# Archivo:Logo Instituto Politécnico Nacional.png - Wikipedia, la enciclopedia libreArchivo:EscudoESCOM.png - Wikipedia, la enciclopedia libreINSTITUTO POLITÉCNICO NACIONAL

# ESCUELA SUPERIOR DE CÓMPUTO

Vargas Hernández Carlo Ariel

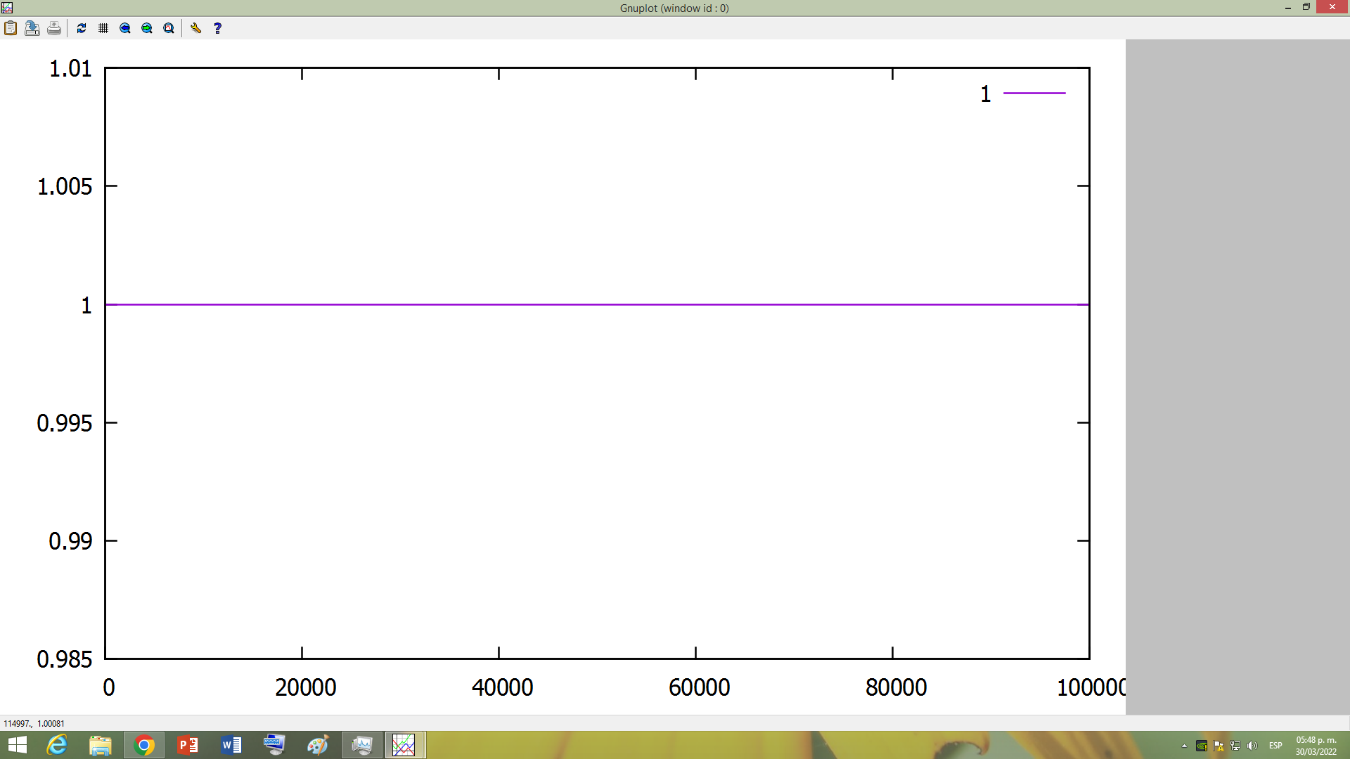
Análisis de algoritmos

3CM15

