Niñas PRO(grama doras)

Niñas PRO(grama doras)

Tips Programación Competitiva

Niñas PRO(grama doras)

Temario

- ★ Tips Programación Competitiva
- ★ Temáticas de Problemas
- ★ Complejidad

Niñas PRO(grama doras)

¿Qué es programación competitiva?

Dados problemas conocidos de ciencias de la computación, resolverlos lo más rápido posible.

Programa rápido

Niñas PRO(grama doras)

Cosas que te pueden ayudar a lograr esto son:

1. Escoge un editor y conócelo
    a. Aprende cómo configurar las vistas y ventanas que necesitas
    b. Atajos de Teclado

2. Practicar mucho

Identifica los problemas

Recibirás varios problemas (entre 4 a 6) de diferentes tipos. Es importante que intentes categorizarlos.

Identifica los problemas

En la competencia Regional los tipos de problemas pueden ser:

1. Ad Hoc
    a. Straightforward
    b. Simulación
    c. Matemáticos o geometría computacional (función módulo, teorema de pitágoras, perímetro y área de figuras, etc.)
2. Búsqueda Exhaustiva
3. Divide y Vencerás
3. Procesamiento de Strings

Identifica los problemas

En la competencia Nacional los tipos de problemas pueden ser alguno de los anteriores o:

1. Grafos
    a. Algoritmos BFS, DFS
    b. Distancia mínima entre nodos con peso (Dijkstra)
    c. Distancia mínima entre nodos sin peso (BFS)
2. Programación Dinámica

Hacer Análisis de Algoritmos

Dado el número y tamaño de la entrada ¿Puede mi solución pasar el límite de tiempo especificado en el problema?

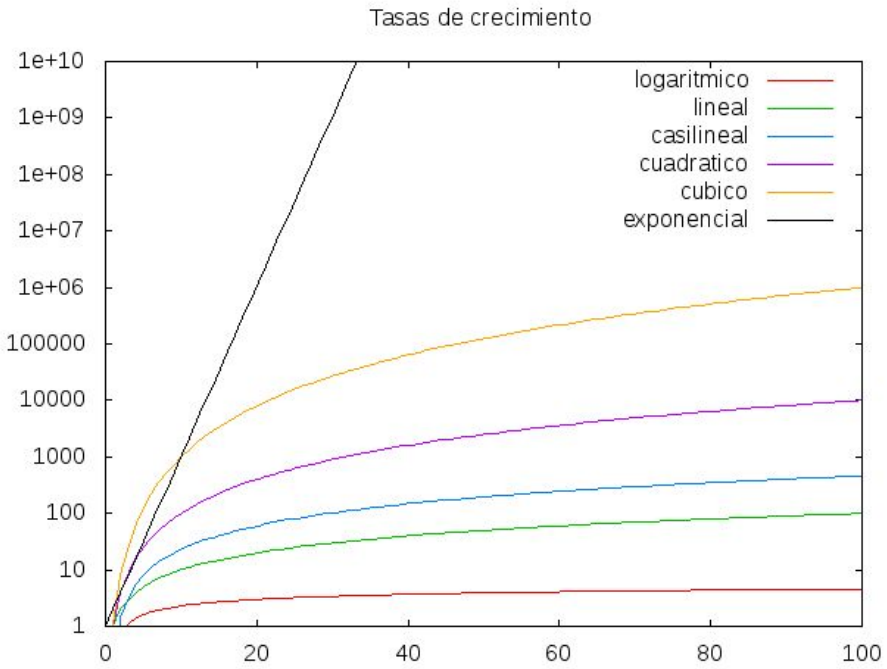
Tu objetivo: “Encontrar la solución más simple que funcione”

Hacer Análisis de Algoritmos

Orden	Nombre
O(1)	constante
O(log n)	logarítmica
O(n)	lineal
O(n log n)	casi lineal
O(n^2)	cuadrática
O(n^3)	cúbica
O(a^n)	exponencial

Principales Órdenes de Complejidad

Hacer Análisis de Algoritmos



Hacer Análisis de Algoritmos

Niñas PRO(grama doras)

Jerarquía de la complejidad
¿Cuál algoritmo es más rápido?

$$O(1) \subset O(\log n) \subset O(n) \subset O(n \log n) \subset O(n^2) \subset O(n^3) \subset O(2^n)$$

Hacer Análisis de Algoritmos

T(n)	n = 100	n = 200
log(n)	1 h.	1.15 h.
n	1 h.	2 h.
nlog(n)	1 h.	2.30 h.
n^2	1 h.	4 h.
n^3	1 h.	8 h.
2^n	1 h.	$1.27*10^{30}$ h.

Efecto de duplicar el tamaño de la entrada.

Hacer Análisis de Algoritmos

T(n)	t = 1h	t = 2h
log(n)	n = 100	n = 10000
n	n = 100	n = 200
nlog(n)	n = 100	n = 178
n^2	n = 100	n = 141
n^3	n = 100	n = 126
2^n	n = 100	n = 101

Efecto de duplicar el tiempo disponible.

Prueba tu código (Debuggear)

Verifica bien los casos de prueba

Incluir casos *tricky*

Prueba con números grandes

¿Cuál es el peor caso con el que debería probar?

Preparación

★ ¡Revisa que tus apuntes estén correctos!

★ Prioriza los problemas fáciles

★ Comienza leyendo todo

★ Usar el Scoreboard

★ Si hay dudas releer el enunciado completo

Línea Gráfica

Lorena Gonzalez - Diseñadora
@soygonzaleztambien