AdaByron 2016

Estadísticas y Soluciones

CORITEL
An Accenture Company

Facultad de Informática

UNIVERSIDAD COMPLUTENSE MADRID

Clasificación de los problemas

Problema	Categoría
A - La máquina calculadora	Grafos, búsqueda en anchura
B - Palmeras en la nieve	Iteración, segmento de longitud máxima
C - Double decker	Expresiones
D - Tiro al patíndromo	Programación dinámica, cadenas
E - El conteo de la rosa	Divide y vencerás
F - El teorema del punto fijo	Permutaciones, m.c.m.
G - Helados de cucurucho	Recursión
H - La ardilla viajera	Grafos, conjuntos disjuntos
I - El acertijo del mercader	Vuelta atrás
J - La comida de los pollitos	Matrices, iteración
K - ¡En primera línea de playa!	Algoritmos voraces, ordenación
L - Teclas del piano	Arrays, cadenas

Estadísticas

Problema	# casos de prueba	Espacio en disco
A - La máquina calculadora	28.013	280 KB
B - Palmeras en la nieve	17.391	6 MB
C - Double decker	13.805	168 KB
D - Tiro al patíndromo	38.339	831 KB
E - El conteo de la rosa	110.107	1,9 MB
F - El teorema del punto fijo	26.038	3,1 MB
G - Helados de cucurucho	99	235 KB
H - La ardilla viajera	1.005	2,9 MB
I - El acertijo del mercader	1.817	24 KB
J - La comida de los pollitos	1.485	417 KB
K - ¡En primera línea de playa!	38.119	7 MB
L - Teclas del piano	2.413	4,5 MB

Estadísticas

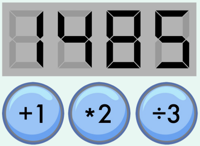
Problema	Primer equipo en resolverlo	Tiempo
A - La máquina calculadora	Sombrero	7
B - Palmeras en la nieve	Sombrero	52
C - Double decker	UAM2	13
D - Tiro al patíndromo	“ OR 1=1 LIMIT 1 –	231
E - El conteo de la rosa	Sombrero	97
F - El teorema del punto fijo	“ OR 1=1 LIMIT 1 –	32
G - Helados de cucurucho	Sombrero	23
H - La ardilla viajera	-	-
I - El acertijo del mercader	-	-
J - La comida de los pollitos	UAM1	161
K - ¡En primera línea de playa!	Sombrero	84
L - Teclas del piano	UAM1	24

Estadísticas

Problema	Envíos	Válidos	% éxito
A - La máquina calculadora	25	5	20 %
B - Palmeras en la nieve	71	14	19 %
C - Double decker	59	36	61 %
D - Tiro al patíndromo	6	2	33 %
E - El conteo de la rosa	33	4	12 %
F - El teorema del punto fijo	34	7	20 %
G - Helados de cucurucho	44	18	40 %
H - La ardilla viajera	3	0	0 %
I - El acertijo del mercader	3	0	0 %
J - La comida de los pollitos	4	3	75 %
K - ¡En primera línea de playa!	11	5	45 %
L - Teclas del piano	41	11	26 %

Estadísticas

	Jueces		Participantes	
Problema	**LOC**	**LOCNC**	**LOC**	**LOCNC**
A - La máquina calculadora	101.66	51.66	69.40	61.00
B - Palmeras en la nieve	109.33	51.83	60.92	48.64
C - Double decker	51.00	24.14	31.19	22.83
D - Tiro al patíndromo	120.28	73.85	89.00	64.00
E - El conteo de la rosa	110.20	58.80	67.00	55.00
F - El teorema del punto fijo	102.00	53.28	68.00	57.71
G - Helados de cucurucho	66.20	35.40	66.83	55.88
H - La ardilla viajera	204.80	112.40	-	-
I - El acertijo del mercader	176.20	57.00	-	-
J - La comida de los pollitos	136.75	96.50	105.66	95.33
K - ¡En primera línea de playa!	81.00	46.75	61.80	47.00
L - Teclas del piano	138.25	81.75	105.18	92.36
- **Total**	1397.67	743,36		

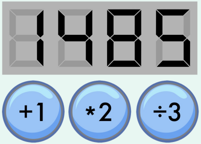


A. La máquina calculadora

Envíos	Válidos	% éxito
25	5	20 %

A. La máquina calculadora

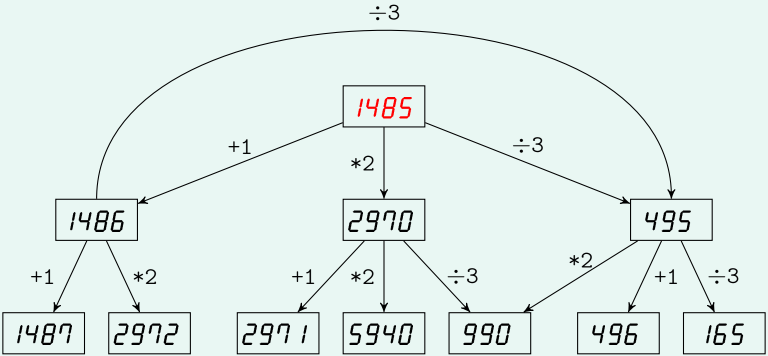
A partir de una configuración de la máquina



queremos llegar a un número deseado con el menor número de pulsaciones.

Podemos modelar el problema con un *grafo (implícito)* donde los vértices son los números y los adyacentes son los números obtenibles con cualquiera de las tres operaciones.

A. La máquina calculadora



Un *recorrido en anchura* del grafo encontrará el camino más corto desde el número origen al número destino.

Llevamos cuenta de la distancia (número de pulsaciones) a cada vértice, la primera vez que se llega a él.

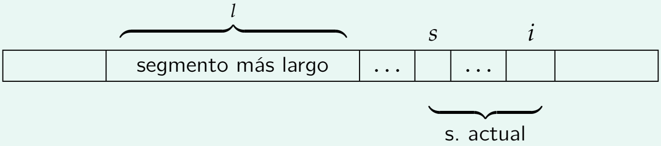
B. Palmeras en la nieve

Envíos	Válidos	% éxito
71	14	19 %

B. Palmeras en la nieve

Es un problema de cálculo del segmento más largo de un vector que cumple una determinada propiedad, en este caso que el número de palmeras que quedan en pie sea menor o igual a 5.
Recorremos el vector en uno de los dos sentidos, por ejemplo de izquierda a derecha, y llevamos guardada:

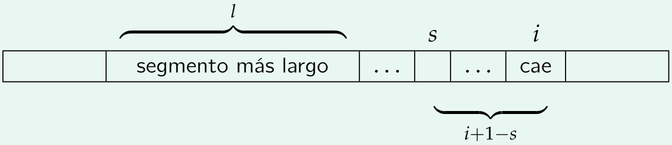
1. La longitud l del segmento más largo hasta el momento.
2. Para el segmento actual, que acaba en la posición i que estamos considerando: el extremo s de comienzo del segmento y el número de palmeras en pie en dicho segmento.



B. Palmeras en la nieve

Pueden suceder tres cosas con la nueva posición a considerar:

- Si la palmera que allí se encuentra cae por el peso de la nieve, la longitud del segmento actual crece en 1 y si dicha longitud es mayor que la mejor hasta el momento se actualiza.



- Si la palmera no cae y aún no se han sobrepasado las 5 palmeras en el segmento, simplemente se incrementa el número de palmeras, y se procede como en el caso anterior.

B. Palmeras en la nieve

- Si la palmera no cae y con esta nueva palmera ya son 6 las palmeras en pie en el segmento, se avanza el extremo izquierdo del segmento hasta rebasar la primera palmera en pie de dicho segmento, de forma que el nuevo segmento a considerar (con 5 palmeras) es el que excluye la que antes era la primera palmera e incluye la nueva palmera.

	s	$\longrightarrow$	$\longrightarrow$	s		i	
...	cae	...cae...	en pie		...	en pie	

$$\underbrace{\qquad\qquad}_{i+1-s}$$

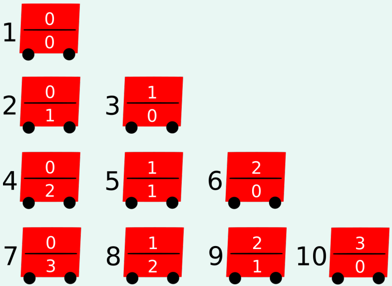
El coste de este algoritmo es lineal con el número de palmeras, ya que cada palmera se considera a lo sumo dos veces: la primera vez para comprobar si cae o no y la segunda en caso de que haya que avanzar el extremo izquierdo del segmento al que pertenece. En este segundo caso, una vez sobrepasado, ya no se vuelve a considerar.

C. Double decker

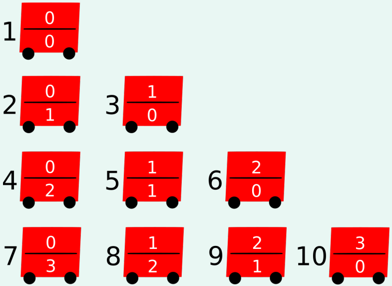
Envíos	Válidos	% éxito
59	36	61 %

C. Double decker

Queremos conocer el *rango* de un autobús conociendo el número de personas en cada piso.



