S8-S9

Materia	Diseño de algoritmos
Fecha	@August 24, 2023 → August 25, 2023

Tipos de complejidad

O(1) Constante

$T(n) \in O(1)$

$T(n) \leq c$

Es comun en algoritmos secuenciales (sin ciclos)

Si tu tiempo de ejecución esta en términos de O(1)

Significa que tu tiempo esta acotado por una constante.

O($log\, n$) → o $log\, n$ → o (logarítmica)

Es común cuando tenemos ciclos o recursividad

O(n) Lineal

Es común en ciclos (solo cuando hay uno, sin anidamiento)

O($nlog\, n$) → $n - log - n$

Es común en algoritmos de ordenamiento

O(n^k) → Polinomial

Para casos especiales

$k = 2$ → es cuadrático

$k = 3$ → cúbico

$k > 3$ → Polinomial

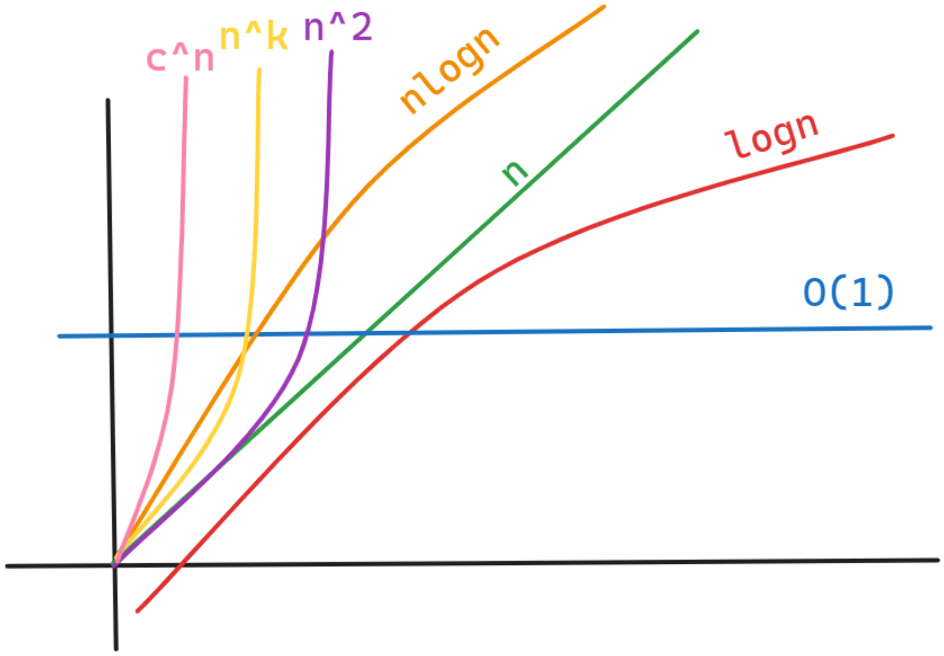
Ciclos anidados

hay un ciclo dentro de un ciclo

Hay un ciclo dentro de un ciclo dentro de un ciclo

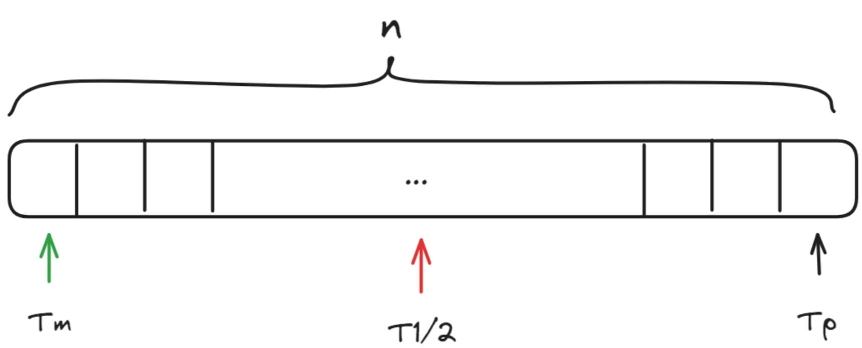
O(c^n) → Exponencial

Ciclos más comunes



Cálculo de $T(n)$

- Se desea que el mejor caso sea $T_p(n) \in O(1)$
- Y que el mejor caso $T_m(n)$
- Caso medio $T_{1/2}(n)$



La complejidad de un algoritmo siempre se define en términos de $T_p(n)$

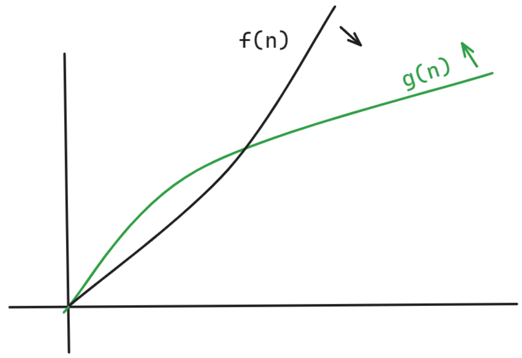
$\rightarrow T_p(n) = T(n) \in O(f(n))$

La complejidad del mejor caso

$\rightarrow T_m(n) \in \Omega(g(n))$

La complejidad de orden exacto

→ Si $T_{1/2}(n) \in \theta(h(n))$



El mejor caso esta por arriba de g(n) (peor caso del mejor caso), el peor caso debajo de f(n)

Operaciones elementales

- Operaciones Aritméticas “básicas”
  - Suma/restas
  - Producto/division → en algunos libros no se consideran como operaciones elementales
  - Asignaciones

    Colocar un dato en algún registro (A ← 8, A← B)
  - Llamadas o retornos de funciones o procedimientos
  - Comparación lógica → no depende de los datos esta acotada

    `if(x==8)`

    `else`
  - Accesos a estructuras

    Arreglos, listas, matrices

Ejemplos de como calcular $T(n)$

Calcular T(n)

```
a = a+1;
b[2] = a+10;
if(a==8)
  b[8] = 10;
else
  a=10;
```

¿Cuantas operaciones elementales se realizan?

