. La comparación de popularidad es **transitiva**: si árbol 1 < árbol 2 y árbol 2 < árbol 3, entonces árbol 1 < árbol 3.
2. No hay **conflictos** entre las rutas: si árbol 1 < árbol 2 en una ruta, no hay ninguna otra en la que árbol 2 < árbol 1.

Gracias a ellas, la comparación define un **orden**. A primera vista parece un orden **parcial**, porque no todas las parejas se comparan directamente. Convertir un conjunto parcialmente ordenado en una secuencia totalmente ordenada es la **ordenación topológica**, y en informática se conoce un algoritmo sencillo para hacerlo (que solo funciona si no hay conflictos). De hecho, si se mira con atención, el orden de los siete árboles es **total**: por transitividad se sabe cómo se comparan dos árboles cualesquiera, así que la ordenación de cualquier subconjunto tiene una única solución.

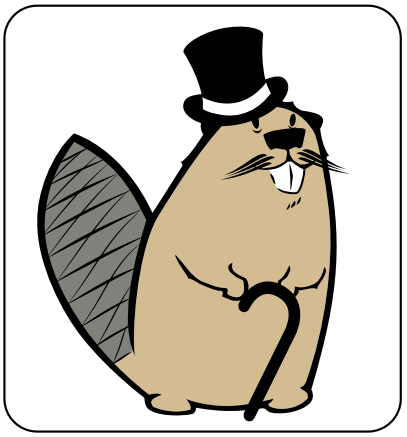
## Esto es pensamiento computacional

Hay que **abstraer** la información de las tres rutas en una sola estructura (un diagrama de relaciones «va antes que»), usar el **razonamiento lógico** (la transitividad) para deducir comparaciones que no aparecen directamente y aplicar un **algoritmo**: elegir en cada paso el árbol que no tiene ningún otro pendiente por delante.

**Palabras clave y enlaces:** Ordenación topológica, orden parcial, transitividad · es.wikipedia.org/wiki/Ordenamiento_topológico

# 5 Alfred en el baile

2024-HU-02 · Hungría
Dificultad: — (fácil en 10–12)



Cuando Alfred, un castor famoso, fue invitado al ayuntamiento, le hicieron esta foto.

Cuatro *influencers* escribieron en las redes sociales distintas afirmaciones sobre él. No todo lo que leemos en las redes sociales es verdad.

**¿Cuál de las siguientes afirmaciones de los *influencers* es verdadera?**

**A)** Alfred llevaba chistera pero no llevaba bastón

**B)** Alfred puede que llevara corbata o puede que llevara bastón

**C)** Alfred llevaba chistera y pajarita

**D)** Alfred llevaba pajarita o no llevaba bastón

**Respuesta correcta:** B) Alfred puede que llevara corbata o puede que llevara bastón.

## Explicación

La afirmación verdadera es la **B**.

Hay que decidir, a partir de la foto, si cada afirmación es verdadera o falsa. Cada frase contiene dos afirmaciones unidas por los operadores lógicos Y (en la A, «pero» funciona como Y) y O.

| p | q | p Y q |
|---|---|---|
| F | F | F |
| F | V | F |
| V | F | F |
| V | V | V |

| p | q | p O q |
|---|---|---|
| F | F | F |
| F | V | V |
| V | F | V |
| V | V | V |

Una afirmación con Y solo es verdadera si las dos partes lo son; una con O es verdadera si al menos una parte lo es. En la foto, Alfred lleva chistera y bastón, pero no lleva corbata ni pajarita.

- **A)** llevaba chistera (V) pero no llevaba bastón (F): V Y F = **falso**.
- **B)** llevaba corbata (F) o llevaba bastón (V): F O V = **verdadero**.
- **C)** llevaba chistera (V) y pajarita (F): V Y F = **falso**.
- **D)** llevaba pajarita (F) o no llevaba bastón (F): F O F = **falso**.

## Esto es informática

Y y O son **operadores lógicos** que permiten tomar decisiones combinando varias condiciones. Por ejemplo: si una persona tiene 18 años o más Y no lleva gafas, se le puede conceder el carné de conducir sin restricciones.

Los ordenadores son, en esencia, sistemas binarios. Los operadores lógicos (con 1 para verdadero y 0 para falso) les permiten realizar operaciones lógicas y matemáticas complejas, imprescindibles en muchos campos de la informática: la programación, el diseño de circuitos de control o las consultas a bases de datos. Las tablas que muestran el resultado de un operador para todas las combinaciones posibles se llaman **tablas de verdad**.

## Esto es pensamiento computacional

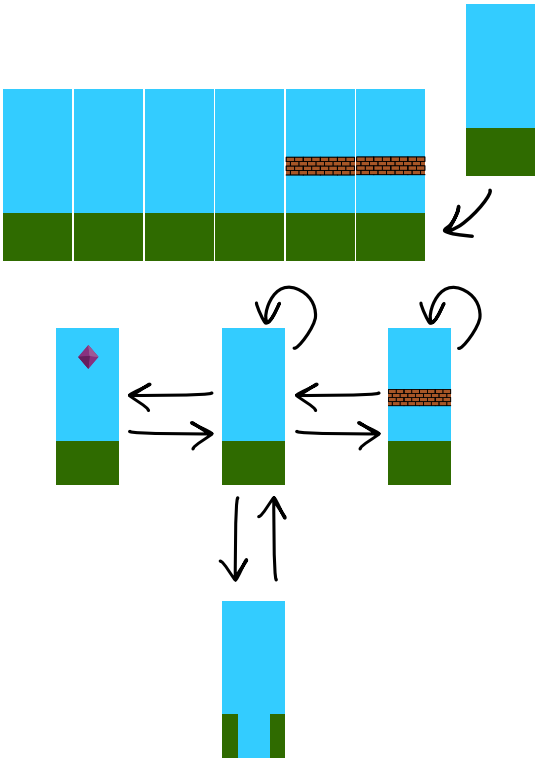
Se usa el **razonamiento lógico**: entender el problema, aplicar de forma sistemática un procedimiento (un pequeño **algoritmo**) para evaluar cada parte de la afirmación a partir de la foto y combinar los resultados según el operador. Es una forma de **evaluación**: comprobar con datos si una afirmación es cierta, algo muy útil también para no creer todo lo que se lee en las redes.

**Palabras clave y enlaces:** Lógica, operadores lógicos, tabla de verdad · es.wikipedia.org/wiki/Tabla_de_verdad

# 6 Superbebras

2024-DE-07 · Alemania
Dificultad: Media

En el videojuego Superbebras, el fondo es una secuencia de baldosas. El ordenador añade continuamente una baldosa nueva por la derecha y, a la vez, quita una por la izquierda. Así crea la ilusión de movimiento.

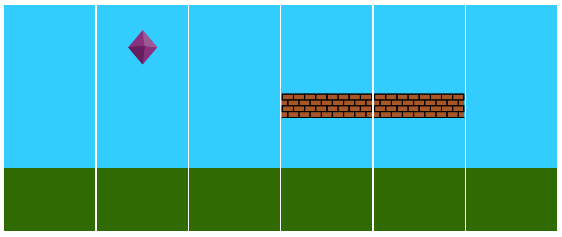


Para elegir la baldosa nueva, el ordenador usa el diagrama de la derecha: mira la baldosa anterior, comprueba las flechas que salen de ella y elige al azar una de las baldosas a las que apuntan.

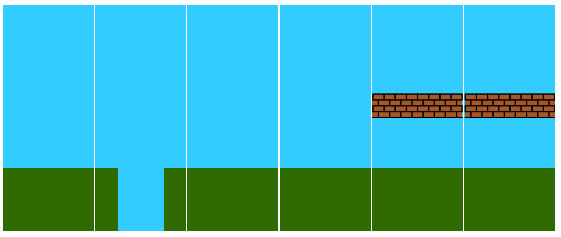
Por ejemplo, después de la baldosa  el ordenador puede elegir la baldosa  o la baldosa .