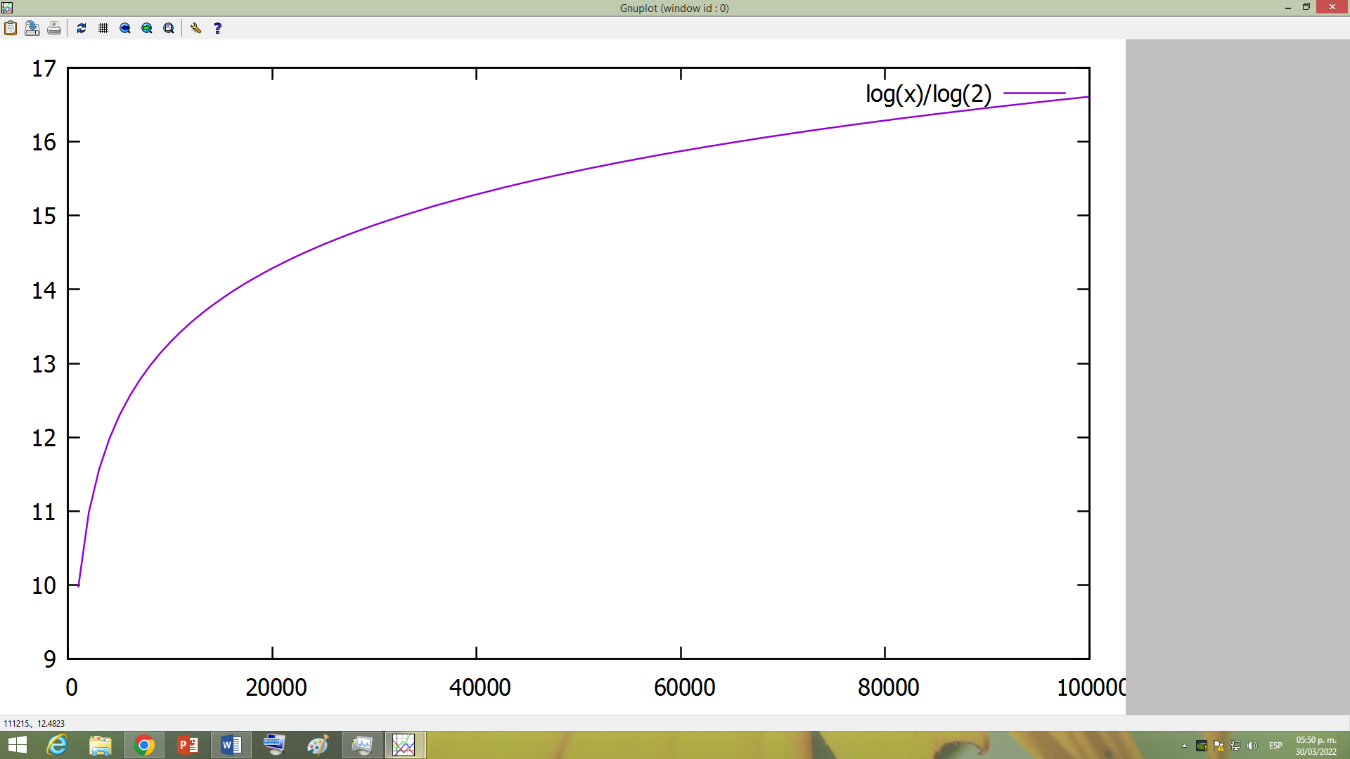
Ejercicio 6



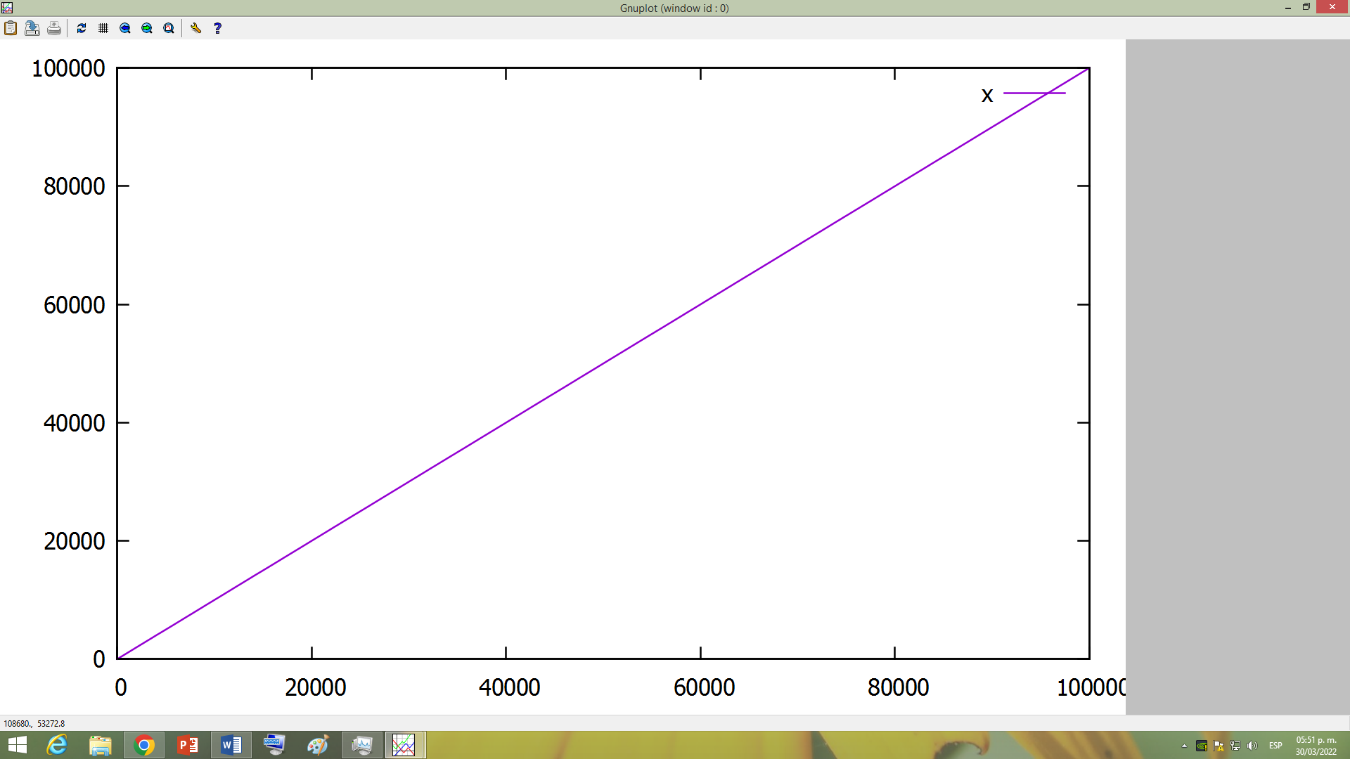
**Complejidad constante [O(1)]:**



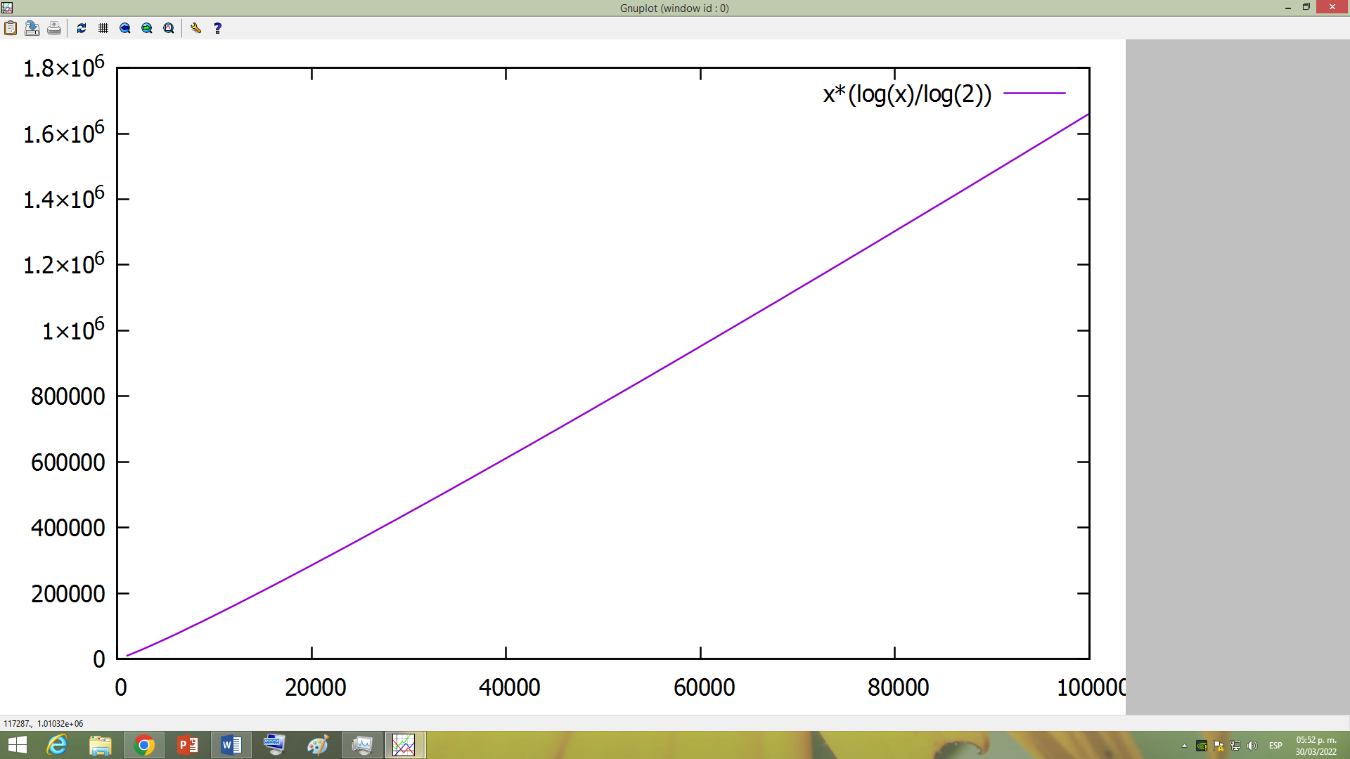
**Complejidad logarítmica [O(log(n))]**



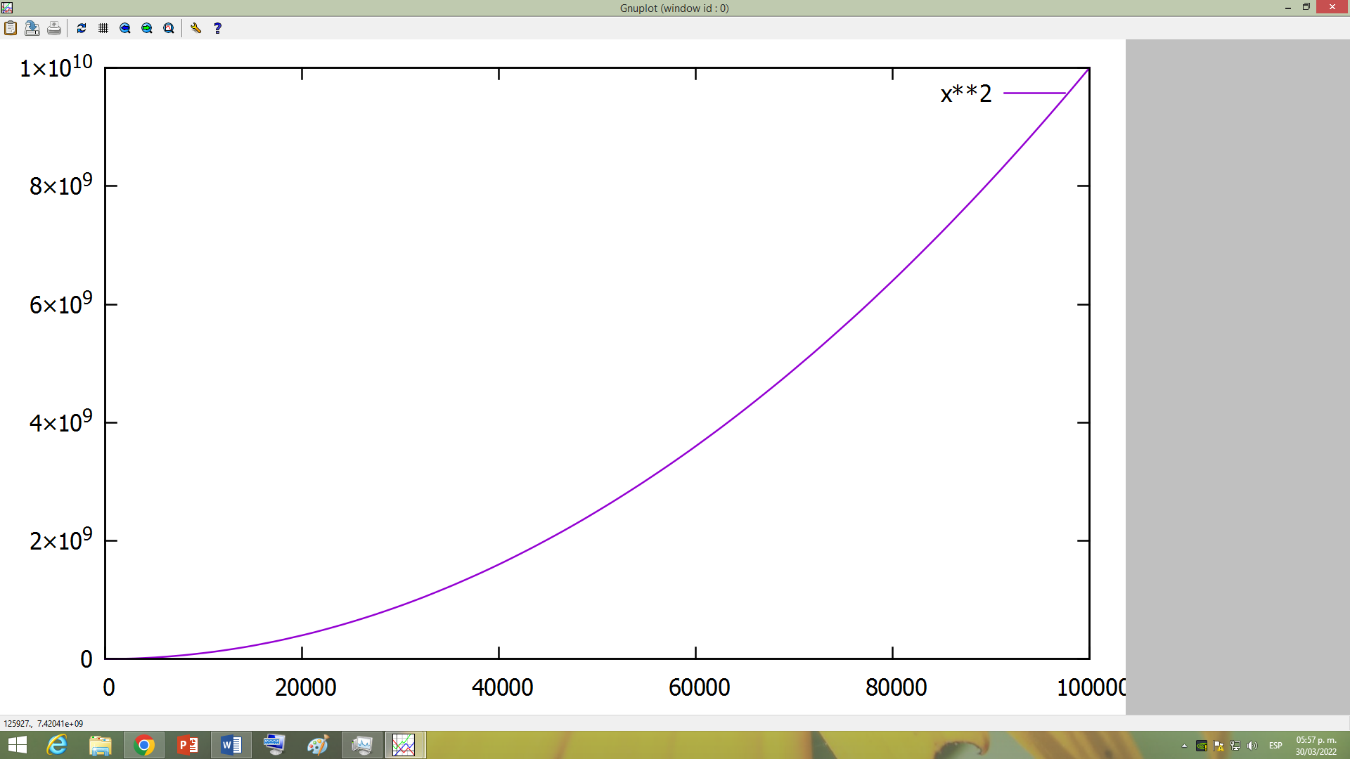
**Complejidad lineal [O(n)]**



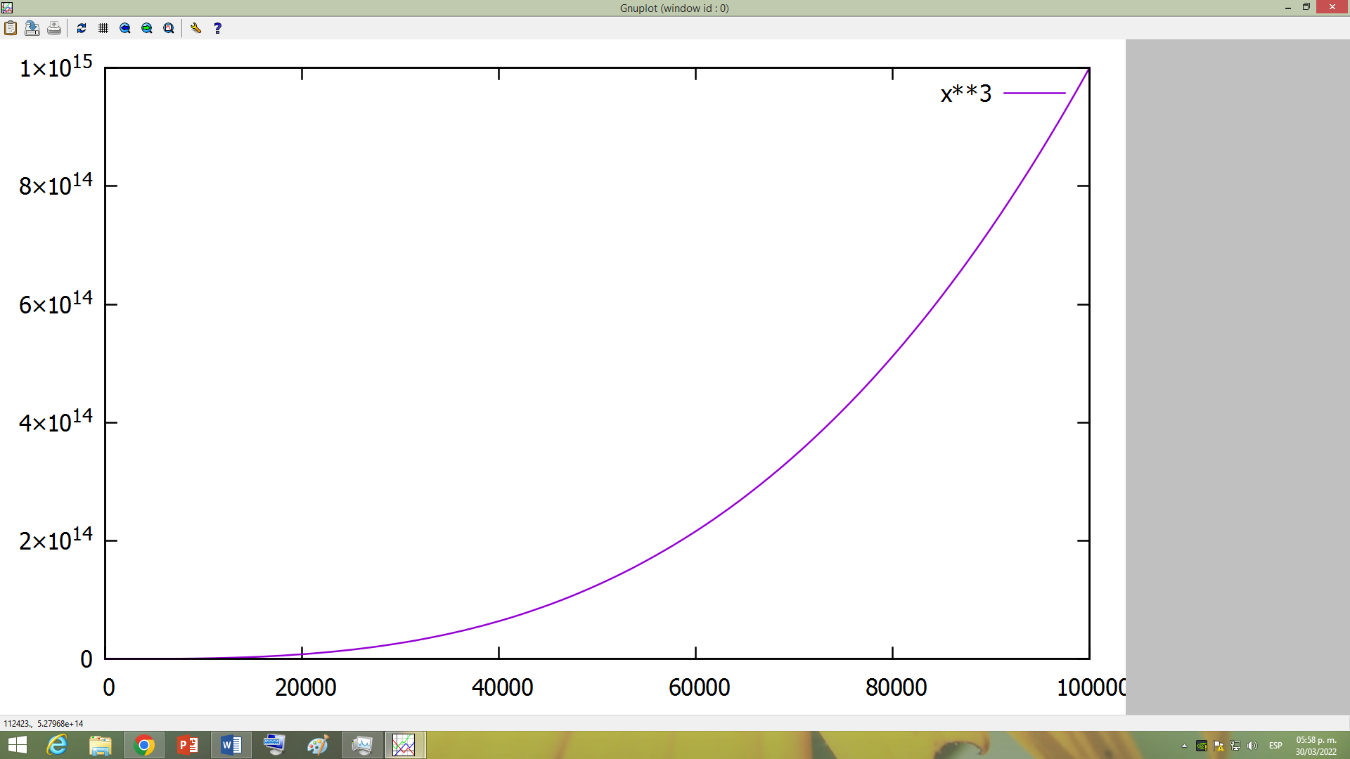