`a =` → asignación(1) , `a + 1` → suma(2)

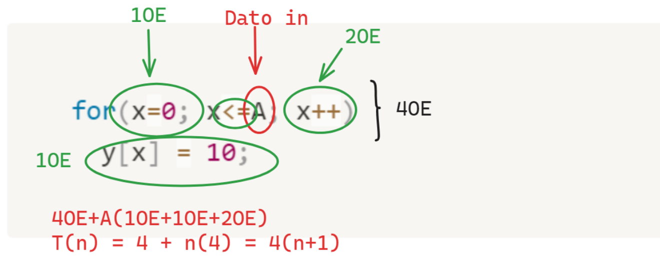
`b[2] =` → asignación y acceso(3), `a+10` → suma(4)

`if` → comparación(5), `b[8]=` → asignación y acceso(6)

Son 6 operaciones elementales

El orden de ejecución es O(1) → $T(n) \in O(1)$

```
for(x=0; x<=A; x++)
  y[x] = 10;
```



$T(n) = 4 + n(4) = 4(n+1) \in O(n)$

@August 25, 2023

Calculo de T(n)

n → se conoce como ejemplo o tamaño del caso

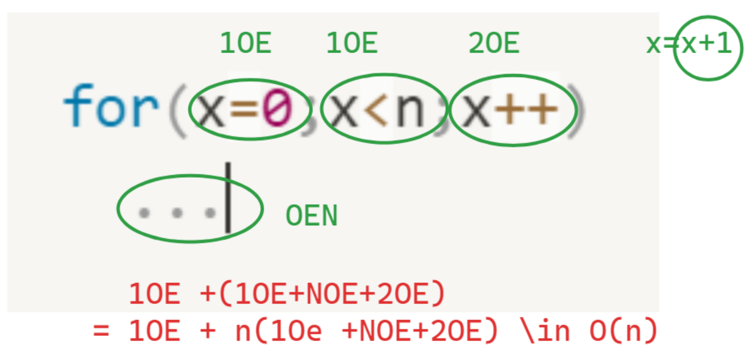
Se refiere al ejemplo que queremos analizar

n puede ser un espacio de donde vamos a tomar los datos o un valor (en caso de ser un espacio, como una arreglo, n es un espacio que describe ese espacio, o una base)

Ejem. Grafo → n = nodos o enlaces

```
for(x=0;x<n;x++)
  ...
```

Es de Orden n → $\in O(n)$



```
if(condición sencilla) then
  N OE
end
```

```
if(condición sencilla) then
  N_1 OE
else
```

```
  N_2 OE
end
```

Regla de composición secuencial

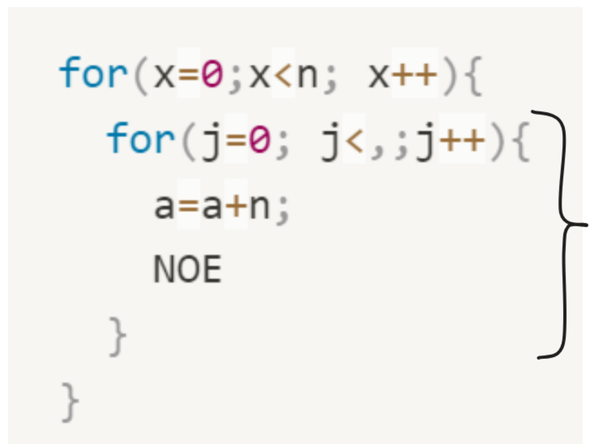
Si tenemos dos secuencias (2 bloques secuenciales de un algoritmo) P_1 y P_2 y sus tiempos son t_1 y t_2, el tiempo será $\theta(max(t_1, t_2))$ → (tomas el mayor de ambos)

```
if(condición sencilla) then
  t_1(n)
else
  t_2(n)
end
```

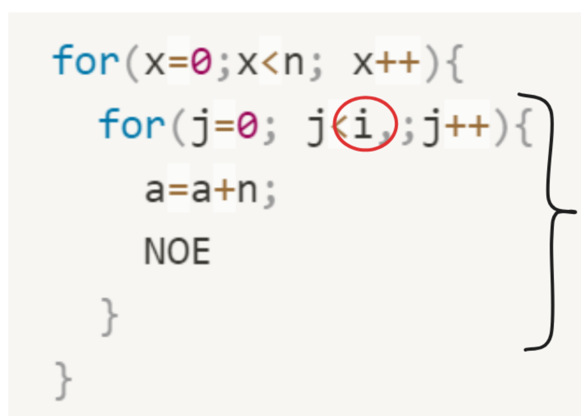
```
for(x=0;x<2n; x++){
  for(j=0; j<n,;j++){
    a=a+n;
    NOE
  }
}
```

```
for(x=0;x<kn; x++){
  for(j=0; j<n,;j++){
    a=a+n;
    NOE
  }
}
```

🍇 O → ≤ o → <
Ω → ≥ ω → >
θ →



si m no depende de n, entonces lo encerrado es de orden constante O(1)



Recursividad, i depende de n