**Una de las siguientes imágenes NO es un fondo válido de Superbebras. ¿Cuál?**

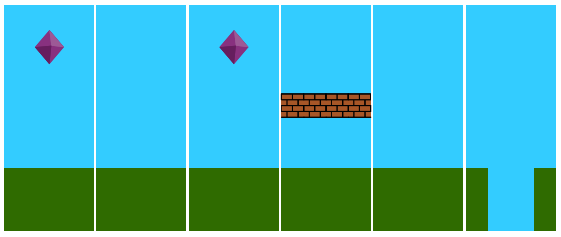
**A)**



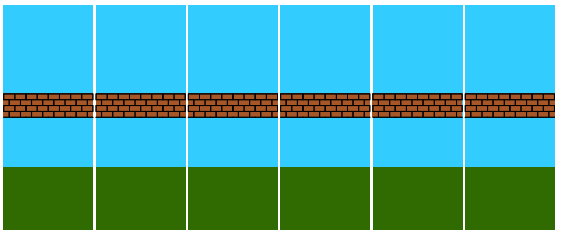
**B)**



**C)**



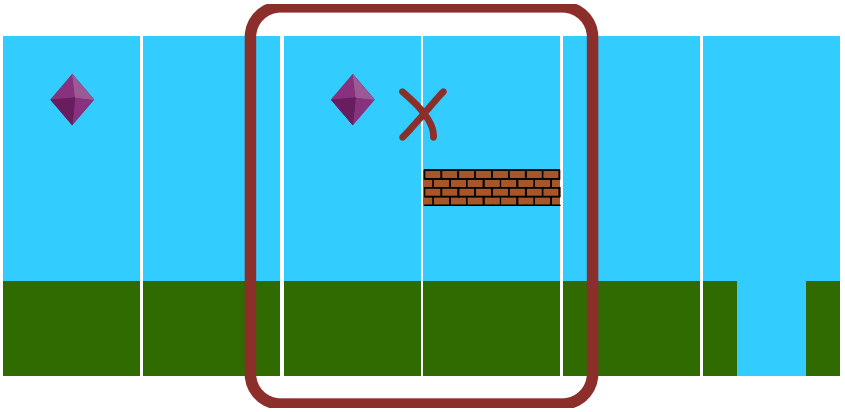
**D)**



**Respuesta correcta:** C.

## Explicación

El fondo de la respuesta **C** no está construido del todo según el diagrama:



Una baldosa con un diamante solo puede ir seguida de , y aquí no es así.

Hay varias formas de encontrar la solución. Una estrategia directa es revisar cada fondo y comprobar cada baldosa con el diagrama. La más rápida es esta: empezar por las baldosas del diagrama de las que sale una sola flecha. Esas flechas son **restricciones**: A → B significa «a la baldosa A le tiene que seguir la baldosa B». Después se comprueba si cada fondo cumple esas restricciones.

## Esto es informática

Algunos videojuegos, como los *endless runners* (juegos de carrera infinita), tienen un fondo que se desplaza horizontalmente y crea la ilusión de que el jugador avanza por un mundo de fantasía. A veces el fondo no es una imagen fija, sino que lo crea automáticamente el ordenador: es la **generación procedimental**. Normalmente se combinan al azar elementos pequeños para crear una enorme variedad de fondos.

Pero la combinación no puede ser totalmente aleatoria: debe cumplir ciertas reglas para no crear situaciones imposibles para el jugador. En la tarea, esas reglas se representan con un diagrama de baldosas y flechas, que es un **grafo dirigido**. Los grafos se usan para modelar todo tipo de situaciones; este en concreto funciona como un autómata que describe qué secuencias son válidas.

## Esto es pensamiento computacional

Hay que leer y usar el grafo dirigido, que determina cómo se compone el fondo y es una **abstracción** del mecanismo que genera las imágenes. También hace falta **reconocimiento de patrones** para dividir visualmente cada fondo en baldosas y detectar las transiciones incorrectas. Por tanto, la tarea pone en juego el **pensamiento algorítmico**, la abstracción y el reconocimiento de patrones.

**Palabras clave y enlaces:** Grafo dirigido, generación procedimental, juegos de plataformas · es.wikipedia.org/wiki/Generación_por_procedimientos

# 7 En busca del tesoro

2024-HU-03 · Hungría
Dificultad: Media



El capitán Boris busca un tesoro escondido en una isla.

Tiene un mapa que muestra dónde está el tesoro. El mapa está dividido en 16 regiones, marcadas con las letras de la A a la P. El capitán puede introducir en un aparato especial cualquier cantidad de regiones, y el aparato le dice si el tesoro está o no en alguna de ellas (por ejemplo, si el aparato responde «sí» para A + C + D, el tesoro está en la región A, en la C o en la D).




**¿Cuál es el menor número de veces que el capitán Boris tendría que usar el aparato para estar seguro de encontrar siempre el tesoro?**

Respuesta: _____

**Respuesta correcta:** 4.

## Explicación

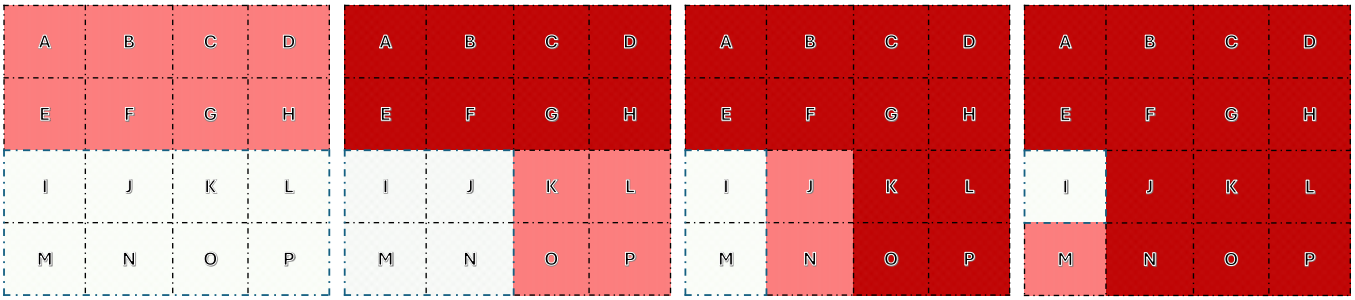
La respuesta es **4**.

Primero, el capitán debe introducir la mitad de las regiones. Puede ocurrir:

a. Que el aparato diga que el tesoro está en esas regiones: entonces divide esas regiones en dos mitades e introduce una de ellas.
b. Que el aparato diga que no está: entonces divide las regiones restantes en dos mitades e introduce una de ellas.

Repite el proceso hasta encontrar la región del tesoro. Cada consulta reduce a la mitad las posibilidades: 16 → 8 → 4 → 2 → 1, es decir, 4 consultas. Con menos no basta: con 3 respuestas sí/no solo se distinguen $2^3 = 8$ casos, y hay 16 regiones.

Por ejemplo, si introduce A, B, C, D, E, F, G, H y responde «no», puede introducir K, L, O, P (valdrían cualesquiera 4 de las restantes). Si vuelve a responder «no», introduce 2 de las que quedan (por ejemplo J y N). Si responde «no», el tesoro está en I o en M. Si introduce M y responde «no», sabe con seguridad que está en I.




## Esto es informática

Para encontrar el tesoro de la forma más eficiente hay que dividir las regiones por la mitad. En informática se usa una técnica parecida para encontrar un elemento en una lista ordenada: dividir repetidamente por la mitad el espacio de búsqueda. Así se descarta la mitad de los elementos en cada paso y se encuentra el elemento en el menor número de pasos. Es la **búsqueda binaria**, que en una lista de n elementos necesita unos $\log_2 n$ pasos.

La búsqueda binaria tiene muchas aplicaciones: buscar una palabra en un diccionario, localizar un registro en una base de datos, encontrar en qué versión de un programa apareció un error (como hace `git bisect`), o decidir con el menor número de preguntas sí/no, como en el juego de «¿Quién es quién?».

## Esto es pensamiento computacional

Es una tarea de **descomposición**: dividir el espacio de búsqueda en mitades para reducir el problema a otro más pequeño del mismo tipo (estrategia de «divide y vencerás»). Además hay que **evaluar** la estrategia en el peor caso: no basta con que funcione si hay suerte, debe garantizar el resultado siempre, y hay que argumentar que no existe una estrategia con menos consultas.