**Complejidad “n log n” [O(nlog(n))]**



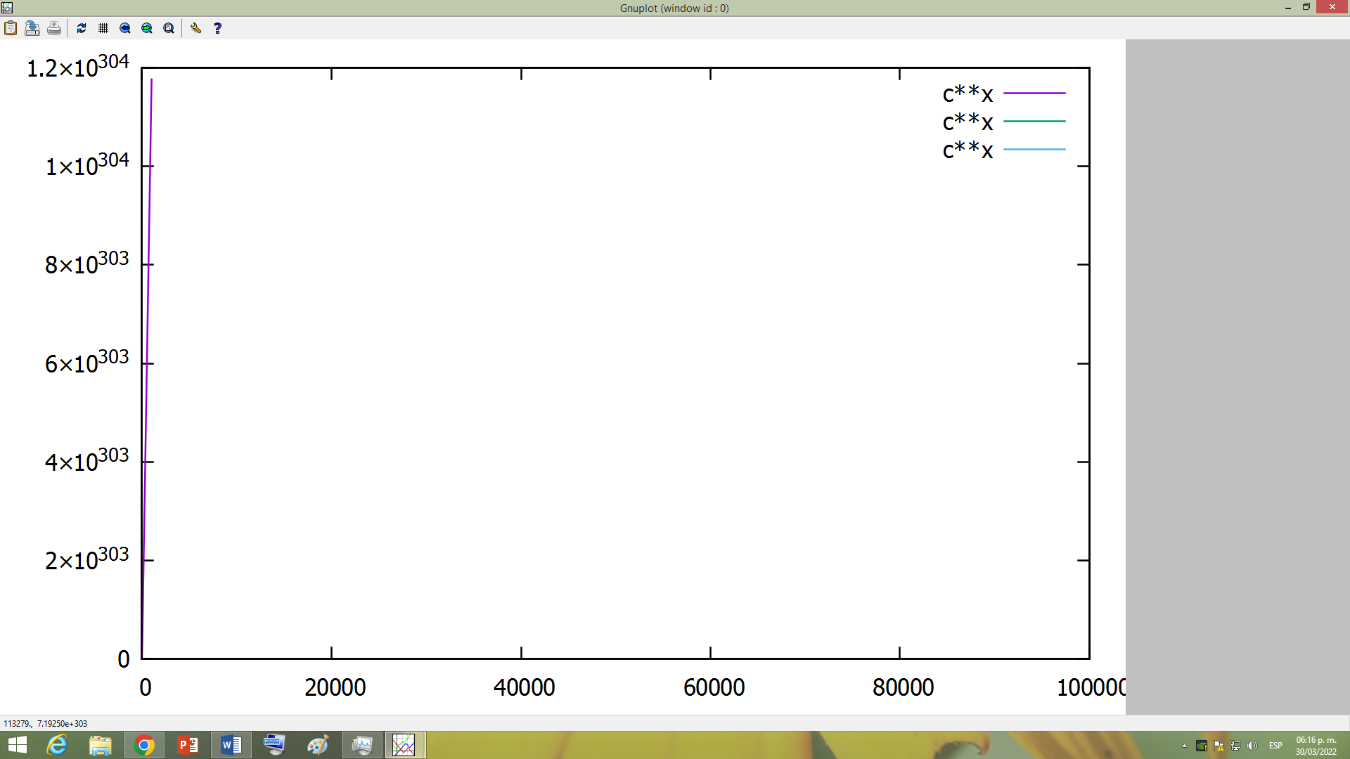
**Complejidad cuadrática [O(n2)]**



**Complejidad cubica [O(n3)]**



**Complejidad exponencial [O(cn);c>1] (con c[2:4])**

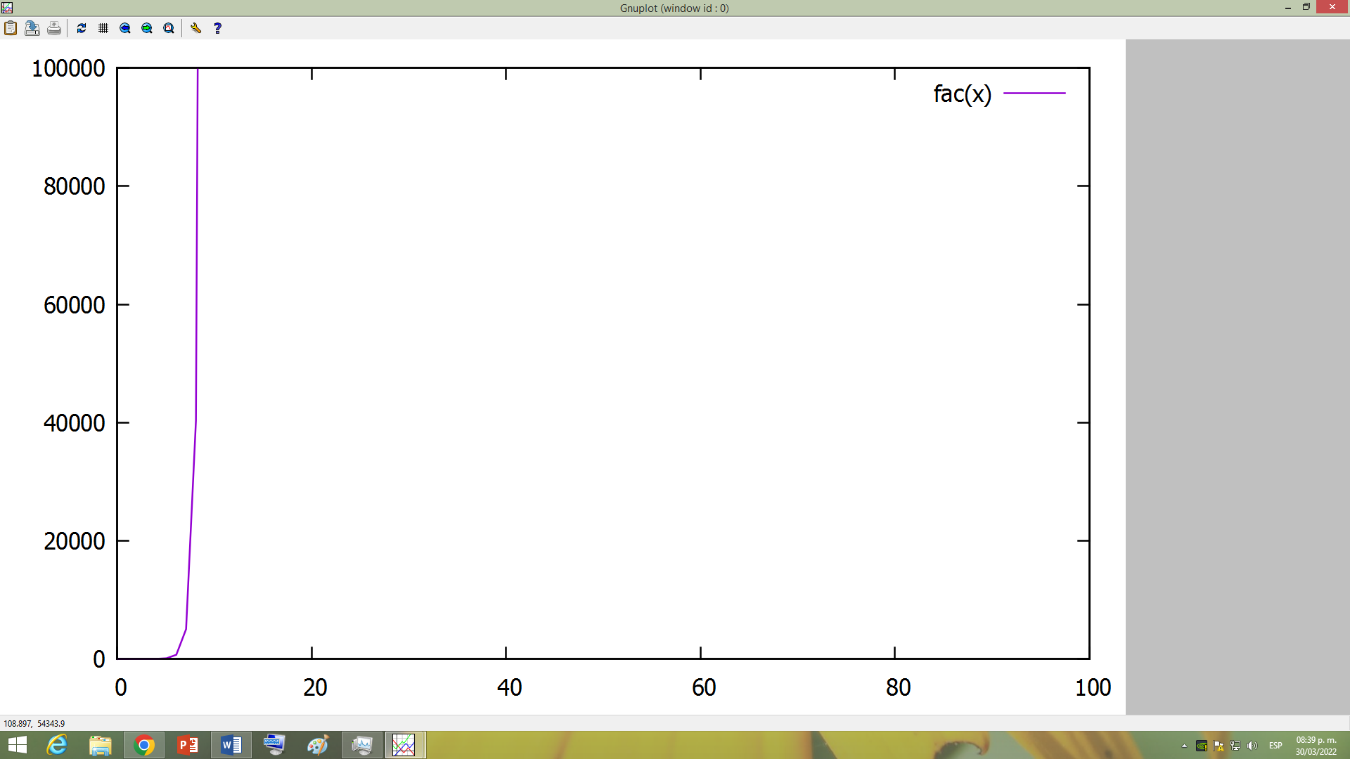


