1. De las siguientes afirmaciones, indicar cuales son ciertas y cuales no:

$$\boxed{f(n) \in O(g(n)) \overset{(1)}{\Longleftrightarrow} \lim_{n\to+\infty} \frac{f(n)}{g(n)} = 0 \overset{(2)}{\Longleftrightarrow} g(n) \in \Omega(f(n)))}$$

1. Es la resolución de "BigO Notation", 2. Cálculo de "Omega Notation".

(a) $n^2 \in (O \wedge \Omega)(n^3)$

1. $\lim_{n\to+\infty} \frac{n^2}{n^3} = \lim_{n\to+\infty} \frac{1}{n} = 0$
2. Este no puede ser, pues ya es O(n)

(b) $n^3 \in (O \wedge \Omega)(n^2)$

1. $\lim_{n\to+\infty} \frac{n^3}{n^2} = \lim_{n\to+\infty} n = +\infty$
2. Por la definición (2) y (a.1), podemos asegurar que $n^3 \in \Omega(n^2)$.

(c) $2^{n+1} \in (O \wedge \Omega)(2^n)$

1. $\lim_{n\to+\infty} \frac{2^{n+1}}{2^n} = \lim_{n\to+\infty} \frac{2^n \cdot 2}{2^n} = \lim_{n\to+\infty} 2 = 2$
2. $2^{n+1} \geq 2^n \Leftrightarrow 2^n \cdot 2 \geq 2^n \Leftrightarrow 2 \geq 1 \Rightarrow 2^{n+1} \in \Omega(2^n)$.

(d) $(n+1)! \in (O \wedge \Omega)(n!)$

1. $\lim_{n\to+\infty} \frac{(n+1)!}{n!} = \lim_{n\to+\infty} \frac{(n+1)\cdot n!}{n!} = \lim_{n\to+\infty} (n+1) = +\infty$
2. $(n+1)! \geq n! \Leftrightarrow (n+1)\cdot n! \geq n! \Leftrightarrow n \geq 0 \Rightarrow (n+1)! \in \Omega(n!)$.

(e) $f(n) \in (O \wedge \Omega)(n) \left(= \lim_{n\to+\infty} \frac{f(n)}{n} = 0\right) \Rightarrow 2^{f(n)} \in (O \wedge \Omega)(2^n)$

1. $\lim_{n\to+\infty} \frac{2^f(n)}{2^n} = \lim_{n\to+\infty} 2^{(f(n)-n)\cdot\frac{n}{n}} = \lim_{n\to+\infty} 2^{(\frac{f(n)}{n}-1)n} = \lim_{n\to+\infty} 2^{-n}$
2. $\lim_{n\to+\infty} 2^{(f(n)-n)\cdot\frac{f(n)}{f(n)}} = \lim_{n\to+\infty} 2^{(1-\frac{n}{f(n)})\cdot f(n)} = \lim_{n\to+\infty} 2^{-+\infty\cdot f(n)}$

(f) $3^n \in (O \wedge \Omega)(2^n)$

1. $\lim_{n\to+\infty} \frac{3^n}{2^n} = \lim_{n\to+\infty} \left(\frac{3}{2}\right)^n = +\infty$, ya que $\frac{3}{2}$ es mayor que 1, diverge.
2. ¿$3^n \geq 2^n \Leftrightarrow log(3)\cdot n \geq log(2)\cdot n$? Lo que es cierto.

(g) $log(n) \in (O \wedge \Omega)(n^{1/2})$

1. $\lim_{n\to+\infty} \frac{log(n)}{n^{1/2}} = \frac{+\infty}{+\infty} = \text{l'Hôpital} = \lim_{n\to+\infty} \frac{\frac{1}{n}}{\frac{1}{2\sqrt{n}}} = \lim_{n\to+\infty} \frac{2\sqrt{n}}{n} = 0$
2. Este converge a 0, por lo que no puede ser $\Omega(n^1/2)$.

(h) $n^{1/2} \in (O \wedge \Omega)(log(n))$

1. Del ejercicio anterior, podemos asegurar que este no puede ser $O(log(n))$, puesto que antes no divergió.
2. Al igual que el apartado (b.2), podemos asegurar que sí lo es.

2. Demostrar que $\forall k \in N$, $log(n)^k \in O(n)$.
   Utilizando la definición (1), es equivalente a resolver:

$$\lim_{n\to+\infty} \frac{log(n)^k}{n} = \frac{+\infty}{+\infty} = \text{ l'Hôpital } = \lim_{n\to+\infty} \frac{klog(n)^{k-1}}{n} = \frac{+\infty}{+\infty}$$