**Palabras clave y enlaces:** Búsqueda binaria, divide y vencerás, logaritmo · es.wikipedia.org/wiki/Búsqueda_binaria

# 8 La máquina de globos

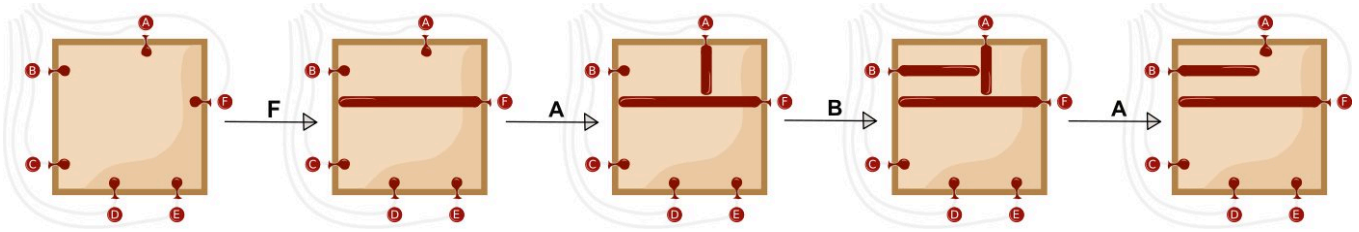
2024-DE-06a · Alemania
Dificultad: — (difícil en 14–16)

Los castores tienen una máquina que crea imágenes inflando globos dentro de un marco cuadrado. Los globos están marcados con las letras A, B, C, D, E y F.

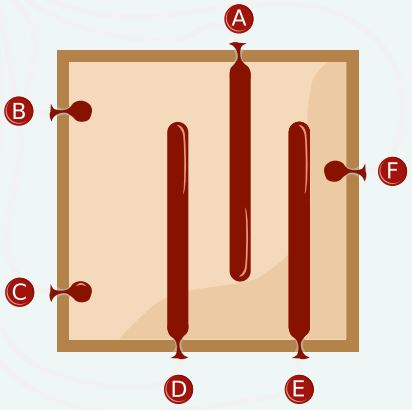
La máquina lee letras de una en una. Cuando lee una letra:

- Si el globo con esa letra está desinflado, lo infla hasta que toca otro globo o el borde opuesto del marco.
- Si no, desinfla el globo con esa letra.

Por ejemplo, si al principio todos los globos están desinflados y la máquina lee F, A, B y después A, hace lo siguiente:




**Al principio todos los globos están desinflados y la máquina lee una secuencia de nueve letras. El resultado es la imagen de abajo. Escribe esa secuencia de nueve letras.**



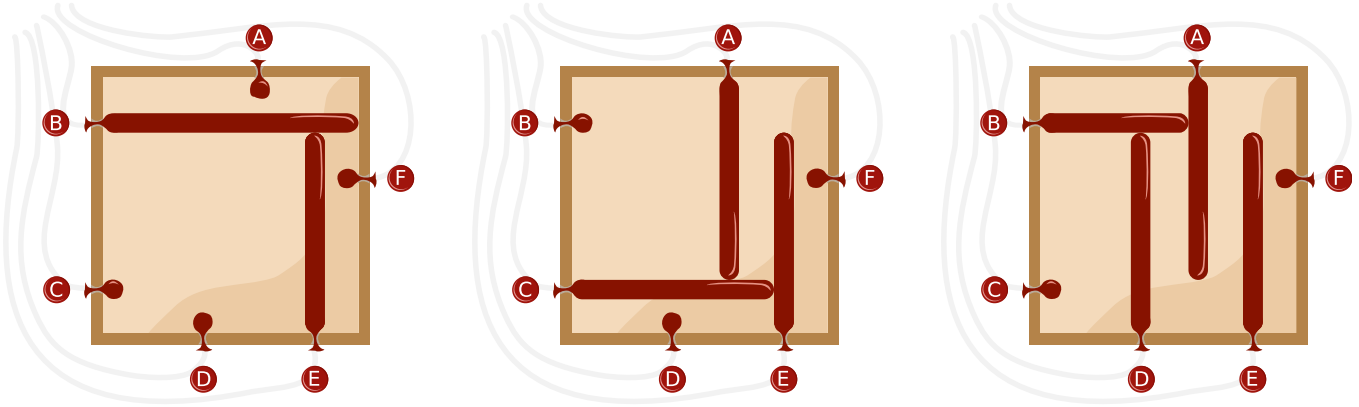
**Respuesta correcta:** Por ejemplo, B E B C A C B D B (hay 4 respuestas válidas).

## Explicación

Hay **4 respuestas correctas** de nueve letras:

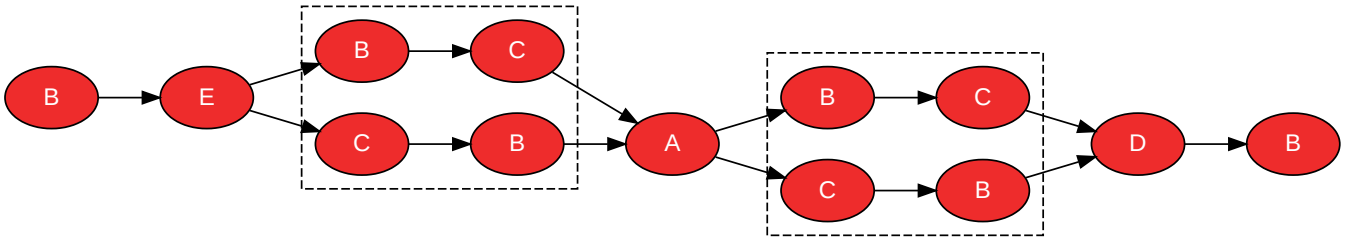
B E B C A C B D B · B E C B A C B D B · B E B C A B C D B · B E C B A B C D B

Explicamos en detalle la primera, B E B C A C B D B. Las imágenes muestran estados intermedios:




- Primera imagen: estado después de B E. Hay que inflar B antes que E para que B sirva de tope al inflar E.
- Segunda imagen: después de B E B C A. El globo C sirve de tope para A, así que debe inflarse antes que A. Además, B debe desinflarse antes de inflar A; si no, A no llegaría a la longitud necesaria.
- Tercera imagen: después de B E B C A C B D. B sirve de tope para D, y C debe desinflarse antes de inflar D.

Para encontrar todas las respuestas ayuda representar el problema con un **grafo dirigido**: una flecha de B a E indica que B debe inflarse antes que E, etc. Hay dos lugares en los que el orden de B y C puede variar (marcados con un rectángulo discontinuo).




Cada respuesta correcta es una **ordenación topológica** de este grafo; como hay 4, hay cuatro soluciones. Un truco para comprobar una respuesta: los globos que acaban inflados (A, D, E) deben aparecer un número impar de veces, y los que acaban desinflados (B, C, F), un número par (incluido 0).

## Esto es informática

La secuencia de letras es un **programa** que controla una máquina. Cada letra es una instrucción que infla o desinfla un globo. Como en casi todos los programas de este tipo, el **orden de las instrucciones** es esencial: la secuencia B E produce una imagen distinta de E B.

El grafo dirigido de la explicación no tiene ciclos, por eso se llama **grafo acíclico dirigido** (DAG). Todo DAG tiene al menos una ordenación topológica y se conocen algoritmos rápidos para construirla. Los DAG tienen muchas aplicaciones en informática, por ejemplo en la planificación de tareas o en la compilación de programas (qué hay que compilar antes de qué).

## Esto es pensamiento computacional

Hace falta **pensamiento algorítmico**: crear un algoritmo, un conjunto de instrucciones para la máquina, en el que el orden es esencial. El **reconocimiento de patrones** ayuda a deducir cuántas veces debe aparecer cada letra. Y es importante **evaluar** la solución: se puede ejecutar el algoritmo mentalmente, pero también usar atajos para descartar secuencias que nunca pueden ser correctas (por ejemplo, contando cuántas veces aparece cada letra). Si el recuento es correcto, todavía hay que comprobar que el resultado es el pedido.