**Complejidad factorial [O(n!)]**

Para esta complejidad grafiqué hasta n=100 ya que tuve que hacer una función en gnuplot

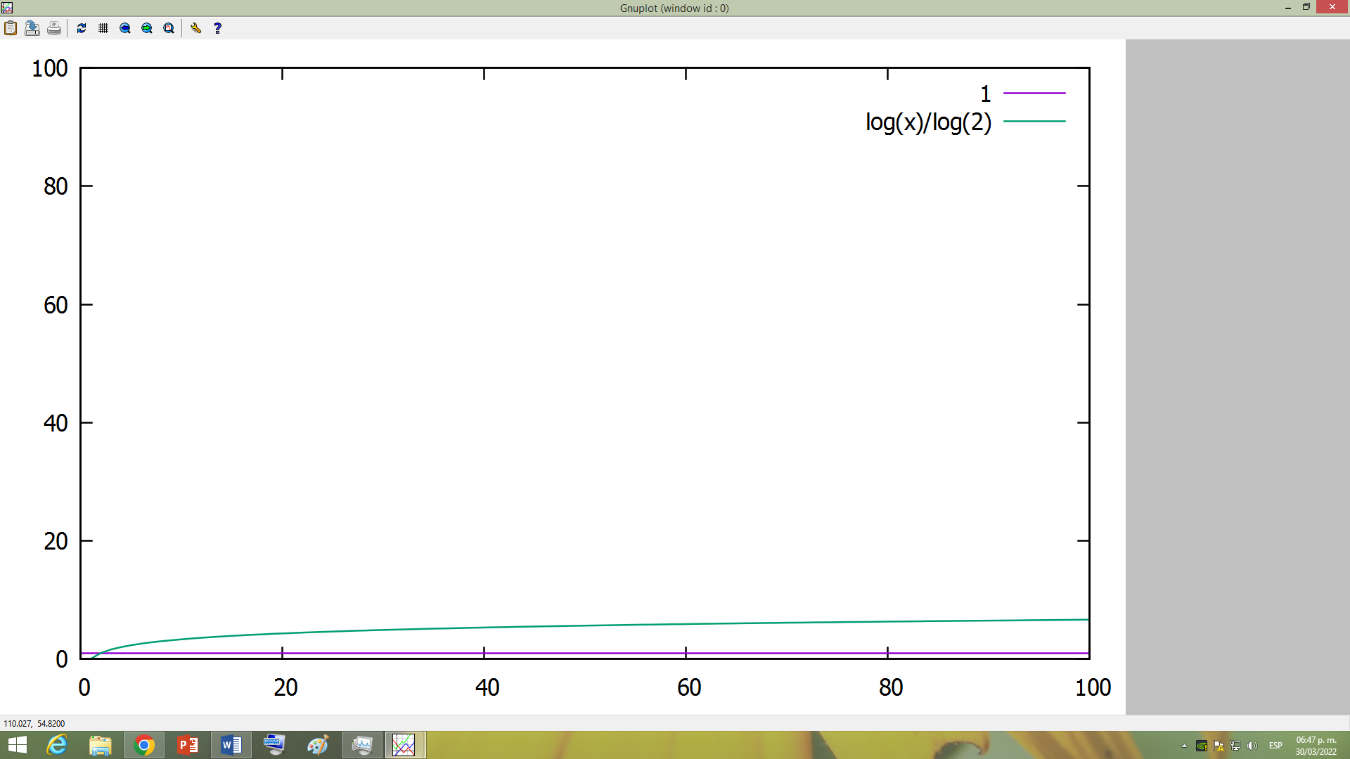
fac(x)=(int(x)==0)?1:int(x)\*fac(int(x)-1)

Con una n muy grande se desbordaba (stack overflow), también acote el eje y a 100000 para que se pudiera apreciar la curva.