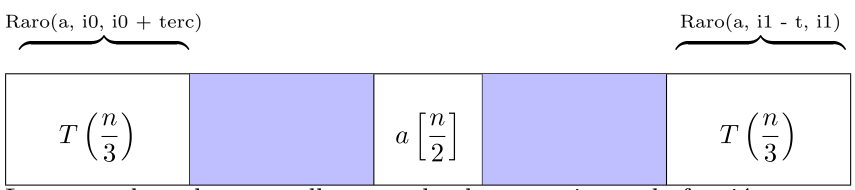
   Repitiendo la regla de l'Hôpital, $k-1$ vez más, obtenemos:

$$\lim_{n\to+\infty} \frac{k! \log(n)}{n} = \frac{+\infty}{+\infty} = \text{ l'Hôpital } = \lim_{n\to+\infty} \frac{k!}{n} = 0.$$

3. Dada la siguiente función.

```
int Raro(const int* a, int i0 ,int i1){
        if (i0 >= i1) return a[i1];
        int m = (i0 + i1)/2;
        int t = (i1 - i0)/3;
        int r0 = Raro(a, i0, i0 + terc);
        int r1 = Raro(a, i1 - t, i1);
        return a[m] + r0 + r1;
}
```

   (a) Calcular el tiempo de ejecucion de la funcion Raro(a,0,n-1) suponiendo que n es potencia de 3.
   Definimos $n = i1 - i0 = 3^k$ (1) según la función dada, analizando el código, obtenemos el siguiente esquema iterativo:



   La zona coloreada es aquella parte donde no se ejecuta la función.

$$T(n) = \begin{cases} 1 & n \leq 0 \\ 1 + 2T\left(\frac{n}{3}\right) & n \geq 1 \end{cases}$$

   Realizando la substitución (1).

$$T(3^k) = 1 + 2T(3^{k-1})$$

   i. La parte **homogénea** es: $T(3^k) - 2T(3^{k-1}) = x - 2 = 0$.
   ii. La parte **no homogénea** es: $1 = 1 \cdot 1^k$.

   Obtenemos como raíces, $\alpha = 2, 1$, el polinomio resultante es:

$$T(3^k) = 2^k A + 1^k B = T(n) = 2^{\frac{\log(n)}{\log(3)}} A + B = n^{\frac{\log(2)}{\log(3)}} A + B = n^{0.630} A + B$$

   (b) Dar una cota de complejidad para dicho tiempo de ejecución.

$$\lim_{n\to+\infty} \frac{n^{0.630}}{n} = \lim_{n\to+\infty} \frac{1}{n^{0.369}} = 0 \Rightarrow O(T(n)) = O(n^{0.630}) \in O(n)$$

4. Consideremos la siguiente función que obtiene de un árbol binario:

```
int Altura(bintree<int>& ab, bintree<int>::nodo n){
        if (n.nulo()){
                return 0;
        }
        int h0 = Altura(ab, ab.hizq(n));
        int h1 = Altura(ab, ab.hdrcah(n));
        return 1 + max(h0, h1);
}
```

Definimos n como la cantidad de nodos de nuestro sub-árbol binario y k como la cantidad a la izquierda del mismo ($n-k-1$ la cantidad derecha).

$$T(n) = \begin{cases} 1 & n = 0 \\ T(k) + T(n-k-1) + 1 & n \geq 1 \end{cases}$$

El peor caso ocurre cuando todos los nodos están **siempre** en un mismo lado, eso es equivalente a decir que $k = 0$.

$$T(n) = T(0) + T(n-1) + 1 = 1 + T(n-1) + 1 = T(n-1) + 2$$

$$T(n-1) = T(n-2) + 2 \Rightarrow T(n) = T(n-2) + 2 + 2$$

Repitiendo n veces la recurrencia, obtenemos la solución a la recurrencia.

$$O(T(n)) = O(c_1 + 2c_2 n) \in O(n)$$

Ahora bien si a la izq. como a la der. hay la mitad del total $k = \dfrac{n = 2^d}{2}$.

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + 1 = 1 + 2T\left(\frac{n}{2}\right) \Rightarrow T(2^d) = 1 + 2T(2^{d-1})$$

(a) La parte **homogénea** es: $T(2^d) - 2T(2^{d-1}) = x - 2 = 0$.

(b) La parte **no homogénea** es: $1 = 1 \cdot 1^d$.

Obtenemos como ecuación no recurrente:

$$T(n) = 2^d c_1 + 1^d c_2 = nc_1 + c_2 \Rightarrow \Omega(T(n)) = \Omega(nc_1 + c_2) \in \Omega(n)$$

Como ambas cotas concuerdan.

$$\Omega(T(n)) \subseteq O(T(n)) \Rightarrow \Theta(T(n)) \in \Theta(n)$$

5. Ordenar las siguientes funciones de acuerdo a su velocidad de crecimiento:

$$n, \sqrt{n}, \log(n), \log(\log(n)), \log^2(n), \frac{n}{log(n)}, \sqrt{n}\log^2(n), \left(\frac{1}{3}\right)^n, \left(\frac{3}{2}\right)^n, 17, n^2$$

