Estructuras de Datos Avanzadas
Tarea 1

1. Para cada función *f(n)* y tiempo *t* determine el tamaño máximo del problema (la *n*) que puede resolverse en tiempo *t*. Suponga que el algoritmo usado para resolver el problema toma *f(n)* microsegundos (reporte sólo el orden de magnitud si los números son demasiado grandes)

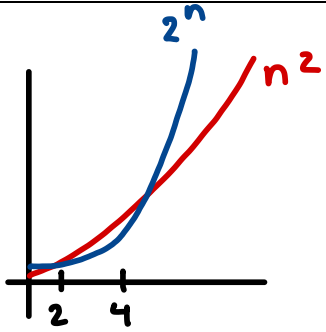
	1 Segundo	1 Minuto	1 Hora	1 Día	1 Mes	1 Año	1 Siglo
$\log_2(n)$	2^1E6	2^6E7	2^3.6E9	2^8.6E10	2^2.6E12	2^3.1E13	2^3.1E15
$\sqrt{n}$	1E12	3.6E15	1.2E19	7.4E21	6.7E24	9.9E26	9.9E30
N	1E6	6E7	3.6E9	8.6E10	2.6E12	3.1E13	3.1E15
$n \log_2(n)$	6.27E4	2.8E6	13E8	2.7E9	7.1E10	7.9E11	6.8E13
n^2	999	7745	5.9E4	2.9E5	1.6E6	5 6E6	5.6E7
n^3	99	391	1532	4420	1.3E4	3.1E4	1.4E5
2^n	19	25	31	36	41	44	51
$n!$	9	11	12	13	15	16	17

2. Supongamos que estamos comparando el desempeño de dos algoritmos de ordenamiento. Para entradas de tamaño n, el algoritmo A toma $8n^2$ operaciones mientras que el algoritmo B toma $64n\log_2(n)$. ¿Para qué valores de n es mejor el desempeño de A? [2, 43]
3. ¿Cuál es el valor más chico de *n* para el cual un algoritmo que toma $100n^2$ es más rápido que uno que toma 2^n (en la misma máquina)? 15
4. Demuestre que $2^n = O(n^2)$ (?) $n^2 = O(2^n)$

4: $2^n = n^2 \iff n = 2 \text{ o } n = 4$

$2^1 = 2 \quad 2^3 = 8 \quad 2^5 = 32$

$1^2 = 1 \quad 2 \quad 3^2 = 9 \quad 4 \quad 5^2 = 25$ $\Rightarrow$



con $n_0 = 5$

$n^2 \leq c\,2^n \Rightarrow n^2 = O(2^n)$

$2^n = \Omega(n^2)$