C. Double decker

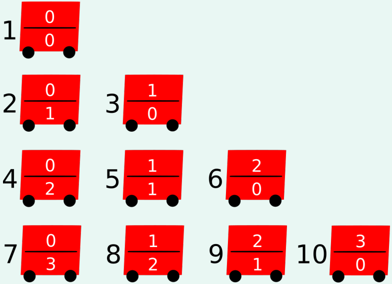


Idea 1

- El número total de personas en el autobús define la fila en la que está el autobús (contando desde la fila 0).

Ejemplo: los autobuses $\frac{1}{2}$ (rango 8) y $\frac{3}{0}$ (rango 10) están en la **fila 3**.

C. Double decker



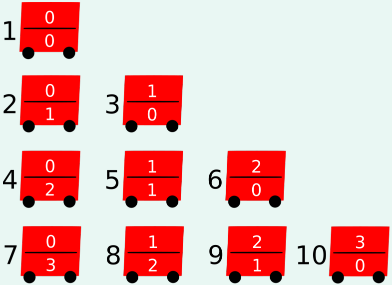
Idea 2

- Para una fila K, el rango de su primer autobús $\frac{0}{K}$ será el **número de autobuses en las filas anteriores <u>más uno</u>**:

$$1 + \sum_{i=0}^{K-1}(i+1) \; = \; 1 + \sum_{i=1}^{K} i \; = \; 1 + \frac{(K+1)K}{2}$$

Rango del 1er autobús de la fila 3: $1 + \sum_{i=1}^{3} i = 1 + \frac{4\times 3}{2} = 1 + 6 = 7$

C. Double decker

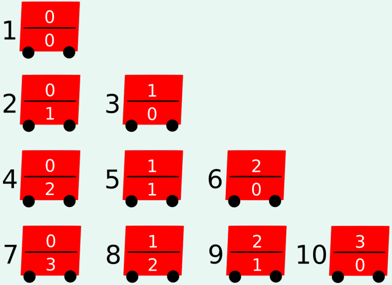


Idea 3

- El número de personas en el piso superior indica el ***desplazamiento* del autobús en su fila**.

Ejemplo: el autobús $\frac{1}{2}$ (rango 8) tiene un desplazamiento de 1, y el autobús $\frac{3}{0}$ (rango 10) un desplazamiento de 3.

C. Double decker

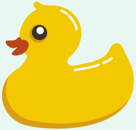


Juntando todo

- El rango del autobús $\frac{N}{M}$ se calcula como:

$$\underbrace{1+\sum_{i=1}^{N+M} i}_{\text{1er autobús en fila } N+M} \quad + \quad \overbrace{N}^{\text{desplazamiento en fila } N+M}$$

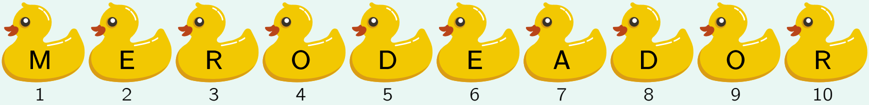
El rango del autobús $\frac{3}{0}$ es $(1+\sum_{i=1}^{3} i)+3=7+3=10$



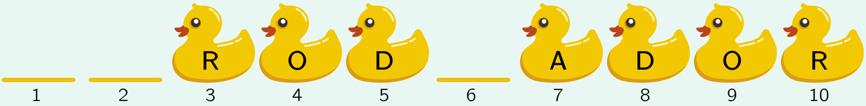
D. Tiro al patíndromo

Envíos	Válidos	% éxito
6	2	33 %

D. Tiro al patíndromo



Conseguir el palíndromo más largo tirando (si es necesario) algunos de los patitos:



D. Tiro al patíndromo

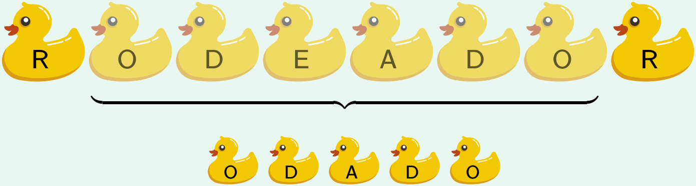
Si los patitos de los extremos coinciden

lo mejor es seleccionarlos y buscar el mejor palíndromo con el resto de patitos:

D. Tiro al patíndromo

Si los patitos de los extremos coinciden

lo mejor es seleccionarlos y buscar el mejor palíndromo con el resto de patitos:

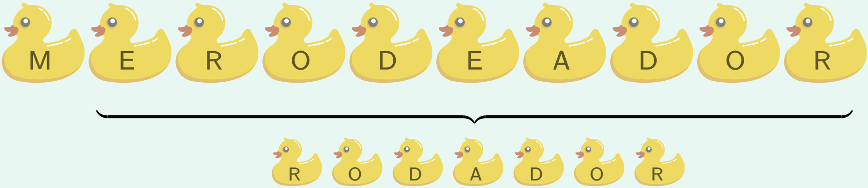


D. Tiro al patíndromo

Si los patitos de los extremos no coinciden, entonces habrá que quitar al menos uno de ellos, buscar recursivamente en cada opción, y quedarse con la mejor:

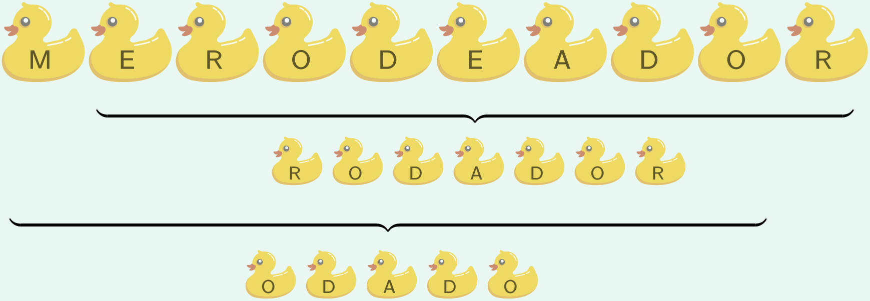
D. Tiro al patíndromo

Si los patitos de los extremos no coinciden, entonces habrá que quitar al menos uno de ellos, buscar recursivamente en cada opción, y quedarse con la mejor:



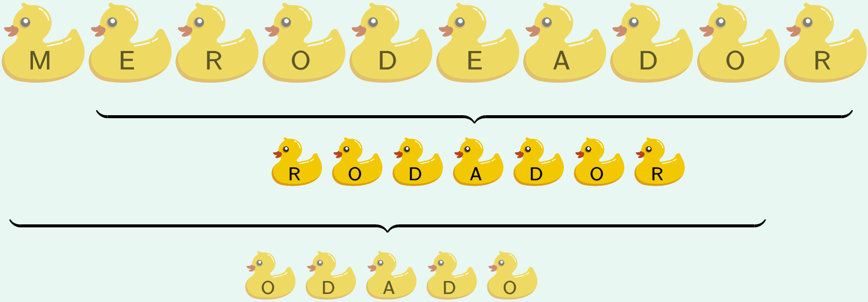
D. Tiro al patíndromo

Si los patitos de los extremos no coinciden, entonces habrá que quitar al menos uno de ellos, buscar recursivamente en cada opción, y quedarse con la mejor:



D. Tiro al patíndromo

Si los patitos de los extremos no coinciden, entonces habrá que quitar al menos uno de ellos, buscar recursivamente en cada opción, y quedarse con la mejor:



D. Tiro al patíndromo

Definición recursiva ($i < j$):

$$patindromo(i,j) = \begin{cases} patindromo(i+1, j-1) + 2 \\ \qquad \text{si } patito[i] = patito[j] \\ \\ \max\{patindromo(i+1, j), patindromo(i, j-1)\} \\ \qquad \text{si } patito[i] \neq patito[j] \end{cases}$$

Casos básicos:

$$\begin{aligned} patindromo(i,i) &= 1 \\ patindromo(i,j) &= 0 \quad \text{si } i > j \end{aligned}$$

Calcular los valores de esta función utilizando *programación dinámica*. Utilizar la tabla rellena para reconstruir la solución.

E. El conteo de la rosa

Envíos	Válidos	% éxito
33	4	12 %

E. El conteo de la rosa

Tenemos que numerar, entre dos personas, las páginas de un libro desde la a hasta la b.

Queremos hacer un reparto justo en *número de dígitos* a escribir.

E. El conteo de la rosa

Tenemos que numerar, entre dos personas, las páginas de un libro desde la a hasta la b.

Queremos hacer un reparto justo en *número de dígitos* a escribir.

Necesitamos:

1. Averiguar la cantidad de dígitos que contienen todos los números entre a y b

E. El conteo de la rosa

Tenemos que numerar, entre dos personas, las páginas de un libro desde la a hasta la b.

Queremos hacer un reparto justo en *número de dígitos* a escribir.

Necesitamos:

1. Averiguar la cantidad de dígitos que contienen todos los números entre a y b
2. Encontrar el punto medio

E. El conteo de la rosa

Cantidad de dígitos que contienen todos los números entre a y b

E. El conteo de la rosa

Cantidad de dígitos que contienen todos los números entre a y b

Contar cuántos dígitos tiene un número es fácil (divisiones sucesivas, secuencia de `if`'s o logaritmo en base 10).

Se puede contar la cantidad de dígitos de un rango de números con un bucle: $\mathcal{O}(b-a)$

E. El conteo de la rosa

Cantidad de dígitos que contienen todos los números entre a y b

Contar cuántos dígitos tiene un número es fácil (divisiones sucesivas, secuencia de `if`'s o logaritmo en base 10).

Se puede contar la cantidad de dígitos de un rango de números con un bucle: $\mathcal{O}(b-a)$

¡Lento!

