.ECCION 3.- CALCULO de la EFICIENCIA de UN CÓDIGO

- RECURSOS MATEMÁTICOS NECESARIOS

- Propiedades de los logaritmos

- $\log_b x \cdot y = \log_b x + \log_b y$
- $\log_b x/y = \log_b x - \log_b y$
- $\log_b x^y = y \cdot \log_b x$
- $\log_a x = \frac{\log_b x}{\log_b a}$

- Propiedades de las exponenciales

- $a^{xy} = (a^x)^y$
- $a^{x-y} = \frac{a^x}{a^y}$
- $a^{x+y} = a^x a^y$
- $b = a^{\log_a b}$
- $b^x = a^{x \cdot \log_a b}$
- $a^{\log_b n} = n^{\log_b a}$ (*)

$\lfloor x \rfloor$: representa al mayor entero menor o igual que x

$\lceil x \rceil$: representa el menor entero mayor o igual que x

SUMATORIAS

- $\sum_{i=s}^{t} f(i) = f(s) + f(s+1) + \dots + f(t)$

SUMA $\sum_{i=0}^{n} 1 = n$

- PROGRESION ARITMÉTICA $f(i) = i$

$$\sum_{i=0}^{n} i = 0+1+2+\dots+n = n \cdot \frac{(n+1)}{2}$$

- PROGRESION GEOMÉTRICA $f(i) = a^i$

$$\sum_{i=0}^{n} a^i = a^0 + a^1 + \dots + a^n = \frac{a^{n+1} - a^0}{a-1}$$

- SUMA de CUADRADOS

$$\sum_{i=0}^{n} i^2 = \frac{n \cdot (n+1)(2n+1)}{6}$$

- Normas para obtener la eficiencia de un código

- Operacion Elemental $\longrightarrow O(1)$
  - Declaraciones
    `int a,b;` $\longrightarrow O(1)$
  - Asignaciones
    `a=b` $\longrightarrow O(1)$
  - Comparaciones simples
    `if (a<b)` $\longrightarrow O(1)$
  - Operaciones Antmética (+,-,*,/)
    `a*b` $\longrightarrow O(1)$
  - Entrada y salida de un dato basico
    `cin>>a;` $O(1)$ `cout<<b;` $O(1)$

REGLA de la SUMA

Sean dos trozos de codigo independientes (C_1, C_2) con eficiencia $T_1(n)$ y $T_2(n)$. Entonces la eficiencia del codigo union es:

$T_1(n) \in O(f(n))$, $T_2(n) \in O(g(n))$ $\Big\}$ $T_1(n) + T_2(n) \in O(\max(f(n), g(n)))$

Ej

```
int a=5, b=100;
if (a<b)
    cout b;
else
    for (int i=0; i<n; i++)
        v[i]=a;
```

$C_1 \rightarrow T_1(n) \in O(1)$ (if: $O(1)$, $O(1)$; else: $O(1)$)

$T_2(n) \in O(n)$ (C_2: for: $O(1)$; $v[i]=a$: $O(1)$ $O(1)$)

(2)

LECCION 3 continuacion

REGLA del PRODUCTO

Sean dos trozos de código dependientes con tiempo de ejecucion $T_1(n)$ y $T_2(n)$
De manera que el código total tendrá eficiencia

$$\left.\begin{array}{l} T_1(n) \in O(f(n)) \\ T_2(n) \in O(g(n)) \end{array}\right\} T_1(n)\cdot T_2(n) \in O(f(n)\cdot g(n))$$

Ejemplo: Bucles anidados, o Bucles simples

BUCLE SIMPLE

```
for (int i=0; i<n; i++)
    A[i]=0
```

suma evalua

$$1 + 1 + \left((1+1) + (1+1)\right)n \simeq n \in O(n)$$

En la práctica, inicialización del for, primera comparación se contabiliza como 1 más el cuerpo por el nº de veces que se haga

$$1 + n\cdot 1 \simeq n \in O(n)$$

BUCLE ANIDADO

```
for (int i=0; i<n; i++) — O(n)
    for (int j=0; j<n; j++) — O(n)
        A[i][j]=0; — O(1)
```

$O(n\cdot n) = O(n^2)$

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} 1 = \sum_{i=0}^{n-1} n = n\cdot n \in O(n^2)$$

$n\cdot(\underbrace{1+1+1+\dots+1}_{n})$

SENTENCIAS IF-ELSE

```
if (A[0][0]==0){
    for(int i=0; i<n; i++)
        for(int j; j<n; j++)
            A[i][j]=0
}
else
    for (int k=0; k<n; k++)
        A[k][k]=0
```

$O(n^2)$ $O(n)$

$O(\max(n^2, n)) = O(n^2)$

MAS SOBRE BUCLES

1) int s=0; — O(1)
2) for (int i=0; i<n; i++)
3) for (int j=i; j<n; j++)
4) s=i+j;

$$\sum_{i=0}^{n-1}\sum_{j=i}^{n-1} 1 = \sum_{i=0}^{n-1}(n-i) =$$

$$\sum_{i=0}^{n-1} n - \sum_{i=0}^{n-1} i = n\cdot(n) - \frac{n}{2}(n$$

$$= \frac{n^2}{2} \dots \in O(n^2)$$

LECCION 3

- Bucles: el incremento del contador

* `for (int i=0; i<n; i+=2) v[i]=0` $\Big] \sum_{i=0}^{n/2} 1 = \frac{n}{2} + 1 \in O(n)$

son iguales

* `for (int i=1; i≤n; i*=2) v[i]=0` $\Big] \sum_{i=0}^{\log_2(n)} 1 = \underbrace{1+1+1\ldots+1}_{\log_2(n)} = \log_2(n) \in O(\log_2(n))$

* `for (int i=n; i≥1; i/=2) v[i]=0` $\Big] \sum_{i=0}^{\log_2(n)} 1 = \log_2(n) \in O(\log_2(n))$

- El logaritmo significa: cuantas veces puedo dividir un nº (en base 2) por 2

```
for (int i=0; i<n; i++)
    for (int j=0; j<n; j*=2)
        v[i][j]=0
```

$\Big] \sum_{i=0}^{n-1} \sum_{j=0}^{\log_2(n)} 1 = \sum_{i=0}^{n-1} \log_2(n) = n \cdot \log_2(n) \in \Theta(n \cdot \log_2(n))$

$O(\log_2(n)) < O(n) < O(n \cdot \log_2(n)) < O(n^2)$