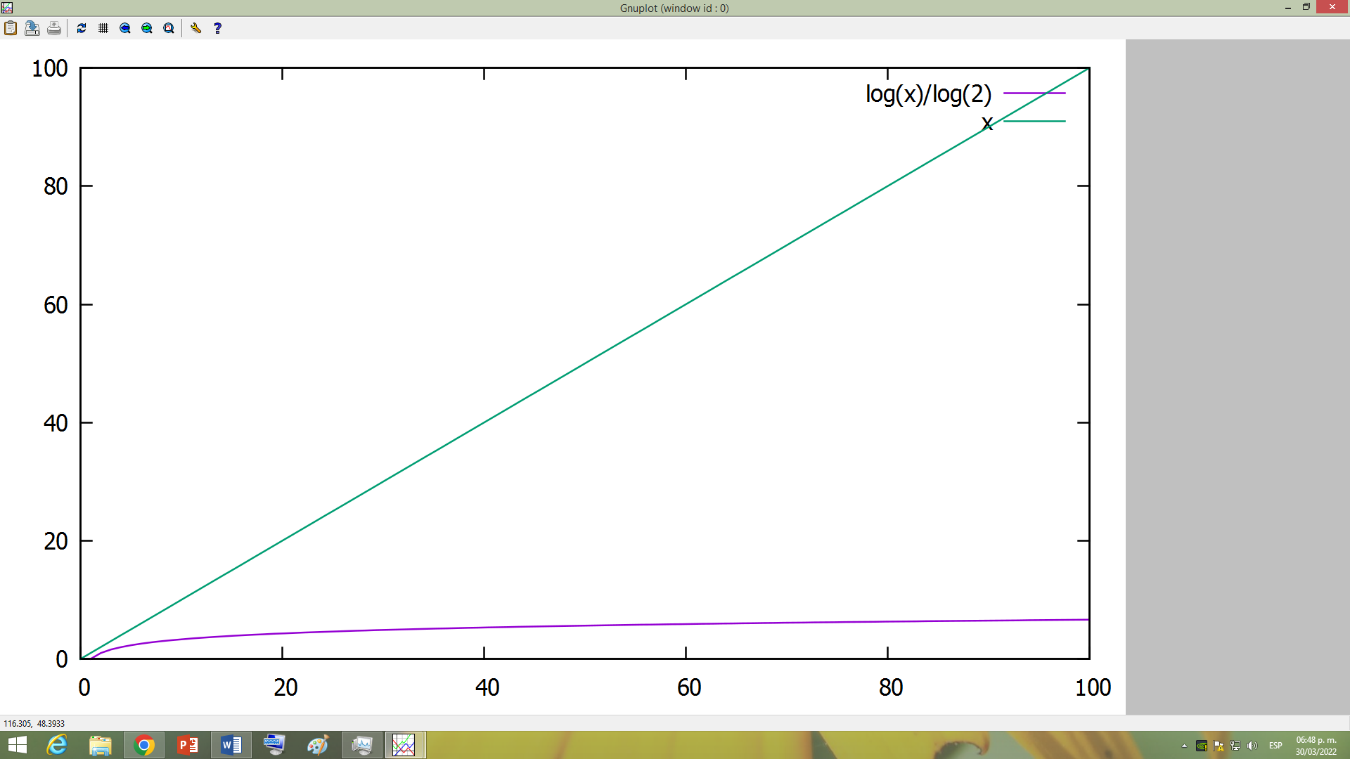
**O(1) vs O(log(n))**

En este caso la mejor complejidad es constante asi que elegiría.



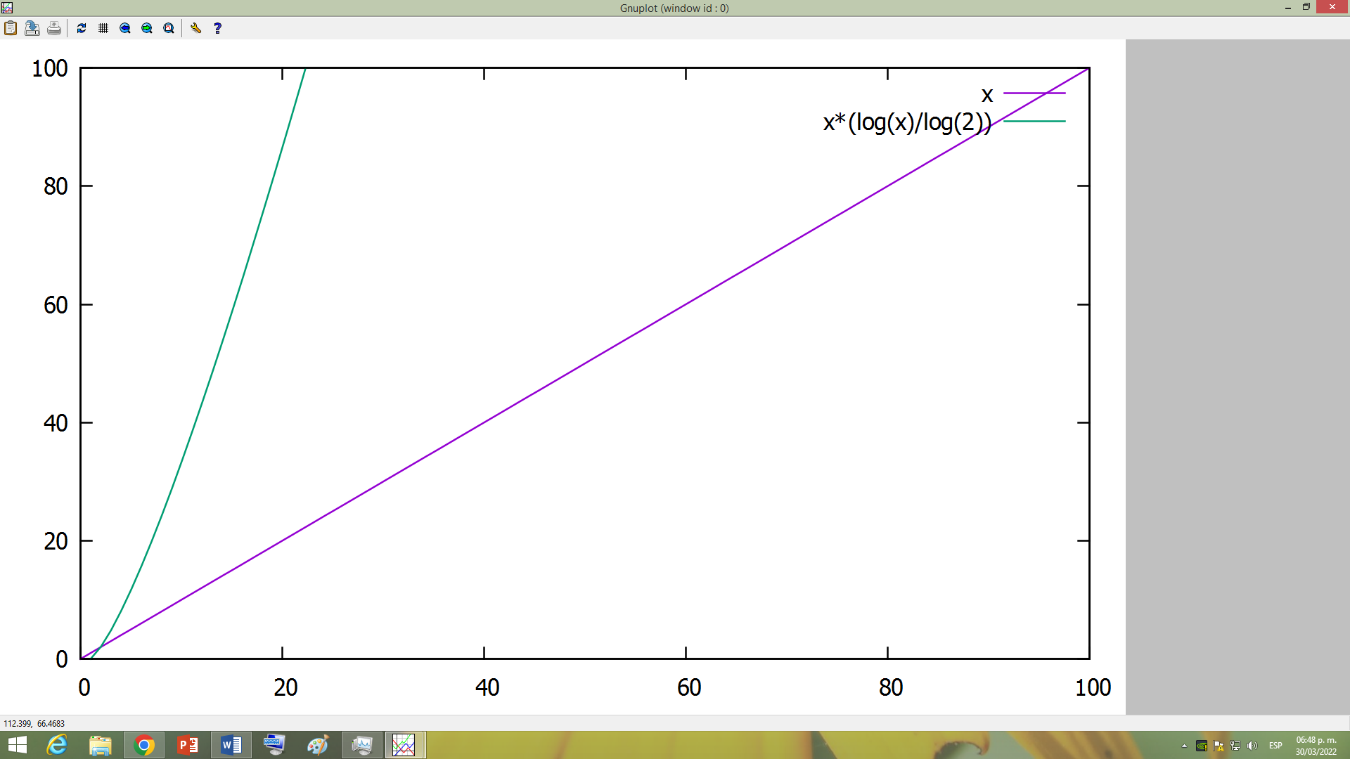
**O(log(n)) vs O(n)**

En la gráfica podemos apreciar que la curva n crece más rápido que la log(n) así que elegiría O(log(n)).



