Practica 8

1) Determinar para cada función t(n) en la siguiente tabla, cual es el mayor tamaño n de una instancia de un problema que puede ser resuelto en cada uno de los tiempos indicados en las columnas de la tabla, suponiendo que el algoritmo para resolverlo utiliza t(n) microsegundos.

t(n)	1 seg.	1 min.	1 hora	1 día	1 mes	1 año	1 siglo
$\log_2(n)$	2^{10^6}	$2^{6 * 10^7}$	$2^{36 * 10^8}$	$2^{864 * 10^8}$	$2^{25920 * 10^8}$	$2^{311040 * 10^8}$	$2^{31104000 * 10^8}$
$\sqrt{n}$	10^{12}	$(6 * 10^7)^2$	$(36 * 10^8)^2$	$(864 * 10^8)^2$	$(25920 * 10^8)^2$	$(311040 * 10^8)^2$	$(31104000 * 10^8)^2$
n	10^6	$6 * 10^7$	$36 * 10^8$	$864 * 10^8$	$25920 * 10^8$	$311040 * 10^8$	$31104000 * 10^8$
$n \times \log_2(n)$	62746	2801417	133380000	2755100000	71871000000	787089606198	6769949846 3641
n^2	$\sqrt{10^6}$	$\sqrt{6 * 10^7}$	$\sqrt{36 * 10^8}$	$\sqrt{864 * 10^8}$	$\sqrt{25920 * 10^8}$	$\sqrt{311040 * 10^8}$	$\sqrt{31104000 * 10^8}$
2^n	$\frac{\ln(10^6)}{\ln(2)}$	$\frac{\ln(6 * 10^7)}{\ln(2)}$	$\frac{\ln(36 * 10^8)}{\ln(2)}$	$\frac{\ln(864 * 10^{88})}{\ln(2)}$	$\frac{\ln(25920 * 10^8)}{\ln(2)}$	$\frac{\ln(311040 * 10^8)}{\ln(2)}$	$\frac{\ln(31104000 * 10^8)}{\ln(2)}$
n!	9	11	12	13	15	16	17

2) Si el tiempo de ejecución en el mejor caso de un algoritmo, tm(n), es tal que tm(n) ∈ Ω(f(n)) y el tiempo de ejecución en el peor caso de un algoritmo, tp(n), es tal que tp(n) ∈ O(f(n)), ¿Se puede afirmar que el tiempo de ejecución del algoritmo es Θ(f(n))?

   Si, se puede afirmar que el tiempo de ejecución del algoritmo es Θ(f(n)), ya que esta acotado inferiormente en el mejor caso por f(n) (tm(n) ∈ Ω(f(n))) y esta acotado superiormente en el peor caso por f(n) (tp(n) ∈ O(f(n))).

   En el mejor caso el tiempo de ejecución va a ser más grande o igual que c1(constante positiva) * f(n) y en el peor caso va a ser menor o igual que c2 (constante positiva) * f(n). Con ello, se cumple la definición de Θ(f(n)) {t:N → R+ / ∃ c1, c2 ∈ R+, n0 ∈ N tq c1 f(n) ≤ t(n) ≤ c2 f(n), n ≥ n0}

3) Un algoritmo tarda 1 segundo en procesar 1000 items en una máquina determinada. ¿Cuánto tiempo tomara procesar 10000 items si se sabe que el tiempo de ejecución del algoritmo es n^2 ? ¿y si se sabe que es $n \times \log_2 n$? ¿Qué se estaría asumiendo en todos los casos?

   1000^2 ➔ 1 segundo

   10000^2 ➔ 100 segundos (regla de 3 simples duh)

   $1000^* \log_2 1000$ ➔ 1 segundo

   $10000^* \log_2 10000$ ➔ 40/3 = 13,3333

   En todos los casos se asume que es el peor.

4) Un algoritmo toma n^2 días y otro n^3 segundos para resolver una instancia de tamaño n de un problema. Mostrar que el segundo algoritmo superara en tiempo al primero solamente en instancias que requieran más de 20 millones de años para ser resueltas.

$T1(n) = n^2$ días

$T2(n) = n^3$ segundos

1. Convertimos a misma unidad de tiempo. Paso días a segundos

   $T1(n) = n^2 * 60 * 60 * 24$ segundos $= n^2 * 86400$ segundos

2. $T1(n) / T2(n) = 86400 / n$

   La relación entre los tiempos T1 y T2 depende inversamente de n. Esto significa que a medida que n aumenta, T1 se vuelve relativamente más pequeño en comparación con T2.

3. Ahora vamos a ver en qué valor de n ambos algoritmos tardan lo mismo:

   - $n^2 * 86400 = n^3 \rightarrow$

     $n = 86400$

   - $n^3 > n^2 * 86400 \rightarrow$

     $n > 86400$

     Es decir, para cualquier valor de n mayor a 86400 el segundo algoritmo va a tardar más que el primero.

4. Ambos algoritmos para 86400 van a tardar un total de 86400^3 segundos que son aproximadamente 20438383,1 años. De esta manera, queda demostrado que para instancias que requieran aproximadamente mas de 20 millones de años para ser resueltas el segundo algoritmo superara en tiempo al primero.