**Palabras clave y enlaces:** Programación, secuencia, grafo acíclico dirigido, ordenación topológica · es.wikipedia.org/wiki/Ordenamiento_topológico · es.wikipedia.org/wiki/Grafo_acíclico_dirigido

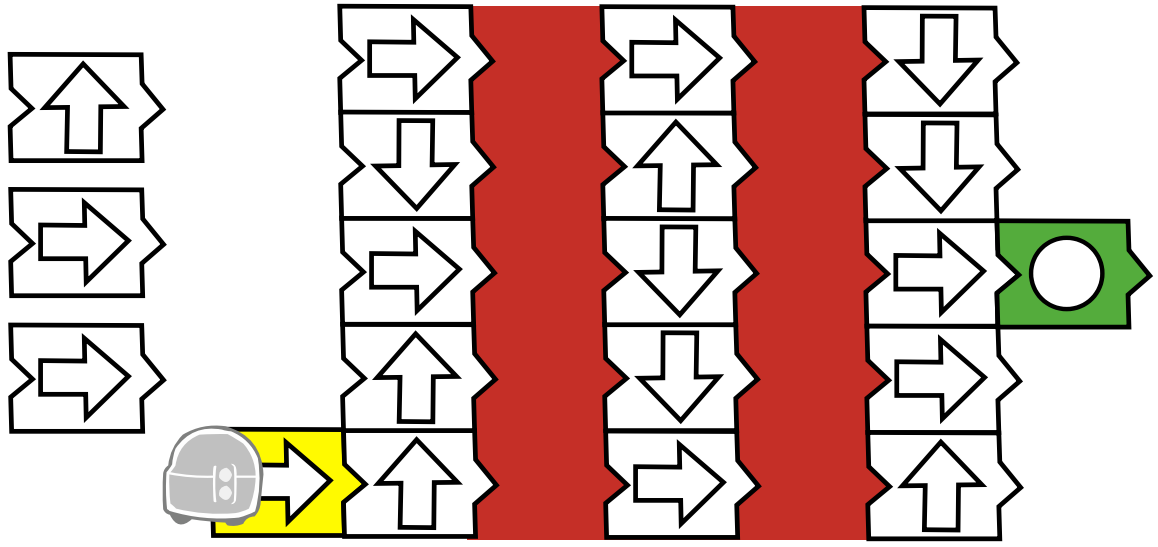
# 9 Un camino para el robot

2024-PL-04 · Polonia
Dificultad: Fácil

El robot de Tom se mueve sobre unas fichas siguiendo estas reglas:

- El robot empieza en la ficha amarilla.
- El robot avanza en la dirección de la flecha de la ficha en la que está.
- Si el robot se sale de las fichas, se detiene.

Las fichas ya están colocadas como se ve en la imagen. Tom tiene además 3 fichas extra (a la izquierda).



**Ayuda a Tom: coloca las 3 fichas extra en las zonas vacías (rojas) para que su robot llegue a la ficha verde con el círculo.**

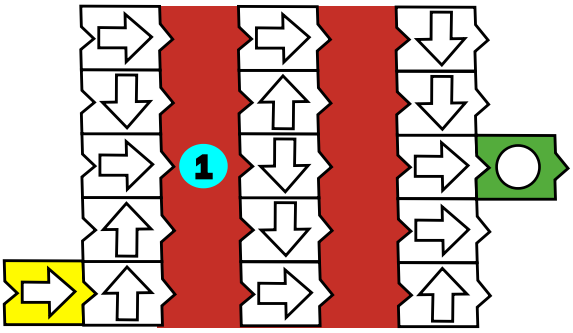
Dibuja en las zonas rojas de la imagen dónde colocarías cada una de las 3 fichas extra.

**Respuesta correcta:** ↑ en la 2.ª columna (3.ª fila), → en la 2.ª columna (2.ª fila) y → en la 4.ª columna (1.ª fila).

## Explicación

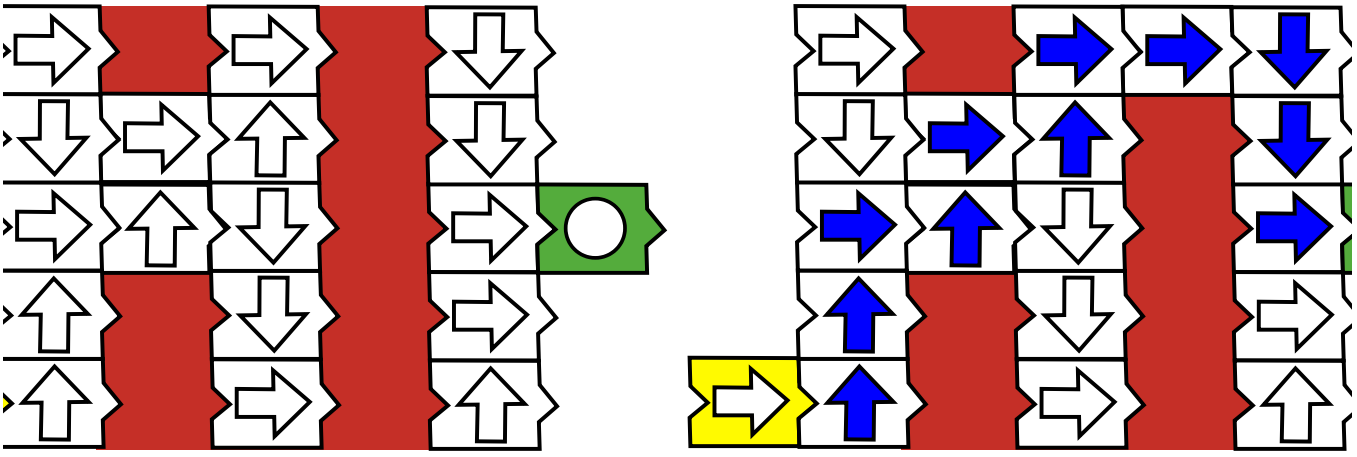
El objetivo es pasar de una columna a otra con las tres fichas hasta llegar al círculo. Solo hay una forma de hacerlo.

Para pasar de la primera columna a la segunda, el robot solo puede salir por la tercera ficha de la primera columna, así que la primera ficha se coloca en la posición marcada con 1.

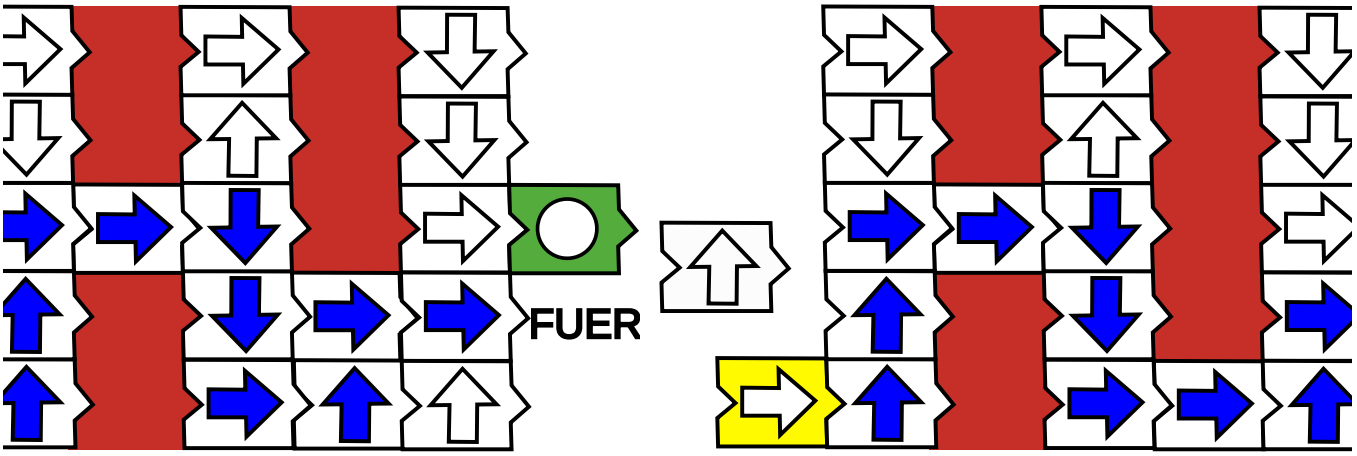



Desde ahí hay dos opciones: ir hacia arriba o hacia la derecha.

**Opción 1: hacia arriba.** Hace falta otra ficha para pasar a la segunda columna. Siguiendo las flechas de la segunda columna, el robot termina en su primera ficha. Con la última ficha cruza a la tercera columna y llega al círculo.



**Opción 2: hacia la derecha.** El robot llega a la ficha de abajo de la segunda columna y de nuevo hay dos posibilidades: a) usar la ficha hacia arriba y después la tercera ficha: el robot no llega al círculo y se sale; b) usar la ficha hacia la derecha: tampoco llega y, además, sobra una ficha, lo que va contra las reglas.