No se aprovecha la coherencia: todos los números entre 100.000 y 999.999 tienen el mismo número de dígitos.

Se necesita una implementación más eficiente basada en el número de dígitos de los extremos: $\mathcal{O}(n \log_{10} n)$

E. El conteo de la rosa

Cantidad de dígitos que contienen todos los números entre a y b

Aprovechar la "*coherencia espacial*" es tedioso en rangos arbitrarios. Es mucho más fácil si el rango empieza en 1.
Averiguar la cantidad en cualquier otro rango es cuestión de restar:

	1		a		b
Cantidad dígitos $[1, i]$		$\dots$		$\dots$	

E. El conteo de la rosa

Cantidad de dígitos que contienen todos los números entre a y b

Aprovechar la "*coherencia espacial*" es tedioso en rangos arbitrarios. Es mucho más fácil si el rango empieza en 1.
Averiguar la cantidad en cualquier otro rango es cuestión de restar:

	1		a		b
Cantidad dígitos [1, i]		...		...	

$$[a, b] = [1, b] - [1, a - 1]$$

Esto es *más ineficiente* que calcular el rango de una vez, pero sólo por una constante multiplicativa.

E. El conteo de la rosa

Encontrar el punto medio

Una vez que sabemos la cantidad de dígitos, calculamos cuántos tienen que escribir cada monje, y *buscamos* el número de página i que lo consigue. Podemos ir *buscando* la posición, calculando la cantidad de dígitos para el rango $[a, i]$: $\mathcal{O}(b - a)$

E. El conteo de la rosa

Encontrar el punto medio

Una vez que sabemos la cantidad de dígitos, calculamos cuántos tienen que escribir cada monje, y *buscamos* el número de página i que lo consigue. Podemos ir *buscando* la posición, calculando la cantidad de dígitos para el rango $[a, i]$: $\mathcal{O}(b - a)$

¡Lento!

El rango es demasiado grande para hacer una búsqueda lineal.

E. El conteo de la rosa

Encontrar el punto medio

El número de páginas que tiene que escribir el primer monje *es creciente* con el valor de i.

i	97	98	99	100	101	102	103
Cantidad dígitos [97, i]	2	4	6	9	12	15	...

E. El conteo de la rosa

Encontrar el punto medio

El número de páginas que tiene que escribir el primer monje *es creciente* con el valor de i.

i	97	98	99	100	101	102	103
Cantidad dígitos $[97, i]$	2	4	6	9	12	15	...

¡Solución!

Búsqueda binaria sobre una secuencia implícita.
Cuidando si el número buscado no "está"

E. El conteo de la rosa

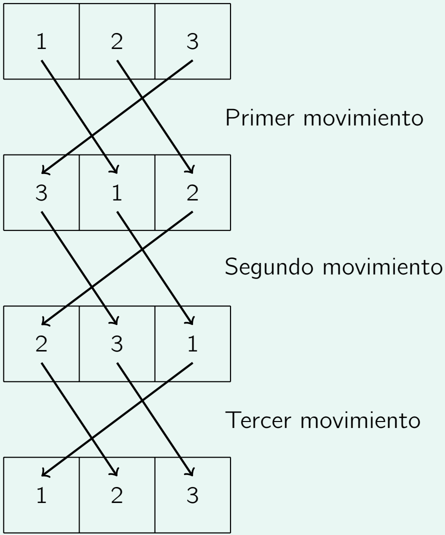
Como participante, este problema *se comprueba solo*. Tras calcular la solución, puedes comprobar si es la posición óptima.

¡¡No debería haber habido WRONG ANSWER!!

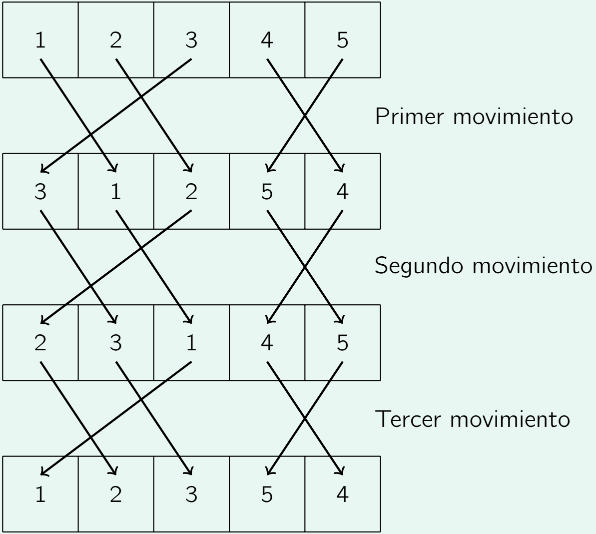
F. El teorema del punto fijo

Envíos	Válidos	% éxito
34	7	20 %

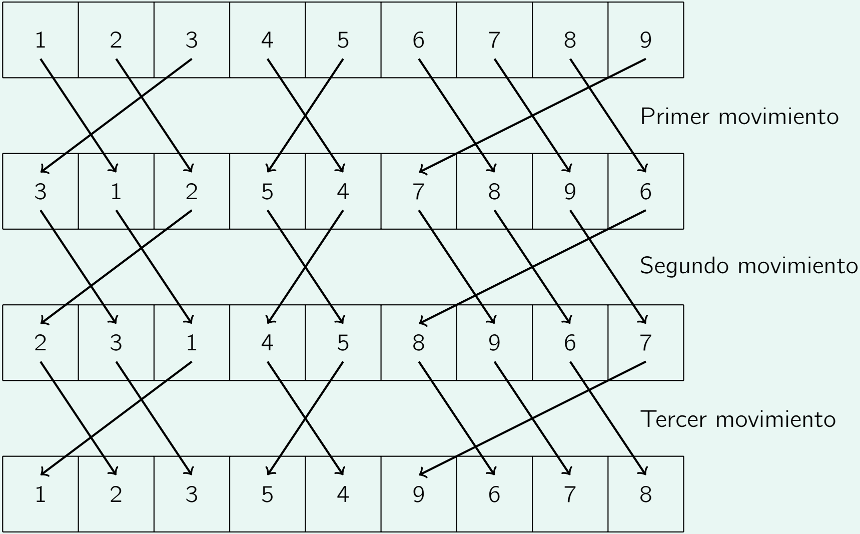
F. El teorema del punto fijo



F. El teorema del punto fijo



F. El teorema del punto fijo



Solución: Sacar la longitud de los ciclos de la permutación.

Solución: Sacar la longitud de los ciclos de la permutación.

Calcular el mínimo común múltiplo de todas ellas.

Solución: Sacar la longitud de los ciclos de la permutación.

Calcular el mínimo común múltiplo de todas ellas.

Para hacerlo, se puede calcular con el MCD

Solución: Sacar la longitud de los ciclos de la permutación.

Calcular el mínimo común múltiplo de todas ellas.

Para hacerlo, se puede calcular con el MCD

Ojo: Cuidado con los desbordamientos.

G. Helados de cucurucho

Envíos	Válidos	% éxito
44	18	40 %

G. Helados de cucurucho

Si tenemos i bolas de helado de vainilla, y j de chocolate, ¿de qué formas se pueden colocar para formar un helado de cucurucho?

G. Helados de cucurucho

Si tenemos i bolas de helado de vainilla, y j de chocolate, ¿de qué formas se pueden colocar para formar un helado de cucurucho?

Recursión

Partiendo de un helado *parcialmente construído*:

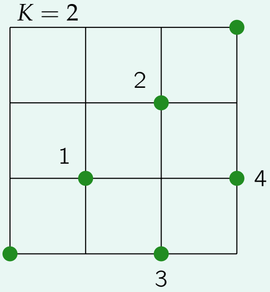
- Si hemos puesto ya todas las bolas de helado, escribimos la configuración.
- Si aún nos quedan bolas de vainilla, añadimos una y probamos.
- Si aún nos quedan bolas de chocolate, añadimos una y probamos.

Quizá lo más difícil fuera la gestión del último espacio. Si se añadía, PRESENTATION ERROR.