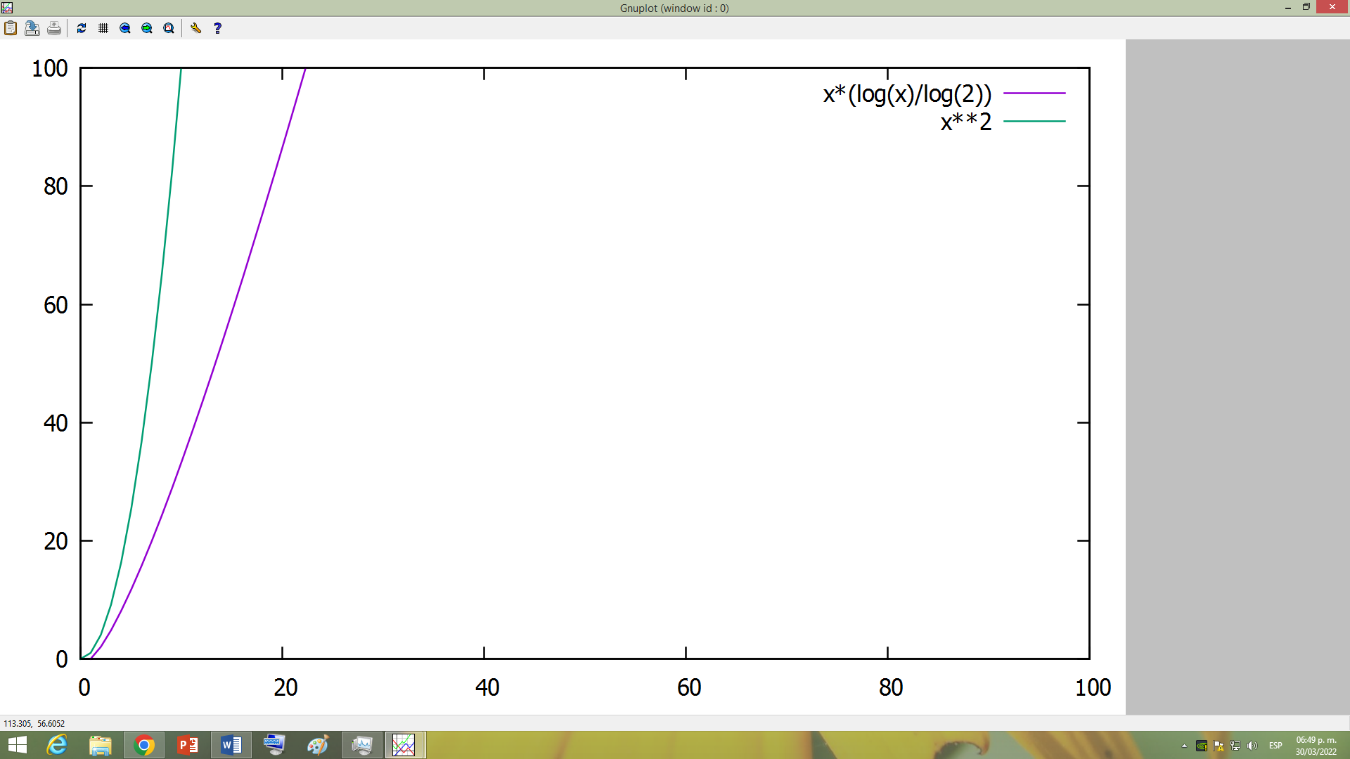
**O(n) vs O(nlog(n))**

En esta grafica podemos ver que para una n muy pequeña es mejor nlog(n), pero es tan pequeña que lo mejor sería usar O(n).



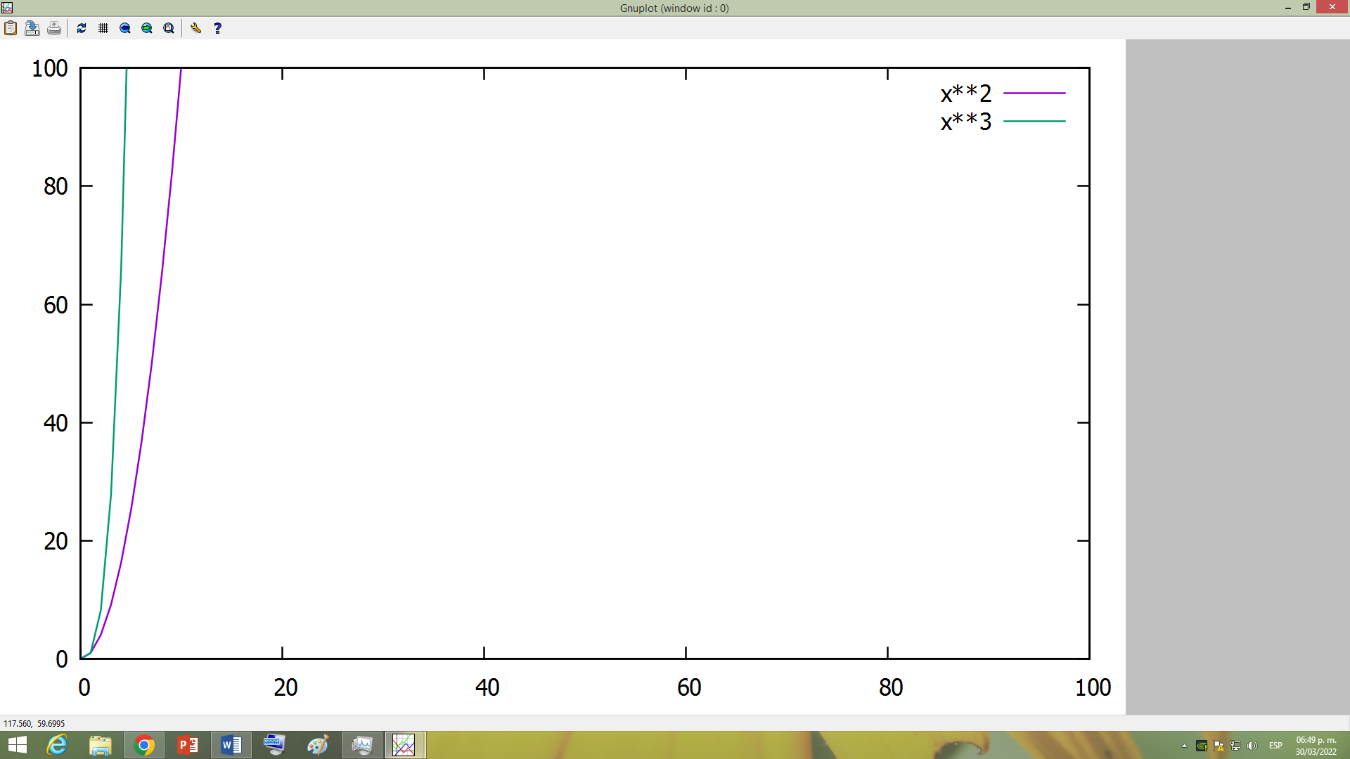
**O(nlog(n)) vs O(n2)**

En esta gráfica se puede apreciar que lo mejor es usar O(nlog(n)) ya que desde un inicio es mejor que la cuadrática.



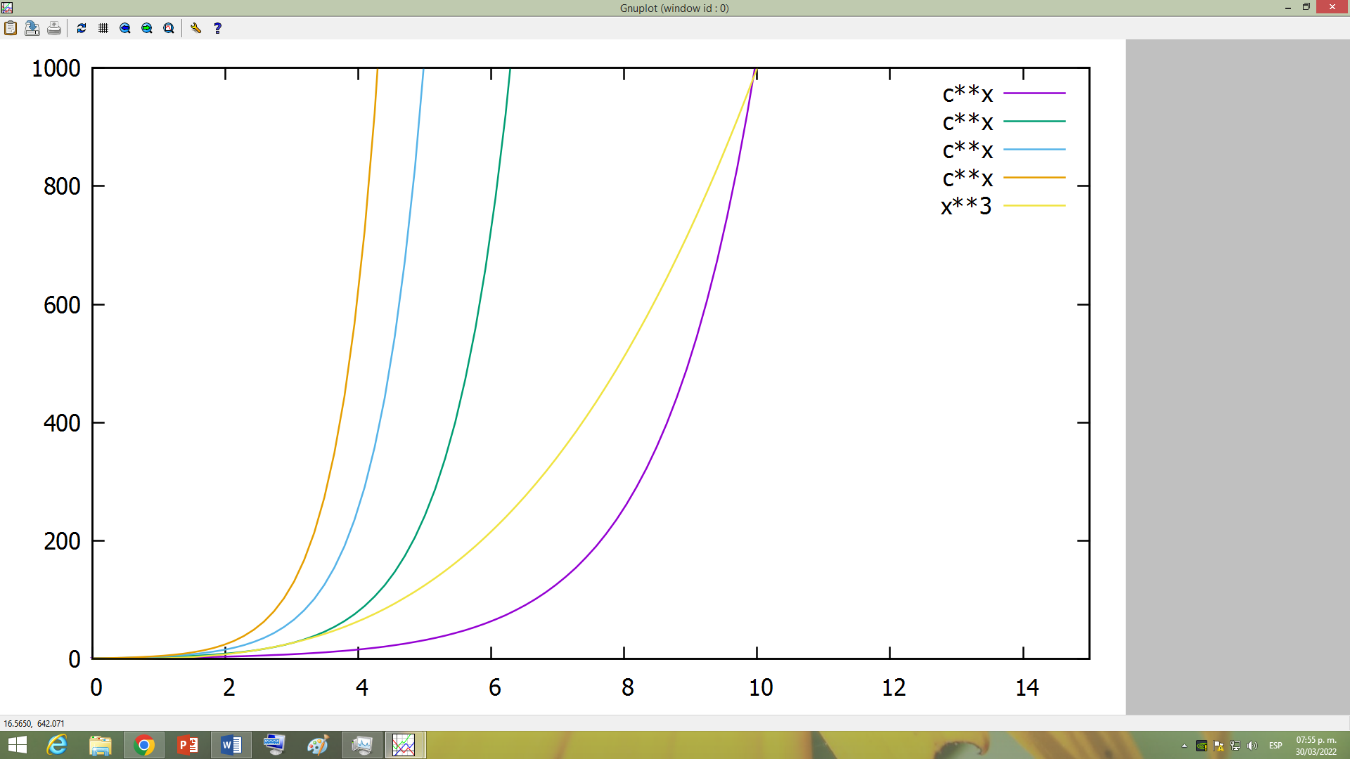
**O(n2) vs O(n3)**

Mismo caso que el anterior.