## Esto es informática

En esta tarea las fichas dirigen el movimiento de un robot y hay que encontrar la ruta que lleva al final. Hay cosas que se saben de antemano: solo se pueden usar tres fichas (dos con flecha a la derecha y una hacia arriba) y solo en la zona roja, así que en cada

columna hay que encontrar una flecha a la derecha para avanzar. En informática esto se llama **restricciones**. Sin ellas el problema podría tener muchas más soluciones y sería más difícil, porque habría que pensar en muchísimos resultados posibles.

Por ejemplo, para ir de un lugar a otro puede haber muchas rutas, pero añadir algunas restricciones (pasar por una tienda, hacer parte del camino en autobús) reduce las opciones y ayuda a elegir la que mejor nos conviene. Los robots y los programas de navegación trabajan así.

## Esto es pensamiento computacional

La tarea requiere sobre todo dos destrezas: **descomposición** y **evaluación**. Primero se descompone el problema en problemas más pequeños: en cada columna, cómo pasar a la siguiente. Así solo hay que pensar en una parte de la ruta y hay menos opciones. Después se evalúa el resultado de cada posibilidad y se descartan las que no llevan a una buena solución.

**Palabras clave y enlaces:** Restricciones, ruta, camino, robótica · es.wikipedia.org/wiki/Problema_de_satisfacción_de_restricciones

# 10 Días soleados

2024-DE-04a · Alemania
Dificultad: Fácil



El castor Tom dice: «Los días soleados, en todos los estanques hay al menos un castor nadando».

Su hermana Kim responde: «¡Eso no es verdad! El domingo pasado es un ejemplo que demuestra que te equivocas».

**Supón que Kim tiene razón. Completa la frase:**
**El domingo pasado hizo sol y _____.**

**A)** todos los estanques estaban llenos de castores nadando
**B)** ningún castor nadó en el estanque de la cascada helada
**C)** el castor Miguel nadó por todos los estanques
**D)** el castor Miguel no nadó nada

**Respuesta correcta:** B) ningún castor nadó en el estanque de la cascada helada.

## Explicación

La solución es la **B**: ningún castor nadó en el estanque de la cascada helada.

En ese caso no todos los estanques tienen un castor nadando, porque el de la cascada helada no lo tiene. Además, sabemos que el domingo pasado hizo sol. Así que el domingo pasado es un ejemplo de día soleado con al menos un estanque sin castores nadando. Por tanto, Tom se equivoca al decir que todos los días soleados hay al menos un castor nadando en cada estanque.

Las demás opciones no demuestran que Tom esté equivocado:

- **A**: «Hizo sol y todos los estanques estaban llenos de castores nadando»: en todos los estanques había castores, lo que coincide con lo que dice Tom.
- **C**: «Hizo sol y el castor Miguel nadó por todos los estanques»: en todos había al menos un castor (Miguel), de nuevo de acuerdo con Tom.
- **D**: «Hizo sol y el castor Miguel no nadó nada»: puede que otros castores nadaran en todos los estanques, y entonces lo que dice Tom se seguiría cumpliendo.

## Esto es informática

Tom y Kim discuten sobre el valor de verdad de la afirmación de Tom. Entran así en el terreno de la **lógica booleana**, en la que cada afirmación es verdadera o falsa. La lógica permite construir afirmaciones complejas con operadores como Y, O, NO y con **cuantificadores** como «para todo» y «existe al menos uno».

Para refutar una afirmación del tipo «para todo...» basta con un **contraejemplo**: un caso concreto que no la cumpla. La lógica ordena las discusiones y ayuda a evitar malentendidos. Además es la base de los ordenadores: los circuitos están hechos de puertas lógicas como NO e Y, y los lenguajes de programación incluyen condiciones lógicas como «SI es verano Y hace sol, ENTONCES enciende el aire acondicionado».

## Esto es pensamiento computacional

Como los ordenadores se basan en la lógica, trabajar con ellos exige manejarla: traducir una situación del mundo real a una condición lógica, o encontrar la negación exacta de una condición para que un programa contemple todos los casos. Esta tarea entrena justamente eso, dentro del **análisis e interpretación de datos**. También hay algo de **descomposición**, porque conviene analizar cada opción por separado.

**Palabras clave y enlaces:** Lógica, cuantificador, contraejemplo · es.wikipedia.org/wiki/Cuantificador

# 11 Clase en línea

2024-MY-03 · Malasia
Dificultad: — (fácil en 10–12)

La profesora Ava da una clase en línea desde su casa.

En la pantalla de su ordenador ve que se han conectado 9 alumnos: Emma, Maya, Bella, Lee, Raúl, Hannah, Diana, Alice y James. Cada uno usa un ordenador distinto de la biblioteca del colegio.




Como los alumnos están sentados uno al lado de otro en la biblioteca, en cada imagen Ava ve también a los compañeros que tiene al lado cada alumno. Así sabe que todos están sentados en fila.

**¿Cuál de los 9 alumnos está sentado en el centro (en la 5.ª posición)?**

**A)** Diana
**B)** Raúl
**C)** Lee
**D)** Bella
**E)** Hannah

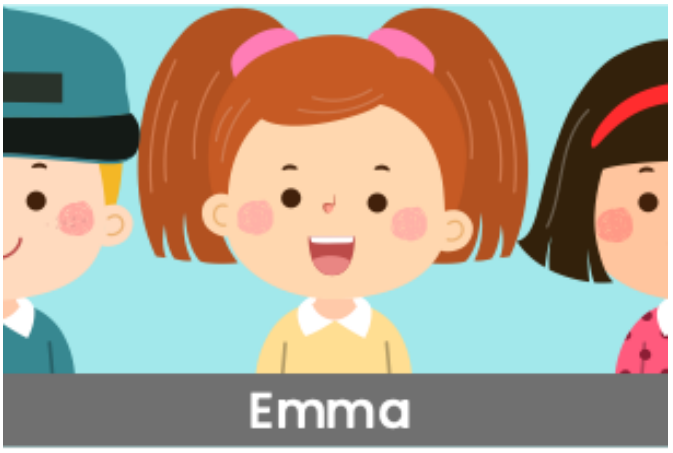
**Respuesta correcta:** E) Hannah.

## Explicación

La respuesta es **Hannah**.

Por lo que se ve en la pantalla, sabemos quién está al lado de cada alumno. En la imagen de James vemos que se sienta junto a Emma, y en la de Emma vemos que está entre James y Diana:




Haciendo lo mismo con el resto de imágenes se obtiene el orden de los 9 alumnos:




`James · Emma · Diana · Lee · **Hannah** · Bella · Alice · Raúl · Maya`

Quien está en el centro ocupa la 5.ª posición contando desde la izquierda o desde la derecha: Hannah.

## Esto es informática

En la tarea, el programa de videoconferencia muestra a los alumnos en recuadros colocados en un orden cualquiera, y hay que reconstruir el orden real para saber quién está en el centro.

La disposición de los alumnos es una estructura de datos llamada **lista doblemente enlazada**. En ella, cada nodo contiene un elemento, una referencia al nodo anterior y otra al siguiente. Aquí cada recuadro (nodo) muestra a un alumno (el elemento) y, a sus lados, a sus vecinos (las referencias). Resolver la tarea es **recorrer la lista** empezando por uno de los extremos, que se reconoce porque solo tiene un vecino.

## Esto es pensamiento computacional

Hace falta **pensamiento lógico**: comprender el problema, idear un procedimiento (**algoritmo**) para ir encadenando la información de cada imagen y aplicarlo con orden. Es como montar un puzle: buscar primero las piezas de los bordes (aquí, los alumnos de los extremos) y avanzar pieza a pieza a partir de sus conexiones.

**Palabras clave y enlaces:** Lista doblemente enlazada, recorrido de listas · es.wikipedia.org/wiki/Lista_doblemente_enlazada

# 12 Cadenas de palabras

2024-CA-02 · Canadá
Dificultad: Difícil




El profesor Castor está enseñando a leer a su alumnado. Para ayudarles crea **cadenas de palabras**: secuencias de palabras en las que, para obtener la palabra siguiente, se cambia exactamente una letra. Por ejemplo, MUG → MUD → MAD → FAD es una cadena de palabras.