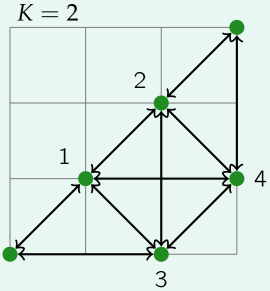
H. La ardilla viajera

Envíos	Válidos	% éxito
3	0	0 %

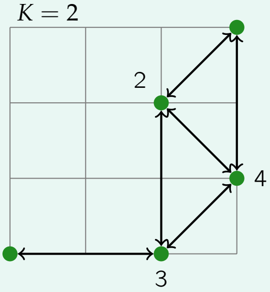
H. La ardilla viajera



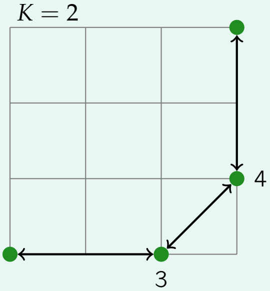
H. La ardilla viajera



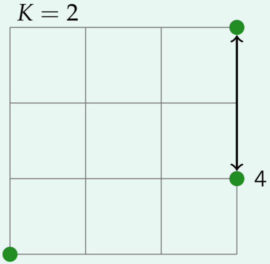
H. La ardilla viajera



H. La ardilla viajera

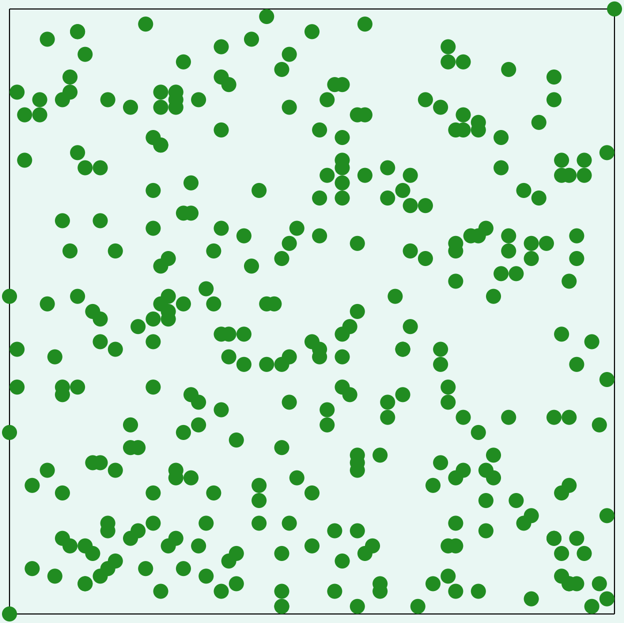


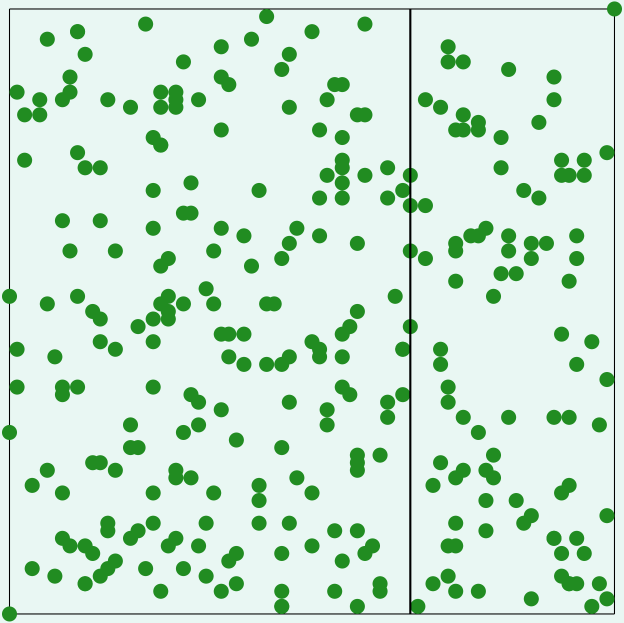
H. La ardilla viajera

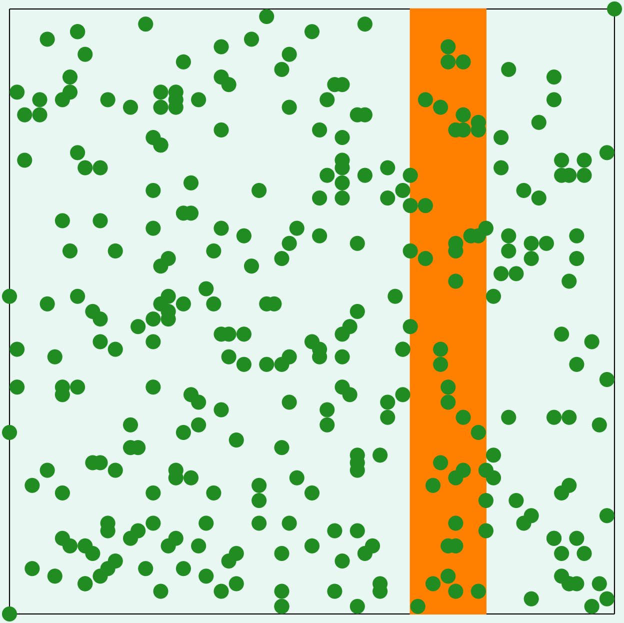


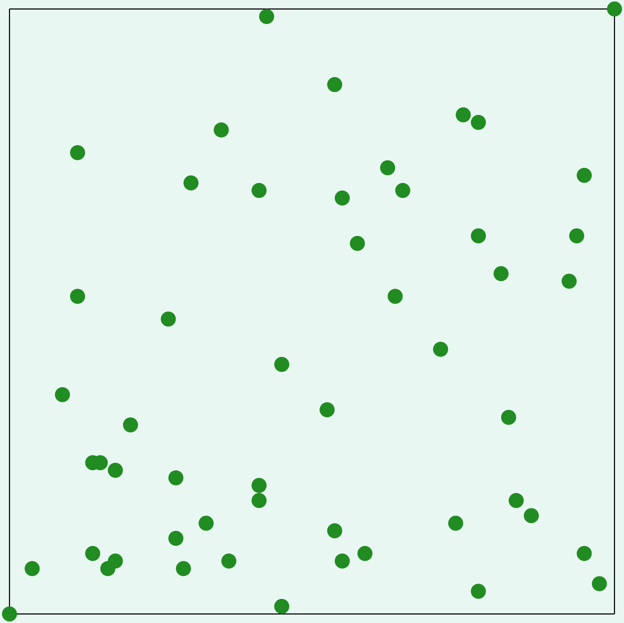
La solución al problema requiere dos fases:

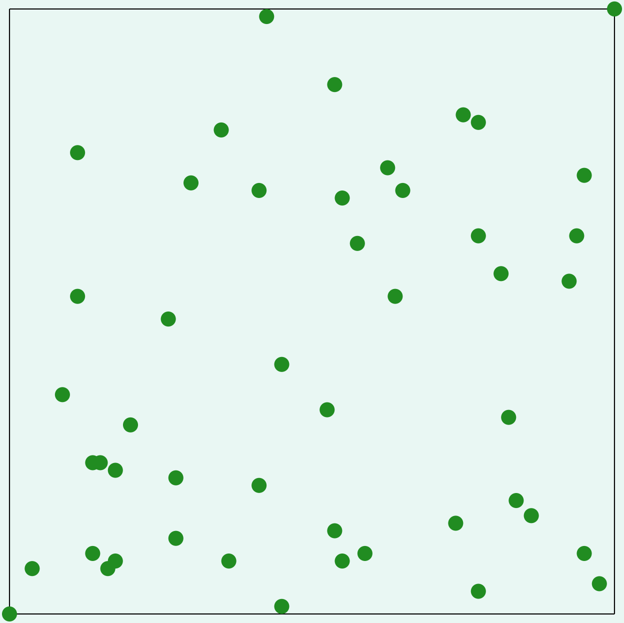
1. Construir el grafo
   Una arista entre cada par de vértices cuya distancia sea $\leq K$.
   La solución evidente es $\mathcal{O}(n^2)$. Con 10.000 vértices, TLE
2. Utilizar el grafo para calcular la respuesta.
   La forma evidente de hacerlo es utilizar BFS ignorando aristas. En el peor de los casos, 10.000 ejecuciones de BFS, TLE.