Ordenando según la velocidad de crecimiento, definimos el símbolo $<$ como una regla de orden decreciente entre el tiempo de ejecución de dos funciones, obtenemos:

$$17 < \left(\frac{1}{3}\right)^n < \log(\log(n)) < \log(n) < \log^2(n) < \sqrt{n} \qquad (*)$$

$$(*) \qquad < \sqrt{n}\log^2(n) < \frac{n}{log(n)} < n < n^2 < \left(\frac{3}{2}\right)^n$$

6. Suponiendo que $T_1 \in O(f)$ y que $T_2 \in O(f)$, indicar cuáles de las siguientes afirmaciones son ciertas:

$$T_1, T_2 \in O(f) \Rightarrow \lim_{n\to+\infty} \frac{T_1(n)}{f(n)} = \lim_{n\to+\infty} \frac{T_1(n)}{f(n)} = 0$$

(a) $T_1 + T_2 \in O(f)$

$$\lim_{n\to+\infty} \frac{T_1(n)+T_2(n)}{f(n)} = \lim_{n\to+\infty} \frac{T_1(n)}{f(n)} + \lim_{n\to+\infty} \frac{T_2(n)}{f(n)} = 0+0=0$$

(b) $T_1 - T_2 \in O(f)$

$$\lim_{n\to+\infty} \frac{T_1(n)-T_2(n)}{f(n)} = \lim_{n\to+\infty} \frac{T_1(n)}{f(n)} - \lim_{n\to+\infty} \frac{T_2(n)}{f(n)} = 0-0=0$$

(c) $\frac{T_1}{T_2} \in O(f)$

$$\lim_{n\to+\infty} \frac{T_1(n)}{T_2(n)f(n)} = \lim_{n\to+\infty} \frac{1}{T_2(n)} \cdot \lim_{n\to+\infty} \frac{T_1(n)}{f(n)} = \lim_{n\to+\infty} \frac{1}{T_2(n)} \cdot 0 = 0$$

(d) $T_1 \in O(T_2)$

$$\lim_{n\to+\infty} \frac{T_1(n)}{T_2(n)} = \lim_{n\to+\infty} \frac{T_1(n)f(n)}{T_2(n)f(n)} = \lim_{n\to+\infty} \frac{T_1(n)}{f(n)} \cdot \lim_{n\to+\infty} \frac{f(n)}{T_2(n)} = 0 \cdot \lim_{n\to+\infty} \frac{f(n)}{T_2(n)}$$

c) y d) se cumplen $\Leftrightarrow \lim_{n\to+\infty} T_2(n) \neq 0$

7. Resolver la ecuación:

$$T(n) = \begin{cases} 0 & n = 0 \\ \dfrac{1}{n}\left(\displaystyle\sum_{i=0}^{n-1} T(i)\right) + cn & n \geq 1 \end{cases}$$

Para realizar los cálculos menos enrevesados, definimos:

$$(*) \qquad K_\alpha = \sum_{i=0}^{\alpha} T(i) = T(\alpha) + \sum_{i=0}^{\alpha-1} T(i) = T(\alpha) + K_{\alpha-1}$$

Calculemos el término $T(n-1)$.

$$T(n-1) = \frac{1}{n-1}K_{n-2} + c(n-1) \Rightarrow K_{n-2} = (T(n-1) - c(n-1))(n-1)$$

Simplificando, obtenemos el valor de K_{n-2}.

$$K_{n-2} = T(n-1)(n-1) - c(n-1)^2$$

Ahora bien, vamos a aplicar la igualdad (*) para $T(n)$.

$$T(n) = \frac{1}{n}K_{n-1} + cn = \frac{1}{n}(T(n-1) + K_{n-2}) + cn$$

Usando el valor de K_{n-2} que hemos calculado antes.

$$T(n) = \frac{1}{n}(T(n-1) + T(n-1)(n-1) - c(n-1)^2) + cn$$

Agrupamos y expandamos los polinomios.

$$T(n) = \frac{T(n-1)(n-1+1)}{n} - \left(\frac{-2n+1}{n}\right)c$$

Simplificamos.

$$T(n) = T(n-1) + 2c - \frac{1}{n}c$$

Para acortar un poco, vamos a llamar $T(n) = t_n$.
Como ya tenemos calculado t_n, vamos a calcular t_{n-1}.

$$t_{n-1} = t_{n-2} + \left(2c - \frac{1}{n-1}c\right) \Rightarrow t_n = \left(t_{n-2} + 2c - \frac{1}{n-1}c\right) + \left(2c - \frac{1}{n}c\right)$$

Agrupamos los términos de t_n

$$t_n = t_{n-2} + (2c + 2c) - \left(\frac{1}{n} + \frac{1}{n-1}\right)c$$

Si seguimos expandiendo, este patrón será visible para cada iteración, por lo que la solución a la ecuación es

$$t_n = 2nc - \left(\sum_{k=0}^{n} \frac{1}{k}\right)c = 2nc - \mathcal{H}_n c$$

Dónde $\mathcal{H}_n$ es conocida como la serie armónica hasta el término n-ésimo.