El profesor Castor tiene estas nueve palabras: BOT, SAD, BAT, CAB, COT, BAD, COB, CAT y SAT. Las agrupa en tres cadenas de tres palabras cada una, de modo que cada una de las nueve palabras aparece en una y solo una de las cadenas.

(Las palabras están en inglés, como en la tarea original; para resolverla solo importan las letras.)

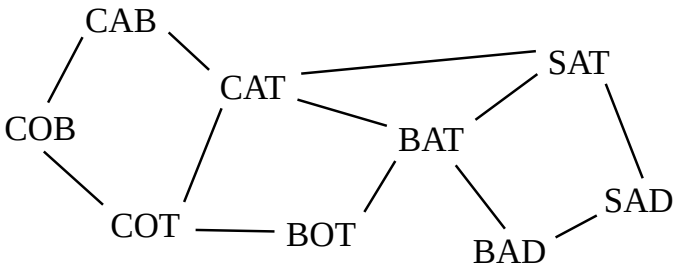
**Ninguna de las siguientes secuencias incumple las reglas de las cadenas, pero una de ellas hace imposible formar tres cadenas sin repetir palabras. ¿Cuál no podría ser una de las cadenas del profesor Castor?**

**A)** SAD → BAD → BAT
**B)** COT → COB → CAB
**C)** BAT → CAT → COT
**D)** CAT → CAB → COB

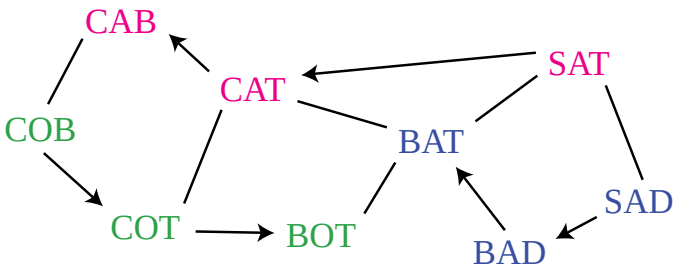
**Respuesta correcta:** C) BAT → CAT → COT.

## Explicación

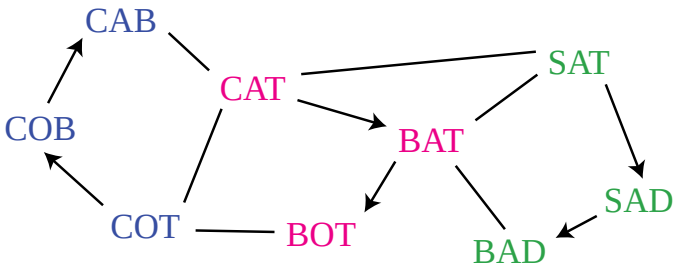
La respuesta es la **C) BAT → CAT → COT**. Para resolverlo ayuda dibujar un diagrama en el que dos palabras se unen con una línea si pueden ir seguidas en una cadena.



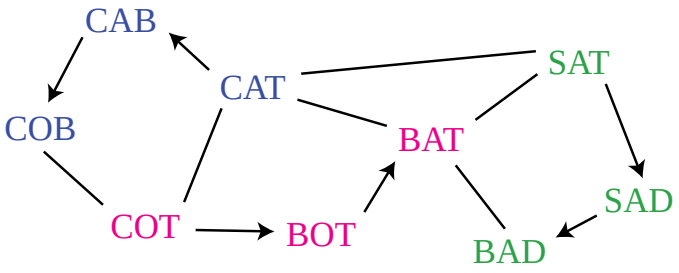

**A** es posible: las otras dos cadenas podrían ser SAT → CAT → CAB y COB → COT → BOT.




**B** es posible: las otras dos podrían ser CAT → BAT → BOT y SAT → SAD → BAD.




**D** es posible: las otras dos podrían ser COT → BOT → BAT y SAT → SAD → BAD.




**C** no es posible: en el diagrama, BOT solo está unida a BAT y a COT, así que tiene que estar en una cadena con al menos una de ellas. Como BAT y COT están las dos en la cadena de la opción C y BOT no, no queda ninguna cadena posible para BOT.

## Esto es informática

En informática, este tipo de diagrama se llama **grafo** y sirve para mostrar conexiones. Está formado por un conjunto de vértices y un conjunto de aristas, cada una de las cuales une dos vértices. Aquí los vértices son las palabras y hay una arista entre dos de ellas si se diferencian en exactamente una letra.

Una cadena de palabras es un **camino** por aristas conectadas. Como hay un camino entre dos vértices cualesquiera, el grafo es **conexo**. Agrupar las nueve palabras en tres cadenas de tres equivale a quitar aristas hasta obtener un grafo no conexo con tres **componentes** de tres vértices cada una (una partición en caminos). La pregunta consiste en comprobar si un conjunto dado de tres palabras puede ser una de esas componentes.

## Esto es pensamiento computacional

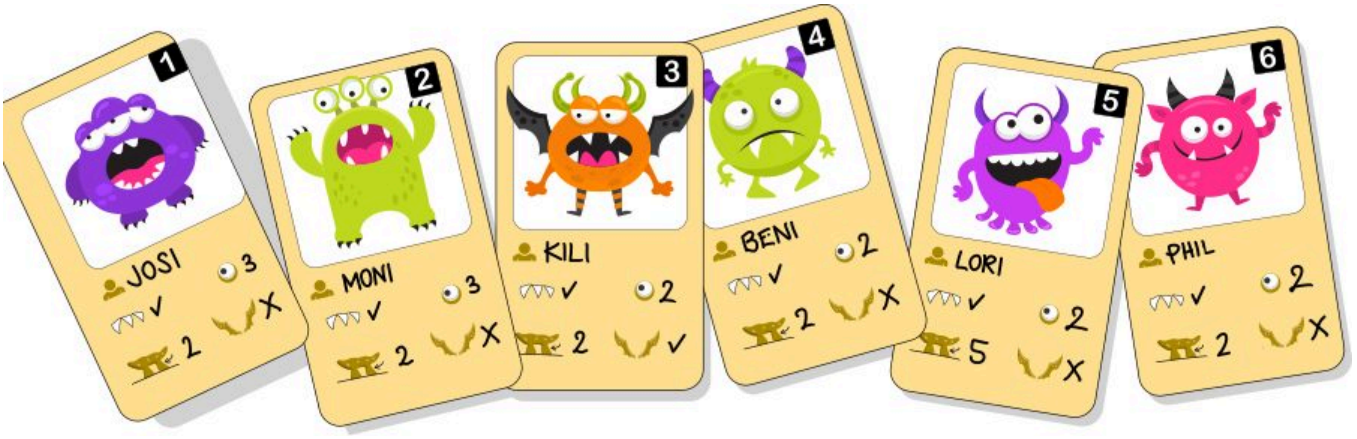
Hay que **abstraer** el problema a un grafo para verlo de un vistazo y **evaluar** cada opción: comprobar si, tras usar esa cadena, las seis palabras restantes se pueden repartir en dos cadenas válidas. La clave para descartar rápido es fijarse en las palabras con pocas conexiones (como BOT), que son las que más restringen las soluciones.

**Palabras clave y enlaces:** Grafo, camino, grafo conexo, componente conexa · es.wikipedia.org/wiki/Grafo_conexo

# 13 Las cartas Ricca

2024-CH-03a · Suiza
Dificultad: Fácil

Bárbara colecciona cartas de monstruos llamados «Riccas». Estas son algunas cartas de su enorme colección:




Los Riccas tienen **atributos**, como el nombre o los dientes . Cada Ricca es distinto, así que los **valores** de los atributos varían. Por ejemplo, en la carta 2 el valor del atributo es «MONI», el nombre del Ricca, y el valor del atributo es «✓», que indica que el Ricca tiene dientes.

Para tener una visión rápida de su colección, prepara una tabla en el ordenador. Pero ¿cómo anotar los valores de las cartas? A cada atributo le asigna uno de estos **tipos**:

- **no/sí** valor «X» o «✓»
- **ABC...** valor en forma de texto
- **123...** valor en forma de número

**Ayuda a Bárbara a terminar de preparar la tabla: indica el tipo de cada columna.**



Escribe en cada casilla gris el tipo: no/sí, ABC... o 123...