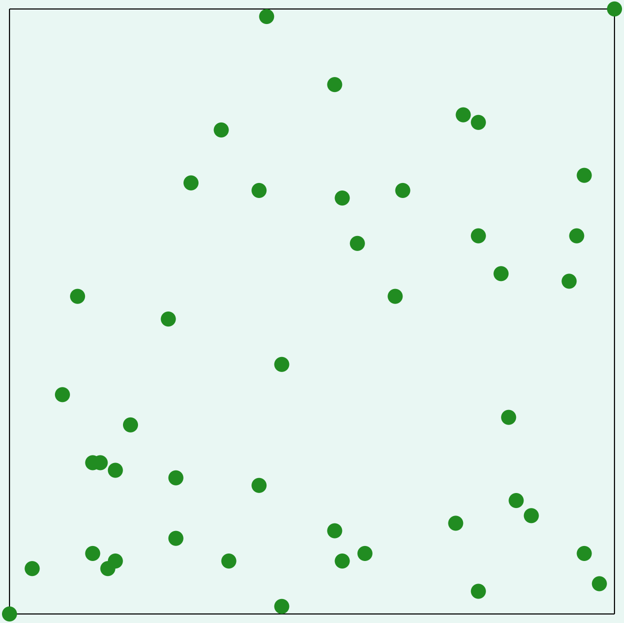


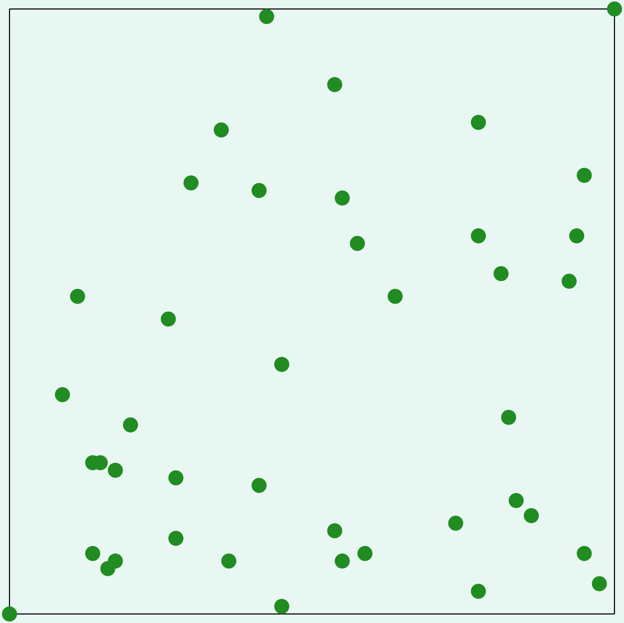


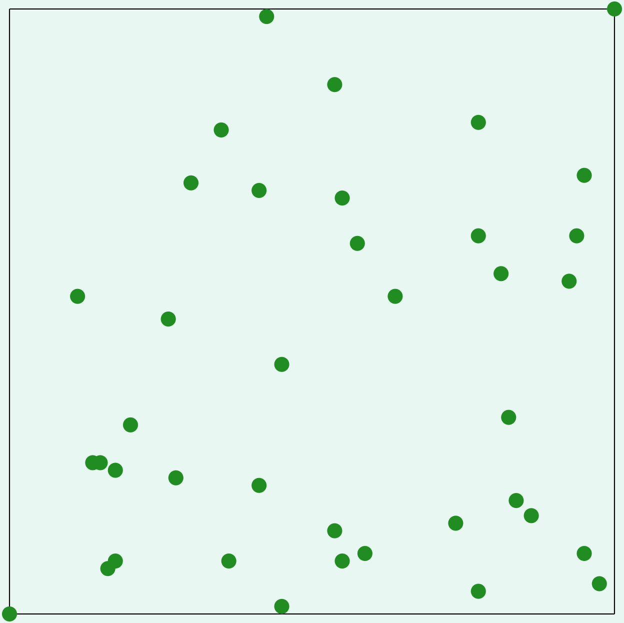


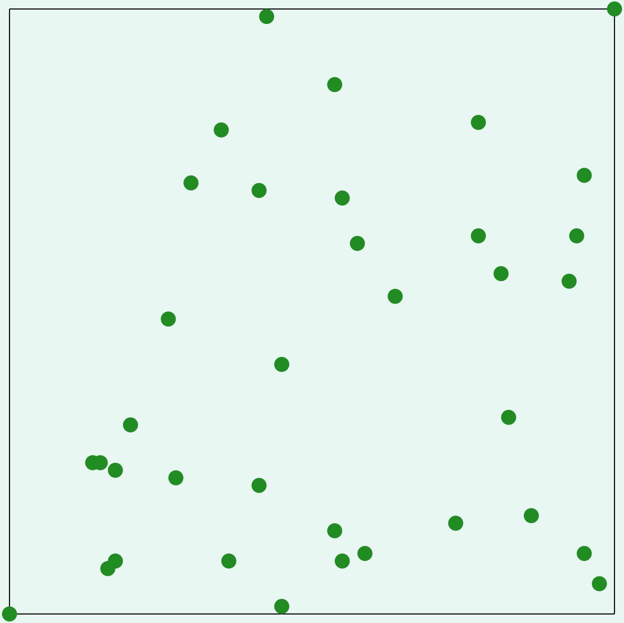


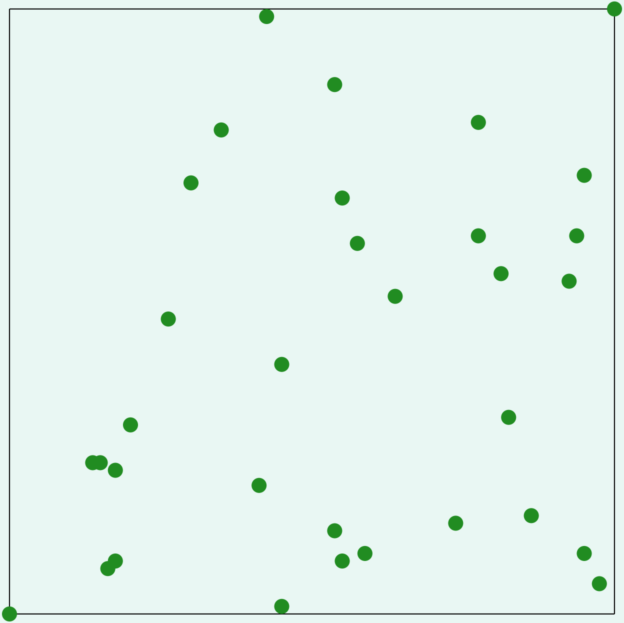


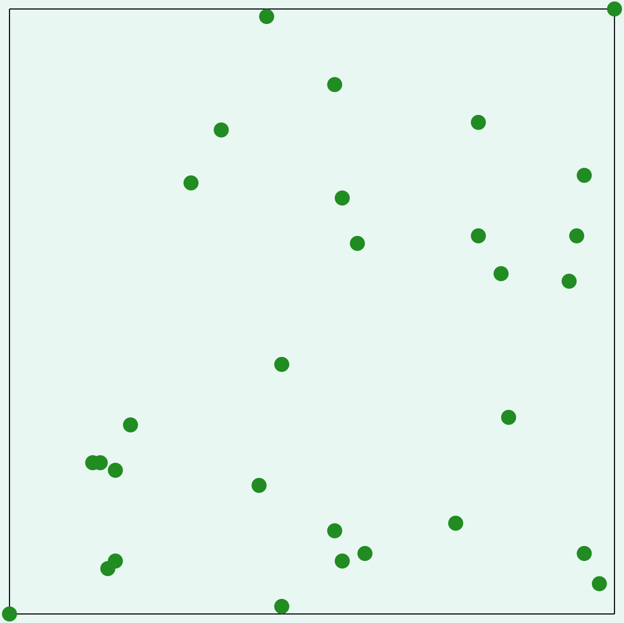


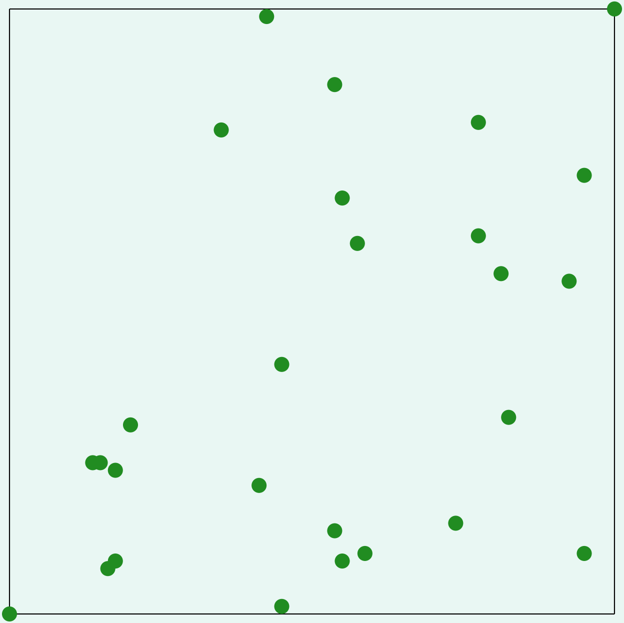


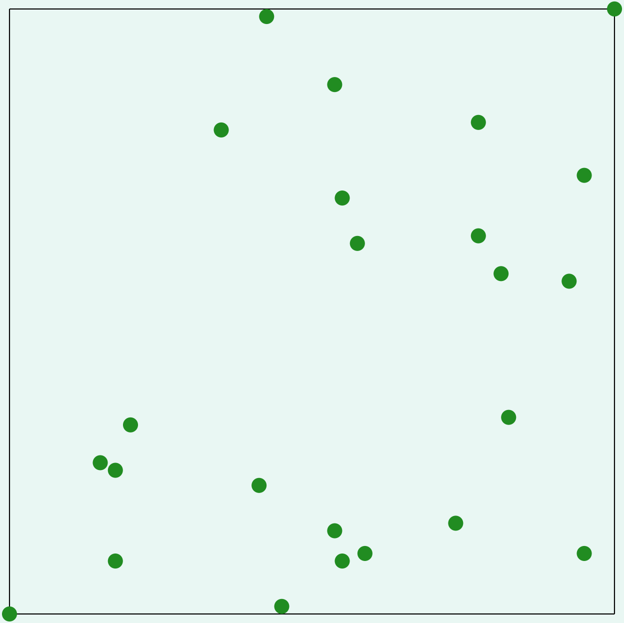


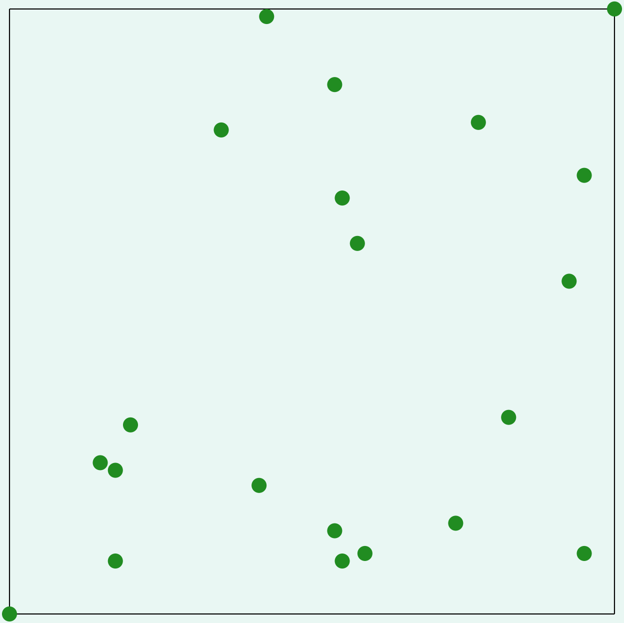


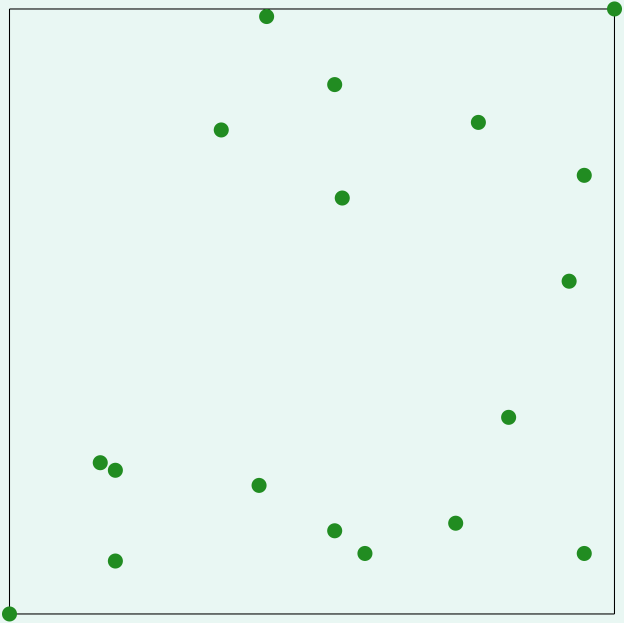


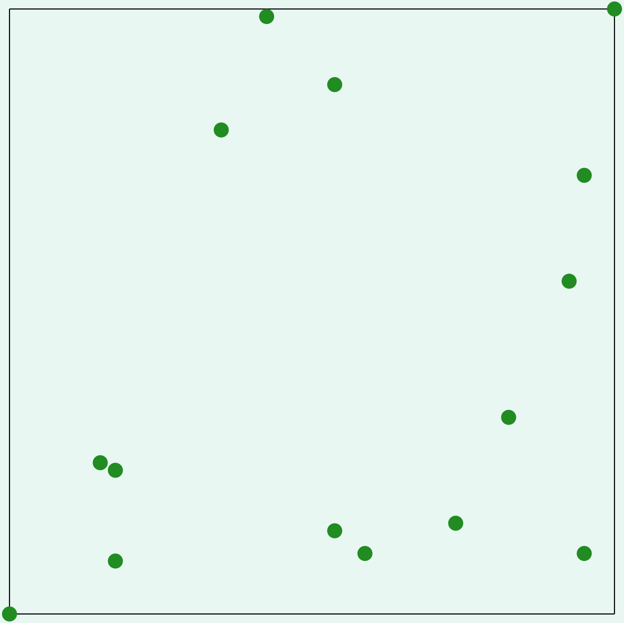


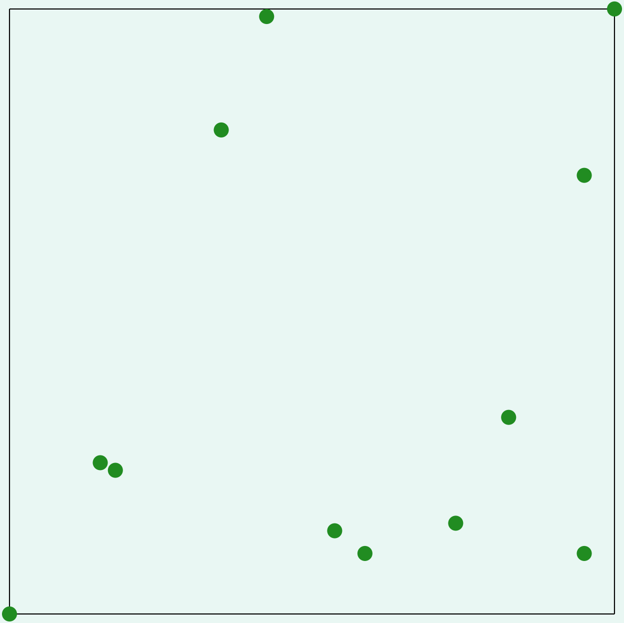


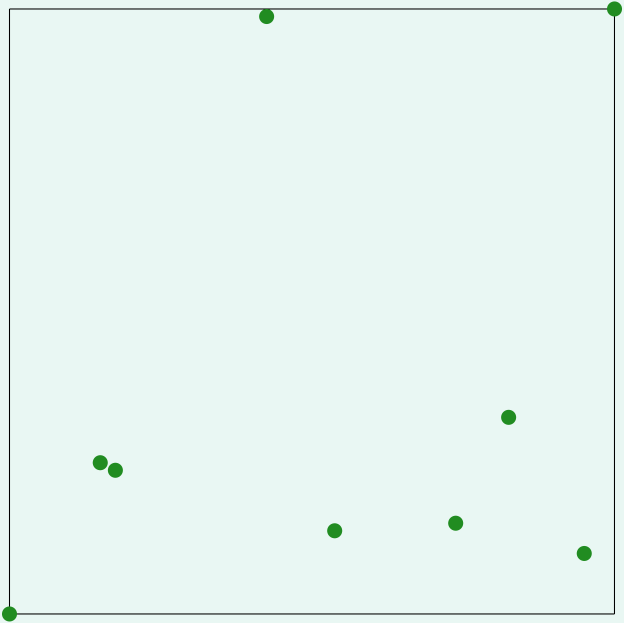


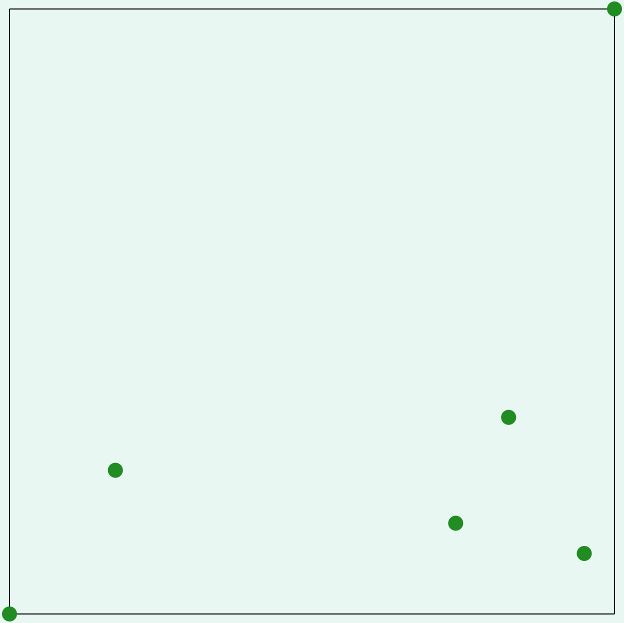


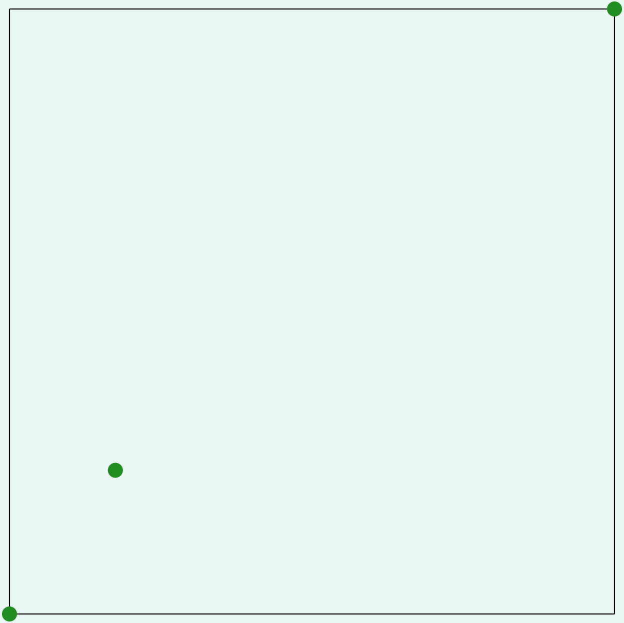












Opción 1

El BFS es la forma de calcular la función:

$$f(x) = \text{puede saltar el día } x$$

Objetivo: Encontrar el primer n en el que $f(n) == \mathit{false}$

Solución: Búsqueda binaria sobre n

Opción 2

Recorrer de derecha a izquierda utilizando UFDS

I. El acertijo del mercader

Envíos	Válidos	% éxito
3	0	0 %

I. El acertijo del mercader

¿Cuál es el menor número de peregrinos que pueden caminar en exactamente d formaciones diferentes?

I. El acertijo del mercader

¿Cuál es el menor número de peregrinos que pueden caminar en exactamente d formaciones diferentes?

