



UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la Computación e Inteligencia Artificial

Reto 1: Eficiencia

J. Fdez-Valdivia

Dpto. Ciencias de la Computación e Inteligencia Artificial
E.T.S. de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Estructuras de Datos

Grado en Ingeniería Informática
Doble Grado en Ingeniería Informática y Matemáticas
Doble Grado en Ingeniería Informática y ADE

1.- Usando la **notación O**, determinar la eficiencia de las siguientes funciones:

(a)

```
void eficiencia1(int n)
{
    int x=0; int i,j,k; _____ O(1)
    for(i=1; i<=n; i+=4)
        for(j=1; j<=n; j+=[n/4]) _____ Se repite 4 veces
            for(k=1; k<=n; k*=2)
                x++; _____ O(1)
}
```

O(n·logn)

(b)

```
int eficiencia2 (bool existe)
{
    int sum2=0; int k,j,n;

    if (existe)
        for(k=1; k<=n; k*=2)
            for(j=1; j<=k; j++)
                sum2++;
    else
        for(k=1; k<=n; k*=2)
            for(j=1; j<=n; j++)
                sum2++; _____ O(1)
    return sum2;
}
```

O(n·logn)

Es claro que en la parte if de la función se hacen menos operaciones que en la parte else, por lo que la eficiencia total será la de la parte else

(c)

```
void eficiencia3 (int n)
{
    int j; int i=1; int x=0; O(1)
    do{
        j=1; O(1)
        while (j <= n){
            j=j*2; O(1)
            x++; O(1)
        }
        i++; O(1)
    }while (i<=n); O(n)
}
```

O(n·logn)

```
void eficiencia4 (int n)
{
    int j; int i=2; int x=0; O(1)
    do{
        j=1; O(1)
        while (j <= i){
            j=j*2; O(1)
            x++; O(1)
        }
        i++; O(1)
    }while (i<=n);
}
```

O(log n!)

log2 + log3 + ... + logn = log n!

Más rápido que el de la izquierda
 ↙

2.- Considerar el siguiente segmento de código con el que se pretende buscar un entero **x** en una lista de enteros **L** de tamaño **n** (el bucle **for** se ejecuta **n veces**):

```
void eliminar (Lista L, int x)
{
    int aux, p;
    for (p=primero(L); p!=fin(L);)
    {
        aux=elemento (p,L);
        if (aux==x)
            borrar (p,L);
        else p++;
    }
}
```

Analizar la eficiencia de la función **eliminar** si:

(a) **primero** es $O(1)$ y **fin**, **elemento** y **borrar** son $O(n)$. ¿Cómo mejorarías esa eficiencia con un ligero cambio en el código?

(b) **primero**, **elemento** y **borrar** son $O(1)$ y **fin** es $O(n)$. ¿Cómo mejorarías esa eficiencia con un ligero cambio en el código?

(c) **todas las funciones son $O(1)$** . ¿Puede en ese caso mejorarse la eficiencia con un ligero cambio en el código?

Consideraciones:

1.- El reto es **individual**

2.- la solución deberá entregarse obligatoriamente en un fichero pdf (se sugiere como nombre reto1.pdf)

3.- Si la solución es correcta, se puntuará con 0.2 para la evaluación continua

4.- El plazo límite de entrega es el 3 de Octubre a las 23.55h