**Respuesta correcta:** nombre: ABC… · ojos: 123… · patas: 123… · alas: no/sí · dientes: no/sí.

## Explicación

Esta es la solución:

|  |  |  |  |  |
|---|---|---|---|---|
| ABC… | 123… | 123… | no/sí | no/sí |

Cada atributo tiene valores distintos en las cartas Ricca:

| |  |  |  |  |  |
|---|---|---|---|---|---|
| 1 | Josi | 3 | 2 | no | sí |
| 2 | Moni | 3 | 2 | no | sí |
| 3 | Kili | 2 | 2 | sí | sí |
| 4 | Beni | 2 | 2 | no | sí |
| 5 | Lori | 2 | 5 | no | sí |
| 6 | Phil | 2 | 2 | no | sí |

Los valores «X» y «✓» de las alas  y los dientes  se representan con el tipo no/sí. Los nombres Josi, Moni, Kili, Beni, Lori y Phil se representan con el tipo ABC…. Y los valores 1, 2, 3… de los ojos  y las patas , con el tipo 123….

Si todos los valores se anotan correctamente y con el mismo tipo, la colección se puede ordenar por cualquiera de los atributos.

## Esto es informática

Para guardar datos de forma organizada hay que pensar no solo en cómo estructurarlos, sino también en qué **tipo de dato** usar para cada valor.

Los ordenadores lo guardan todo con unos y ceros, que podrían representar cualquier cosa; los tipos de datos le dicen al ordenador cómo interpretarlos y manipularlos. En programación también sirven para comprobar que se usa la clase de información correcta. En la tabla de Bárbara hay tres tipos:

- ABC… **cadena de texto** (*string*): datos formados por caracteres de un alfabeto.
- 123… **número entero** (*integer*): se puede limitar qué números se aceptan; aquí solo tienen sentido los enteros, porque ningún Ricca tiene 3,5 patas.
- no/sí **booleano**: solo representa los valores verdadero o falso.

El tipo adecuado depende del uso que se quiera dar a los datos. Si alguien quisiera saber cuántas alas tiene cada Ricca, el booleano no bastaría y habría que usar también un número. Y hay datos que no se pueden ordenar, como el color, que puede representarse con palabras («morado») o con números, como los códigos hexadecimales de la web (#8E36D9) o las cartas RAL.

## Esto es pensamiento computacional

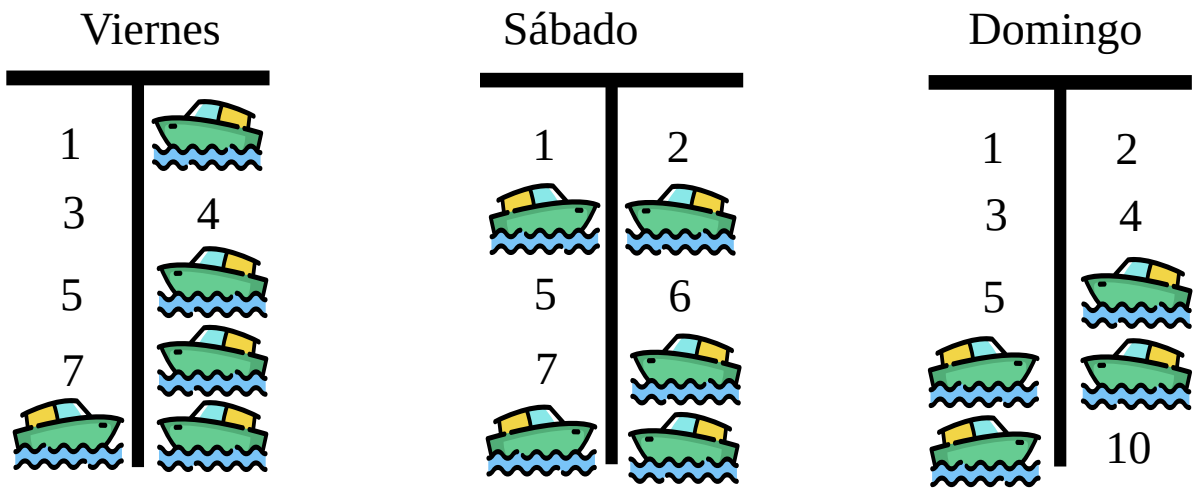
**Abstracción**: hay que filtrar la información relevante (los atributos) y asignar a los valores de cada uno un tipo de dato. Al crear una tabla o una base de datos es importante elegir bien los tipos para poder filtrar y buscar después. Saber montar una base de datos bien estructurada es muy útil, por ejemplo para gestionar el catálogo de una tienda o la producción de una fábrica.

**Palabras clave y enlaces:** Tipos de datos, cadena, entero, booleano · es.wikipedia.org/wiki/Tipo_de_dato · es.wikipedia.org/wiki/Tipo_de_dato_lógico

# 14 Plazas libres en el puerto

2024-MT-02 · Malta
Dificultad: — (fácil en 10–12)

En un puerto deportivo hay diez amarres, que pueden estar reservados (con un barco) o libres, como se ve en la imagen. Por ejemplo, el amarre 1 está libre el viernes, pero el amarre 2 está reservado el viernes.




Tom tiene que elegir si llega el viernes o el sábado. Además, tiene que elegir un amarre que pueda reservar durante dos días seguidos. Por ejemplo, una opción es llegar el sábado y reservar el amarre 1.

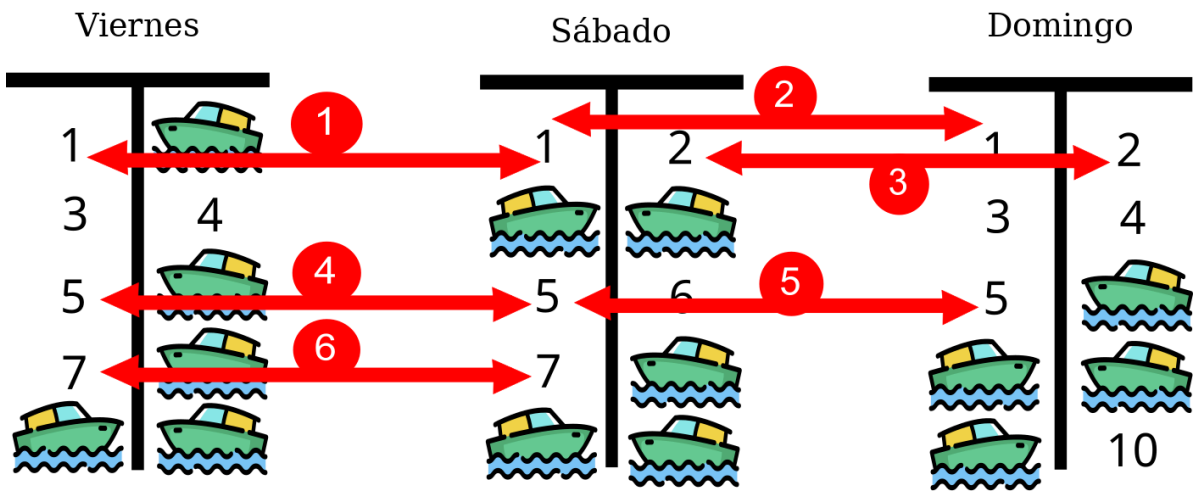
**¿Cuántas opciones tiene Tom?**

**A)** 4
**B)** 5
**C)** 6
**D)** 8
**E)** 10

**Respuesta correcta:** C) 6.

## Explicación

La respuesta correcta es la **C) 6**. Estas son las seis opciones que tiene Tom:




- Llegando el viernes (viernes y sábado), están libres los dos días los amarres 1, 5 y 7.
- Llegando el sábado (sábado y domingo), están libres los dos días los amarres 1, 2 y 5.

En total, 3 + 3 = 6 opciones.

Nota: en el documento original la explicación indica la opción B (5), pero tanto su imagen de la solución como su propio cálculo dan 6, que es la respuesta correcta.

## Esto es informática

Todos los datos se pueden ver como secuencias de ceros y unos. Cada cero o uno es un **bit**, y la secuencia es un código o número **binario**.

Aquí podemos representar un amarre libre con 0 y uno reservado con 1; cada amarre es un bit. Así, cada día se convierte en una secuencia de 10 bits (del amarre 1 al 10): viernes 0100010111, sábado 0011000111 y domingo 0000011110.