¿Cuál es el menor número n que tiene exactamente d divisores?

I. El acertijo del mercader

¿Cuál es el menor número de peregrinos que pueden caminar en exactamente d formaciones diferentes?

¿Cuál es el menor número n que tiene exactamente d divisores?

Consideraciones:

1. $1 \leq n \leq 10^9$
2. Sacar el número de divisores de un número i es $\mathcal{O}(sqrt(i))$

I. El acertijo del mercader

¿Cuál es el menor número de peregrinos que pueden caminar en exactamente d formaciones diferentes?

¿Cuál es el menor número n que tiene exactamente d divisores?

Consideraciones:

1. $1 \leq n \leq 10^9$
2. Sacar el número de divisores de un número i es $\mathcal{O}(sqrt(i))$

Una búsqueda lineal con i desde 1 hacia arriba es demasiado lenta.

I. El acertijo del mercader

¿Cuál es el menor número de peregrinos que pueden caminar en exactamente d formaciones diferentes?

¿Cuál es el menor número n que tiene exactamente d divisores?

Consideraciones:

1. $1 \leq n \leq 10^9$
2. Sacar el número de divisores de un número i es $\mathcal{O}(sqrt(i))$

Una búsqueda lineal con i desde 1 hacia arriba es demasiado lenta.

¿Búsqueda binaria?

No: el número de divisores *no es* una función monótona.