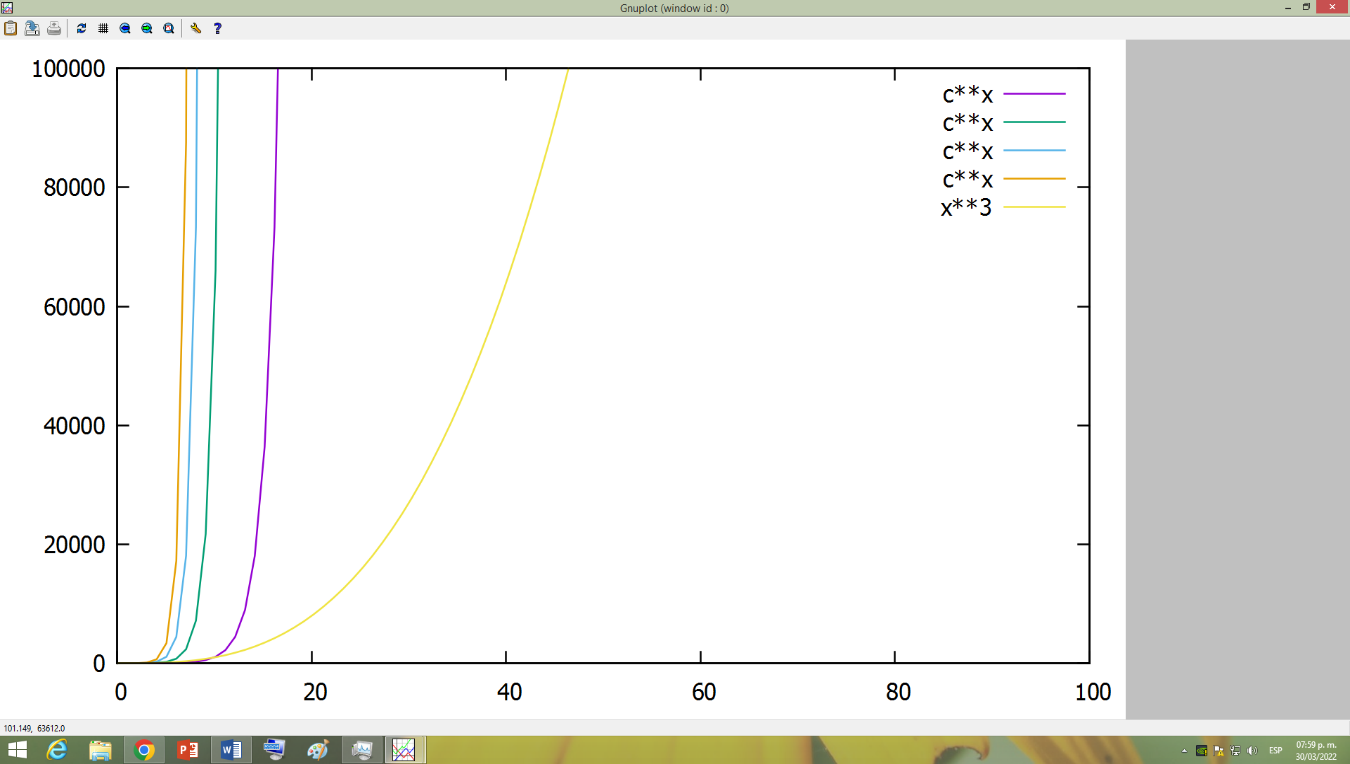


**O(n3) vs O(cn), c>1 con c[2:3] en la gráfica**

En esta comparacion hay que saber que tan grande es la n para saber que es mejor usar, por ejemplo si graficamos la comparacion con c que toma valores desde 2 hasta 5 con una n=15 y n=100, las graficas se ven así:

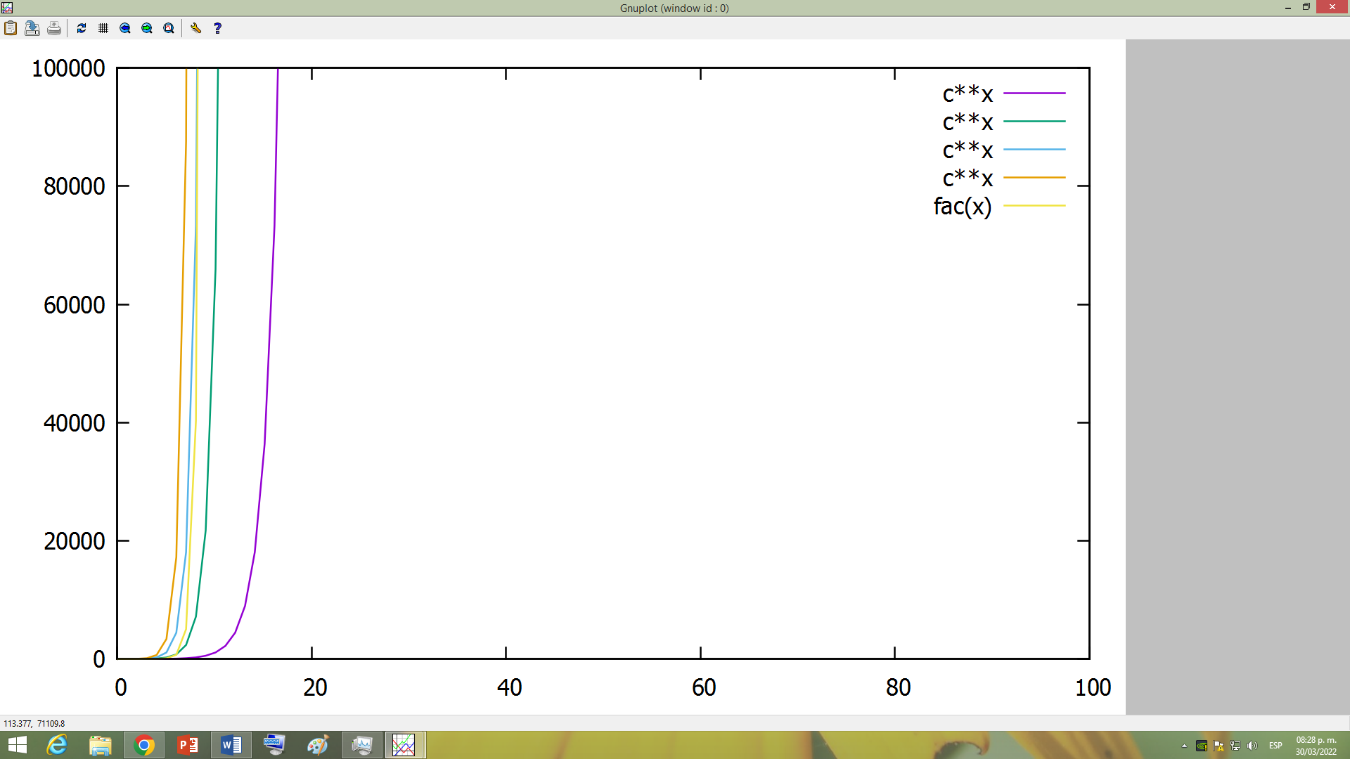


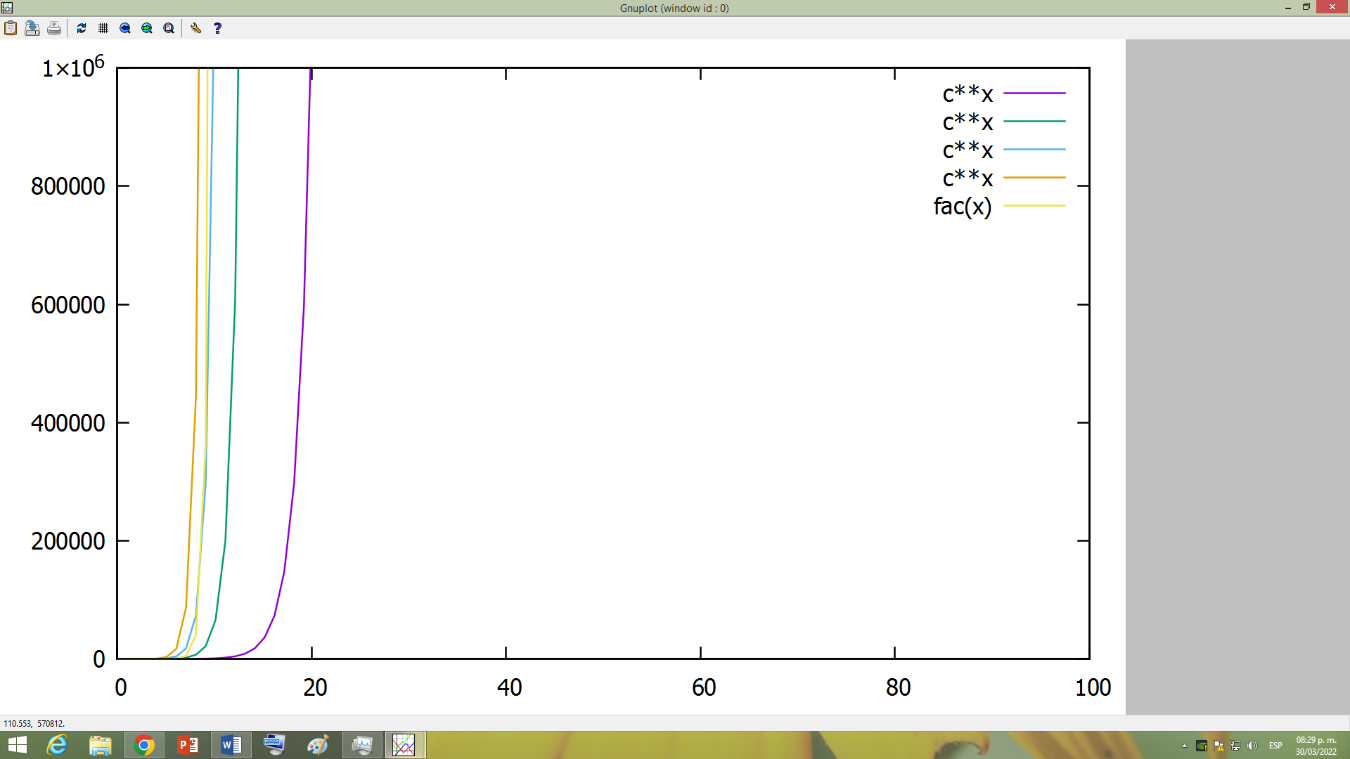
En esta primera gráfica de n=15 y con el eje y acotado a 1000 podemos ver que si la n es aproximadamente 10 y c=2 lo mejor sería usar O(cn).



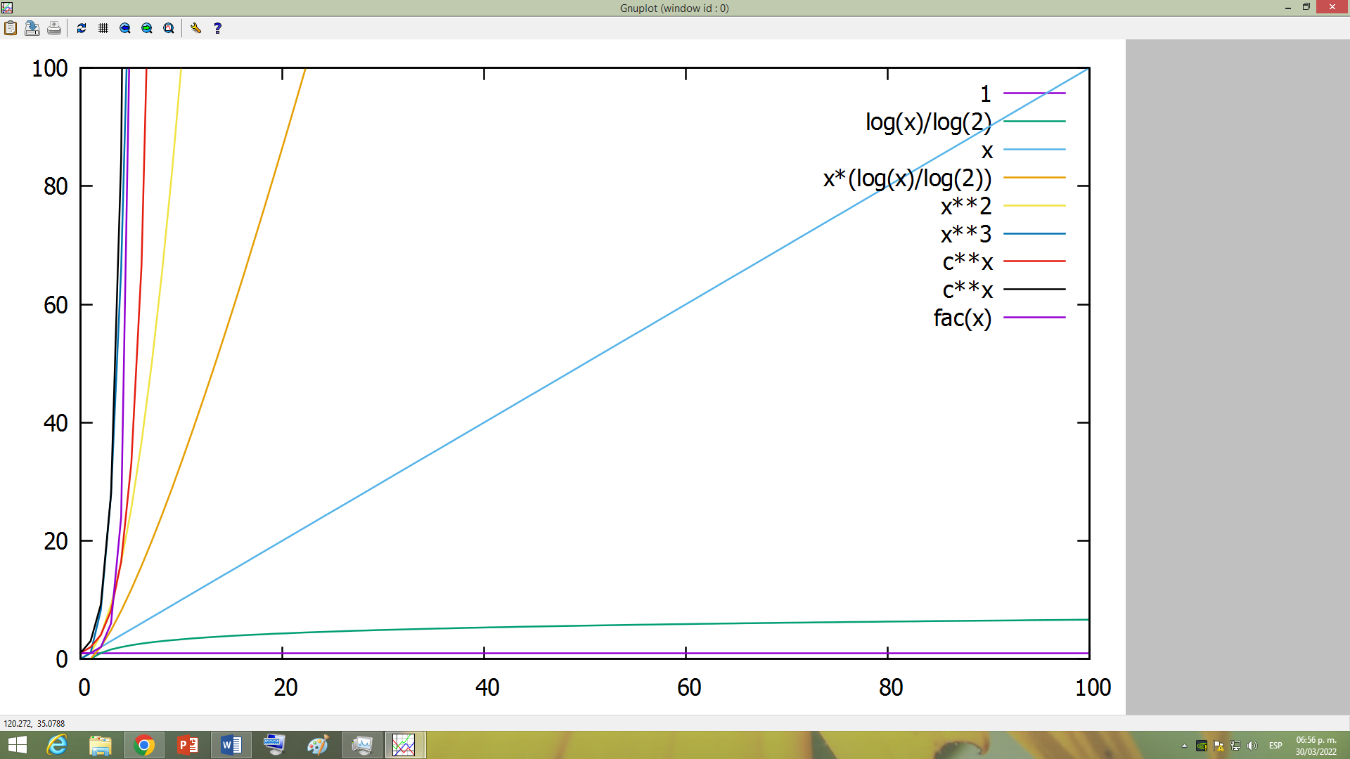
Sin embargo después de ese punto es mejor usar la complejidad cuadrática ya que la otra complejidad (para efecto el ejemplo de comparación) solo tiene ventaja con c=2 y n~10.

**O(cn) vs O(n!), c>1 con c[2:3] en la gráfica**

Para la factorial pasa algo similar a la anterior, hay n para las cuales O(n!) es mejor en ciertos valores de c en O(cn), quizá para valores muy grandes de c haya alguna diferencia considerable, pero en general es mejor O(cn).



**Todas las complejidades**



Por lo que pude observar, para tamaños del problema pequeños es casi indiferente que algoritmo usar, incluso en ciertos casos es mejor usar alguna complejidad que se considera “peor”. Fuera de estos casos con condiciones particulares lo ideal es usar la complejidad que tenga la curva más cerca al eje de las x.