TEMA 1: EFICIENCIA DE LOS ALGORITMOS

2. Análisis por casos

- **El coste de un algoritmo será en general una función dependiente de la talla del problema. Pero también puede depender de la diferente configuración de la entrada del problema.**

DEF. (Instancia): Una instancia de un problema corresponde a todas las configuraciones diferentes de la entrada, de una talla determinada, que dan lugar al mismo comportamiento del algoritmo.

2. Análisis por casos

Ejemplo 1:

```
ALGORITMO sumar vector
DATOS:              a:vector[1..n] de N
RESULTADO:          s:N
AUXILIAR:           i:1..n
METODO:
   para i ← 1 hasta n hacer                    //n + 1//
     s ← s + a[i]                              //n//
   fpara
fsumar vector
```

- El algoritmo sumar vector tiene una sola instancia. Sea como sea el vector de entrada, el coste del algoritmo será siempre el mismo (T(n) = 2n + 1).

2. Análisis por casos

Ejemplo 2:

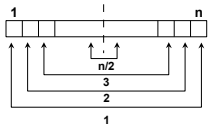
```
ALGORITMO busqueda lineal
DATOS:              a:vector[1..n] de N
                    x: N
RESULTADO:          m:1..n
METODO:
  m ← 1                                        //1//
  mientras m ≤ n ∧ a[m] ≠ x hacer               //1,n+1//
    m ← m+1                                    //0,n//
  fmientras
fbusqueda lineal
```

- El coste del algoritmo búsqueda lineal va a depender de la configuración del vector y del elemento que estemos buscando. Si el elemento está en la primera posición, el coste es **T(n)=2**. Si el elemento no está, el coste es **T(n)=2n + 2**.
- En total hay **n+1 instancias**.
- ¿Los problemas de buscar el valor 1 en el vector {1,3,5,9} y en el {1,9,2,1} son instancias distintas?

2. Análisis por casos

Ejemplo 3:

- Un algoritmo que determina si un vector binario (0,1) de n elementos, siendo n par, es capicúa o no.



- **¿Talla del problema?**
- **¿Número de instancias?**

2. Análisis por casos

DEF:

- **Coste en el caso mejor ($T^m(n)$):** Corresponde al coste de la instancia del problema que presenta el menor coste.
- **Coste en el caso peor ($T^p(n)$):** Coste de la instancia del problema que presenta el mayor coste.
- **Coste en el caso medio ($T^\mu(n)$):** Corresponde a la media de los costes de todas las instancias del problema. En el caso más general se necesita conocer la probabilidad de que se dé cada una de las instancias (media ponderada).

2. Análisis por casos

Formalizándolo matemáticamente:

Sea A un algoritmo cuya talla viene dada por los parámetros $n = (n_1,...,n_k)$ $(n \in N^k)$. χ el dominio de datos de entrada para A.

Sea $\chi_n = \{x \in \chi : |x| = n\}$ el conjunto de todas las entradas de talla n.

Sea $T(n; x_n)$ el coste asociado al suceso $x_n \in \chi_n$

Sea $p(x_n)$ la probabilidad de ocurrencia de x_n.

Coste mejor: $T_A^m(n) = \min_{\forall x_n \in \chi_n} T_A(n; x_n)$

Coste peor: $T_A^p(n) = \max_{\forall x_n \in \chi_n} T_A(n; x_n)$

Coste Medio: $T_A^\mu(n) = \sum_{\forall x_n \in \chi_n} p(x_n) T_A(n; x_n)$

2. Análisis por casos

Ejemplo 1:

Análisis en el caso mejor, peor y medio suponiendo que todas las instancias son equiprobables.

```
ALGORITMO mínimo // Busca la posición del mínimo de un vector
DATOS:                a:vector[1..n] de N
RESULTADO:            m:1..n
AUXILIAR:             i:1..n
METODO:
1. m ← 1
2. para i ← 2 hasta n hacer
3.   si a[i] < a[m] entonces
4.           m ← i
5.   fsi
6. fpara
fminimo
```


2. Análisis por casos

Ejemplo 2: Análisis en el caso mejor, peor y medio.

```
ALGORITMO busqueda_lineal
DATOS:                a:vector[1..n] de N
                      x: N
RESULTADO:            m:1..n
METODO:
  m ← 1                                          //1//
  mientras m ≤ n ∧ a[m] ≠ x hacer                 //1,n+1//
    m ← m+1                                      //0,n//
  fmientras
fbusqueda_lineal
```

1. **Suponer que el elemento se encuentra siempre en el vector y que todas las posiciones son equiprobables.**
2. **Suponer que la probabilidad de que el elemento no exista es del 50 %**
3. **Suponer que el vector es de enteros de 16 bits y que cada número tiene la misma probabilidad de aparecer en una determinada posición del vector.**

2. Análisis por casos

Ejemplo 3: Análisis en el caso mejor, peor y medio.

```
Algoritmo simetrica(a: vector[1..n][1..n] de nat) dev s: nat
sim ← cierto
s ← 0
si a[1, n] ≠ a[n, 1] entonces
    para i ← 1 hasta n
            para j ← 1 hasta i
                    s ← s + a[i, j]
            fpara
    fpara
Fsi
```

1. **Suponer que la probabilidad de que a[1, n] sea igual a a[n, 1] es del 25 % .**