I. El acertijo del mercader

¿Cuántos divisores tiene un número n?

I. El acertijo del mercader

¿Cuántos divisores tiene un número n?

Si la descomposición en factores primos de n es

$$n = A^a \cdot B^b \cdot \ldots \cdot K^k$$

el número de divisores es:

$$d = (a+1) \cdot (b+1) \cdot \ldots \cdot (k+1)$$

I. El acertijo del mercader

Nos piden *lo contrario*. Dado un d (caso de prueba):

$$d = (a+1) \cdot (b+1) \cdot \ldots \cdot (k+1)$$

encontrar el menor n tal que su desomposición sea:

$$n = A^a \cdot B^b \cdot \ldots \cdot K^k$$

I. El acertijo del mercader

Nos piden *lo contrario*. Dado un d (caso de prueba):

$$d = (a+1) \cdot (b+1) \cdot \ldots \cdot (k+1)$$

encontrar el menor n tal que su desomposición sea:

$$n = A^a \cdot B^b \cdot \ldots \cdot K^k$$

Necesitamos sacar los divisores de d y **construir** el mejor n.

Ideas:

- Usar los menores primos: $A = 2$, $B = 3$, $C = 5$, ...
- Usar los divisores mayores de d en los primos más pequeños.

I. El acertijo del mercader

¡CUIDADO! *No* sirve un algoritmo voraz.

De hecho, los divisores de d *no* tienen por qué ser primos.

Caso de prueba de ejemplo:

$$64 = 2^6 = 2^2 \cdot 2^2 \cdot 2 \cdot 2 = (3+1) \cdot (3+1) \cdot (1+1) \cdot (1+1)$$

$$7560 = 2^3 \cdot 3^3 \cdot 5^1 \cdot 7^1$$

I. El acertijo del mercader

$$30.240 = 2^5 \cdot 3^3 \cdot 5^1 \cdot 7^1$$

I. El acertijo del mercader

$$30.240 = 2^5 \cdot 3^3 \cdot 5^1 \cdot 7^1$$

$$\Downarrow$$

$$(5+1) \cdot (3+1) \cdot (1+1) \cdot (1+1)$$

$$\|$$

$$6 \cdot 4 \cdot 2 \cdot 2 = 2^6 \cdot 3 = \mathbf{96}\ \textit{divisores}$$

I. El acertijo del mercader

$$30.240 = 2^5 \cdot 3^3 \cdot 5^1 \cdot 7^1$$

$$\Downarrow$$

$$(5+1) \cdot (3+1) \cdot (1+1) \cdot (1+1)$$

$$\|$$

$$6 \cdot 4 \cdot 2 \cdot 2 = 2^6 \cdot 3 = \mathbf{96} \ \textit{divisores}$$

$$27.720 = 2^3 \cdot 3^2 \cdot 5^1 \cdot 7^1 \cdot 11^1$$

I. El acertijo del mercader

$$30.240 = 2^5 \cdot 3^3 \cdot 5^1 \cdot 7^1$$

$$\Downarrow$$

$$(5+1) \cdot (3+1) \cdot (1+1) \cdot (1+1)$$

$$\|$$

$$6 \cdot 4 \cdot 2 \cdot 2 = 2^6 \cdot 3 = \mathbf{96}\ \textit{divisores}$$

$$27.720 = 2^3 \cdot 3^2 \cdot 5^1 \cdot 7^1 \cdot 11^1$$

$$\Downarrow$$

$$(3+1) \cdot (2+1) \cdot (1+1) \cdot (1+1) \cdot (1+1)$$

$$\|$$

$$6 \cdot 4 \cdot 2 \cdot 2 = 2^6 \cdot 3 = \mathbf{96}\ \textit{divisores}$$

I. El acertijo del mercader

Hay que buscar todas las formas de reescribir d como secuencia de multiplicaciones, y, para cada una, construir cada n:

Reescritura	Salida
64	$2^{63} = \infty$
$32 \cdot 2$	$2^{31} \cdot 3^{1} = \infty$
$16 \cdot 4$	$2^{15} \cdot 3^{3} = 294.912$
	$\ldots$
$4 \cdot 4 \cdot 2 \cdot 2$	$2^{3} \cdot 3^{3} \cdot 5^{1} \cdot 7^{1} = 7560$
	$\ldots$

I. El acertijo del mercader

Cosas importantes:

1. Hay *muchas* formas de reescribir los números altos. Usamos **vuelta atrás** para cancelar las reescrituras que no vayan a mejorar el mejor n construído.
2. Cuando se calcule K^k hay que vigilar que no se supera el límite y anular la vuelta atrás si lo hace.

I. El acertijo del mercader

Solución alternativa

Todos los números de la salida tienen la misma forma:

$$2^a \cdot 3^b \cdot 5^c \cdot \ldots$$

con $a \geq b \geq c \ldots \geq 0$

I. El acertijo del mercader

Solución alternativa

Todos los números de la salida tienen la misma forma:

$$2^a \cdot 3^b \cdot 5^c \cdot \ldots$$

con $a \geq b \geq c \ldots \geq 0$

Idea:

- Generar todos los posibles que no desborden
- Para cada uno calcular su número de divisores, $a+1 \cdot b+1 \cdot \ldots$
- Guardar en una tabla hash `respuestas[numDivisores]=menorGenerado`
- Contestar cada caso mirando en la tabla hash

I. El acertijo del mercader

Solución alternativa

Todos los números de la salida tienen la misma forma:

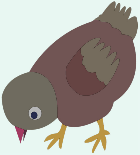
$$2^a \cdot 3^b \cdot 5^c \cdot \ldots$$

con $a \geq b \geq c \ldots \geq 0$

Idea:

- Generar todos los posibles que no desborden
- Para cada uno calcular su número de divisores, $a + 1 \cdot b + 1 \cdot \ldots$
- Guardar en una tabla hash `respuestas[numDivisores]=menorGenerado`
- Contestar cada caso mirando en la tabla hash

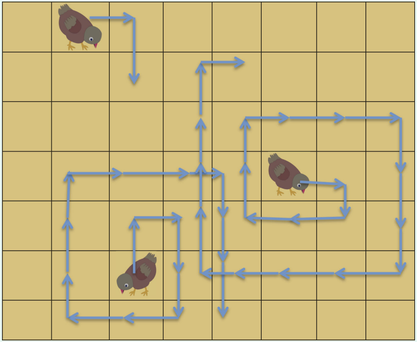
¡Sólo hay 266 entradas que *no* dan `+INF`!



J. La comida de los pollitos

Envíos	Válidos	% éxito
4	3	75 %

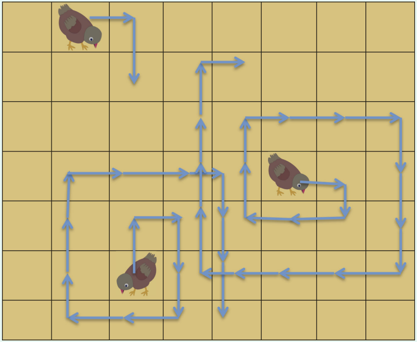
J. La comida de los pollitos



Simular el movimiento de cada pollito de forma independiente y acumular en una matriz los granos necesarios para cada uno.

J. La comida de los pollitos

Vamos haciendo avanzar al pollito en cada una de las cuatro direcciones e incrementamos el número de pasos a dar cada dos tramos recorridos.

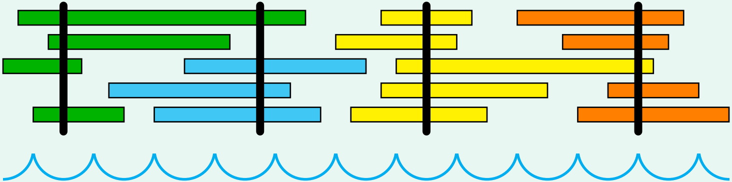


- K. ¡En primera línea de playa!

Envíos	Válidos	% éxito
11	5	45 %

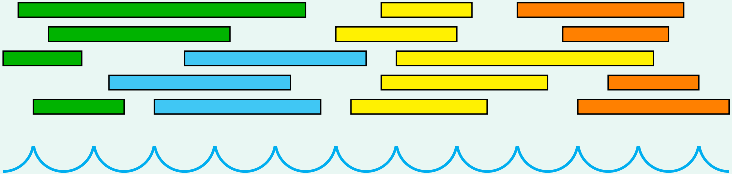
K. ¡En primera línea de playa!

Queremos saber cuántos túneles como *mínimo* serían necesarios para atravesar todos los edificios.



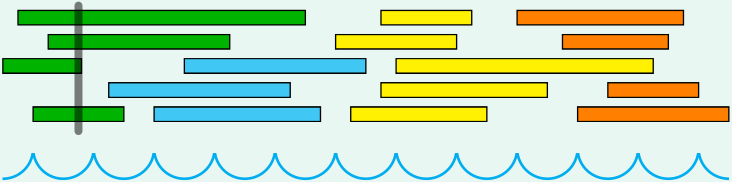
Descripción abstracta: Dada una serie de intervalos, calcular el menor conjunto de puntos P tal que todo intervalo tenga al menos un punto en P.