La tarea pide buscar, en dos días seguidos, los bits (amarres) que valen 0 en los dos. Eso corresponde al operador lógico **NOR** («ni... ni...»), que da 1 solo cuando las dos entradas son 0:

```
0100010111 NOR 0011000111 = 1000101000
0011000111 NOR 0000011110 = 1100100000
```

El número total de unos en los dos resultados (3 + 3) es la respuesta de la tarea.

## Esto es pensamiento computacional

Hace falta **reconocimiento de patrones** para distinguir qué amarres están libres cada día y detectar las secuencias de libres y reservados a lo largo de los tres días. La **descomposición** ayuda a examinar cada amarre por separado, o cada pareja de días por separado, y sumar después los resultados.

**Palabras clave y enlaces:** Binario, bit, operador NOR · es.wikipedia.org/wiki/Puerta_NOR

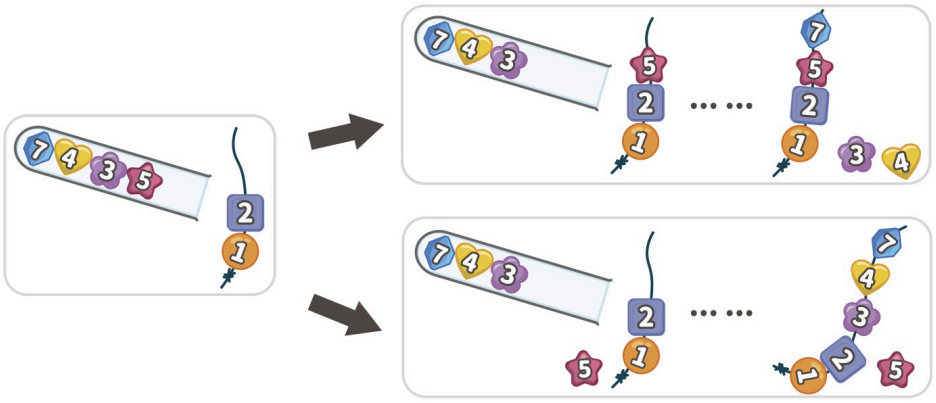
# 15 La pulsera Bebras

2024-TW-03 · Taiwán
Dificultad: Media

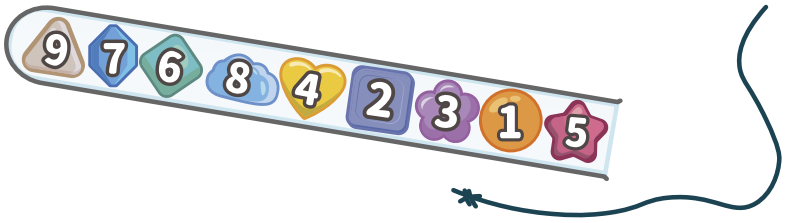
Castor está haciendo una pulsera Bebras. Saca de un tubo cuentas numeradas, de una en una. Con cada cuenta decide si la ensarta en el hilo o la aparta y no la usa. Pero solo puede ensartar una cuenta si:

- el hilo está vacío, o
- el número de la cuenta es mayor que el de la última cuenta ensartada.

En este ejemplo, la última cuenta del hilo es la n.º 2. Así que Castor puede ensartar la cuenta n.º 5 o apartarla.



Castor va a hacer una pulsera nueva con las cuentas de este tubo (la primera que sale es la 5):



**¿Cuál es el mayor número de cuentas que puede ensartar?**

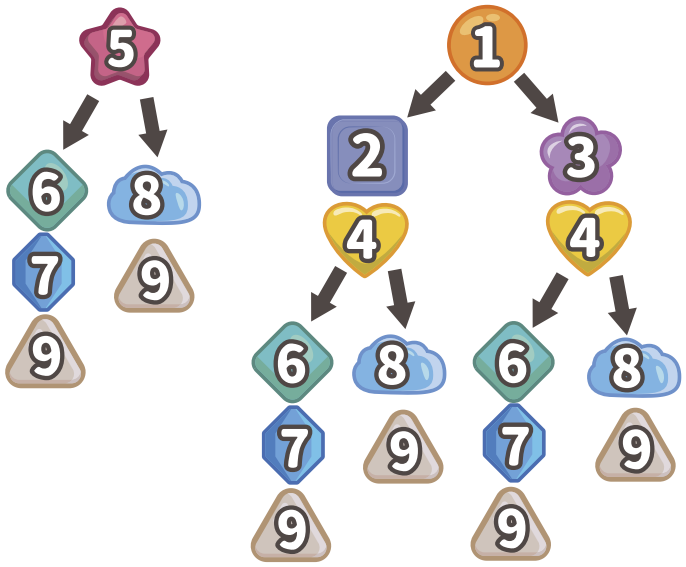
**A)** 3 cuentas
**B)** 4 cuentas
**C)** 5 cuentas
**D)** 6 cuentas
**E)** 7 cuentas

**Respuesta correcta:** D) 6 cuentas.

## Explicación

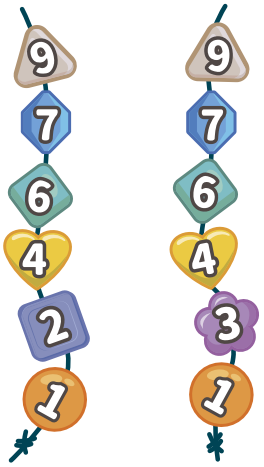
La respuesta correcta es la **D) 6 cuentas**.

Podemos construir todas las pulseras posibles con este tubo y ver cuáles tienen más cuentas. La primera cuenta que sale es la 5. Si se ensarta, después solo se pueden ensartar la 6, la 7, la 8 y la 9, y las secuencias más largas posibles son 5 6 7 9 y 5 8 9. Si se aparta la 5 y se ensarta primero la siguiente (la 1), hay más secuencias y más largas:




Si se empieza por cualquier otra cuenta, la secuencia más larga posible ya está incluida en una de las anteriores, que empiezan por la 1 o por la 5. Por ejemplo, empezar por la 2 lleva a secuencias como 2 4 6 7 9 o 2 4 8 9, incluidas en 1 2 4 6 7 9 y 1 2 4 8 9.

Por tanto, las pulseras con más cuentas empiezan por la 1 y tienen 6 cuentas: **1 2 4 6 7 9** y **1 3 4 6 7 9**.




## Esto es informática

En informática, una **secuencia** es una serie ordenada de datos, como una cadena de caracteres o una secuencia de números. Cada miembro es un elemento, y una **subsecuencia** se forma eligiendo elementos de la secuencia original sin cambiar su orden.

Tanto las cuentas del tubo (en el orden en que salen) como las pulseras son secuencias de números si ignoramos la forma y el color. La tarea pide encontrar la **subsecuencia creciente más larga** (*longest increasing subsequence*) de la secuencia del tubo.

Con secuencias mucho más largas, enumerar todas las subsecuencias crecientes llevaría demasiado tiempo. Por suerte, existen algoritmos eficientes que la encuentran en un tiempo aproximadamente proporcional a n·log n: para un millón de elementos, del orden de veinte millones de pasos, nada para un ordenador moderno. Muchos problemas de subsecuencias se resuelven con la técnica de **programación dinámica**.

## Esto es pensamiento computacional

Enumerar todas las soluciones posibles llevaría mucho tiempo, y muchas ni siquiera cumplirían los requisitos. Por eso la clave es encontrar una forma eficiente de identificar las soluciones posibles y descartar las imposibles: **pensamiento algorítmico**. Además, al decidir si se ensarta una cuenta hay que valorar cómo limita las cuentas que se podrán usar después; por ejemplo, ensartar la 5 impide usar de la 1 a la 4. Esa **simulación** de las consecuencias de cada decisión es otro aspecto del pensamiento computacional.

**Palabras clave y enlaces:** Subsecuencia creciente más larga, programación dinámica · en.wikipedia.org/wiki/Longest_increasing_subsequence · es.wikipedia.org/wiki/Programación_dinámica

## Créditos y licencia

Las tareas proceden del taller internacional Bebras 2024 (Serbia). Cada tarea indica su código y el país que la propuso. Las imágenes son de los autores y autoras de las tareas originales.

Traducción y adaptación al castellano: Bebras España (bebras.ehu.es).

Copyright © Bebras – International Challenge on Informatics and Computational Thinking. Esta obra está bajo una licencia Creative Commons Reconocimiento-CompartirIgual 4.0 Internacional (CC BY-SA 4.0): creativecommons.org/licenses/by-sa/4.0.