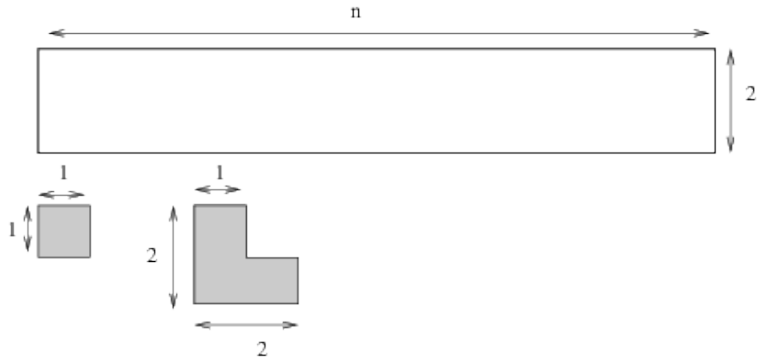
Tarea I

Para entregar el jueves 23 de agosto al inicio de la clase.

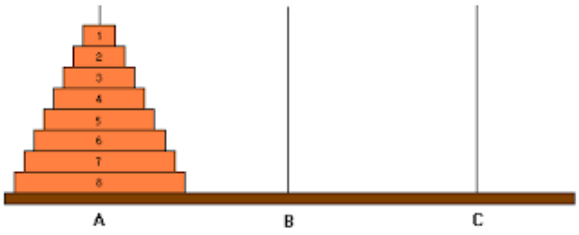
1. Demuestra que el algoritmo en la página 26 de las notas de la primera clase efectivamente calcula a^n.

2. Se tiene una caja de tamaño $2 \times n$ y una cantidad ilimitada de dos tipos de piezas como se ve en la siguiente figura:



   Define a_n como el número de diferentes maneras para llenar completamente la caja con estas piezas. Busca una recursión para a_n y resuélvela.

3. Tomamos el problema de los torres de Hanoi 2.0 que consiste en mover n discos de A a C. Solamente se puede mover un disco a la vez y nunca se puede poner un disco sobre uno más chico. No está permitido mover un disco entre A y C (solamente entre A y B, y entre B y C)



Escribe un algoritmo eficiente en pseudo código para resolverlo y calcula su complejidad.

4. Consideramos una cadena de longitud n donde en cada posición aparece 0, 1, 2 o 3. Sea a_n el número de diferentes cadenas que se pueden formar con un número par de 0's.

   Verifica que

   $$a_{n+1} = 2a_n + 4^n.$$

   Resuelvela.

5. Ecuaciones de diferencias pueden ser usadas para calcular aproximaciones a números irracionales.
   Ejemplo: supongamos que queremos calcular una aproximación a $\sqrt{7}$. Para eso, generamos la siguiente secuencia (de números naturales):

   $$a_n = 4a_{n-1} + 3a_{n-2} \quad a_0 = 0,\ a_1 = 1$$

   Usaremos $\frac{a_{n+1}}{a_n} - 2$ como aproximación a $\sqrt{7}$ para n suficientemente grande.

   Encuentra una expresión para a_n. Verifica que efectivamente $\frac{a_{n+1}}{a_n} - 2 \to \sqrt{7}$ si $n \to \infty$

6. (no entregar)

   $$a_{n+3} + 6a_{n+2} + 12a_{n+1} + 8a_n = 4, \quad a_0 = 1, a_1 = -2, a_2 = 8$$

7. Para cada una de las siguientes afirmaciones, verifica si siempre se cumple o no. Motiva tu respuesta.

   (a) Si $f(n) \in O(n^2)$ y $g(n) \in O(n)$, entonces $f(n)/g(n) \in O(n)$.

   (b) $2^{2n+3} \in O(2^n)$

   (c) $nlog(n) \in O(log(n)^{log(n)})$

8. (no entregar) Un programa de algebra matricial contiene el siguiente código:

```
read(n);
for i:= 1 to n do
   for j:= 1 to i do
      for k:= j to i+j do
             f(i,j,k)
```

Sea $T(n)$ el número de llamadas a $f(\cdots)$ como función de n. Calcula una expresión explícita para $T(n)$.

9. (no entregar) Decimos que la función $f(n)$ es **menor** que $g(n)$ si f es $O(g(n))$. Ordena las siguientes funciones (de menor a mayor): $nlog(n), (log(n))^2, 5n^2 + 7n, n^{5/2}, n!, 4^n, n^n + log(n), 5^{log(n)}, log(n!)$, $\sqrt{n}, \exp(n), 10^n + n^{20}$.

10. (no entregar) Si f_1 es $O(h_1)$ y f_2 es $O(h_2)$, demuestra que $(f_1 + f_2)^2$ es $O(h_1^2 + h_2^2)$.

11. (bonus) Supongamos que se tenga un pastel en forma de un polígono convexo de n lados. Se decide cortar el pastel por los diagonales del polígono. Llama a_n el máximo número de rebanadas que se pueda obtener de esta manera. Muestra que a_n cumple con:

$$a_{n+1} = a_n + (n-1) + \left(\begin{array}{c} n \\ 3 \end{array} \right)$$