K. ¡En primera línea de playa!



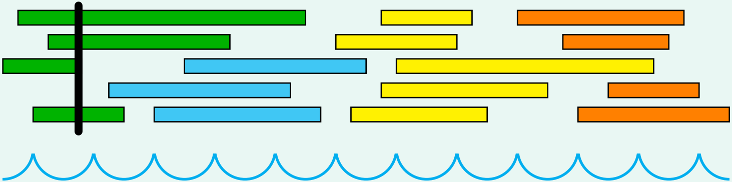
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



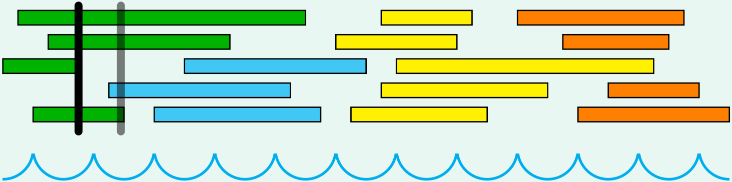
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



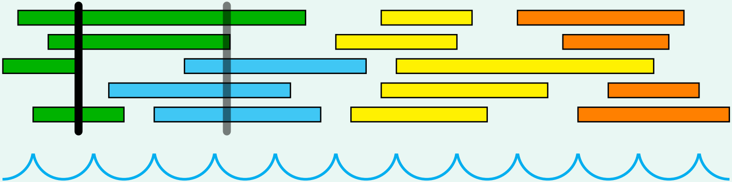
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



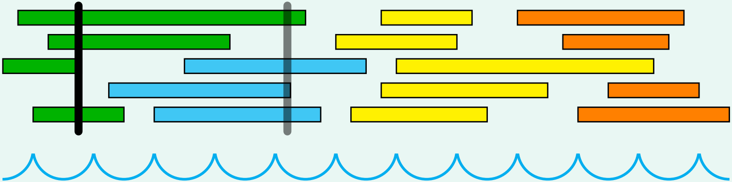
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



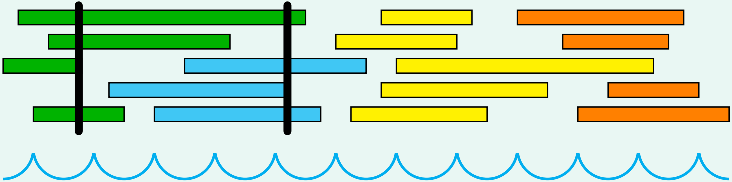
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



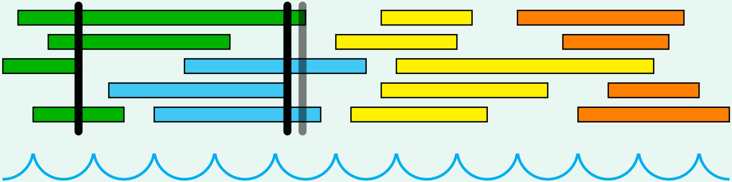
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



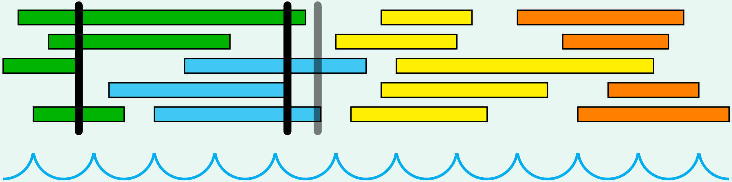
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



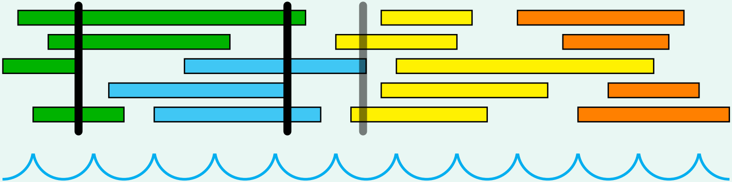
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



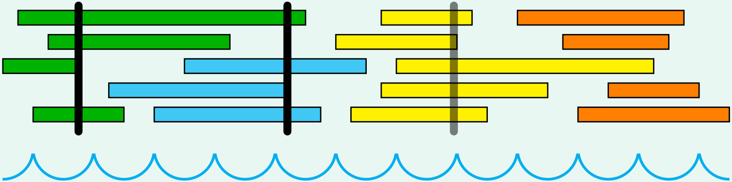
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



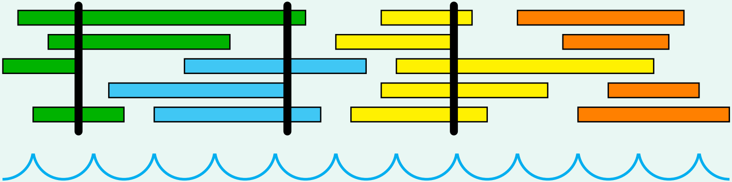
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



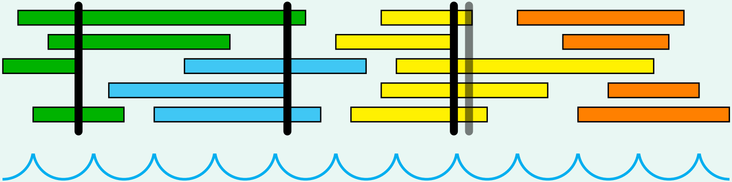
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



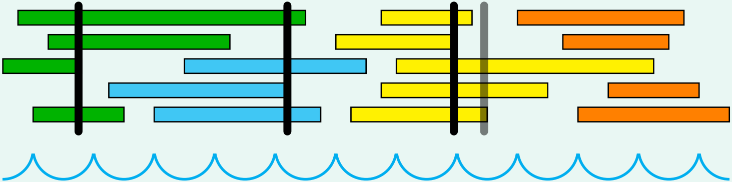
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



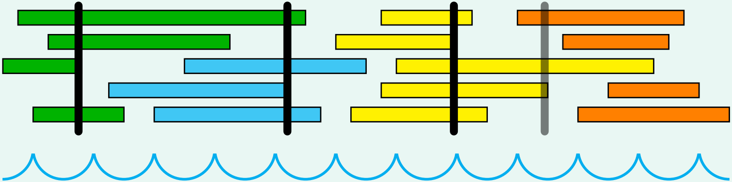
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



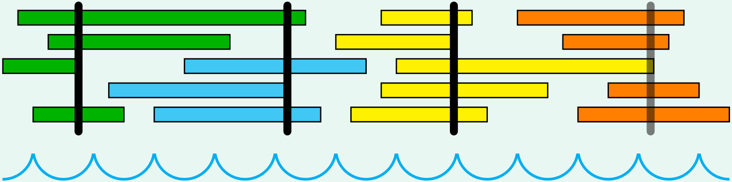
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



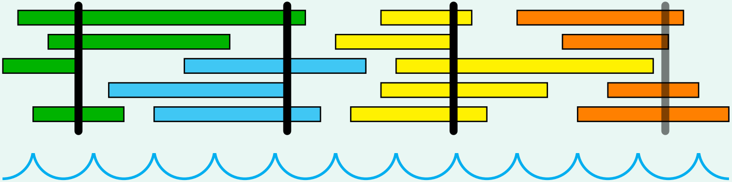
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



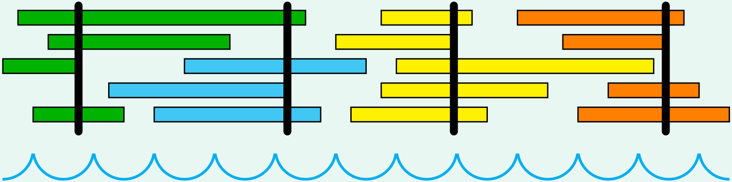
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



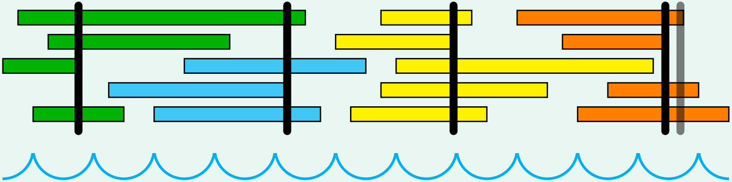
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



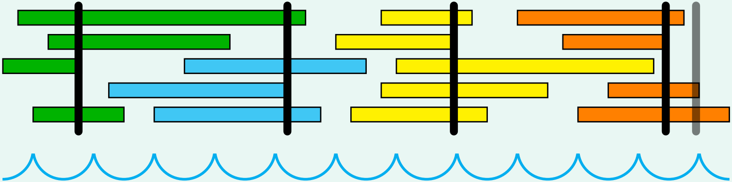
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



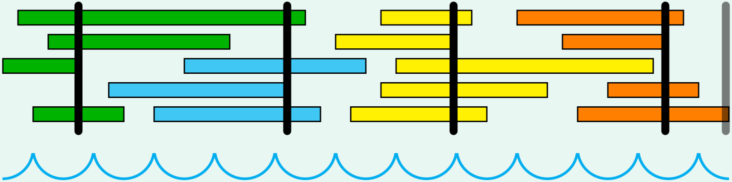
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



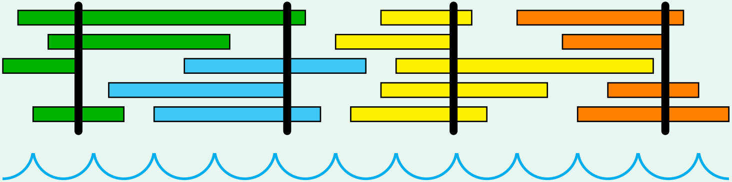
Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

K. ¡En primera línea de playa!



Estrategia *voraz*: Considerar los edificios de menor a mayor extremo oriental y para cada edificio sin túnel, colocar un túnel en ese extremo.

- L. Teclas del piano

Envíos	Válidos	% éxito
41	11	26 %

L. Teclas del piano