



# Taller de programación competitiva

*An amateur approach*

---

Ignacio Ballesteros González

ballesteros@acm.org

Samuel García Haro

samgh96@gmail.com

20 de febrero de 2018

1. Introduction
2. Estructuras de Datos básicas
3. Entrada/Salida
4. Métodos algorítmicos

# Introduction

---

# Estructuras de Datos básicas

---

## Entrada/Salida

---

- La entrada nos da pistas de la modelización del problema.

- La entrada nos da pistas de la modelización del problema.
- Es importante **no perder más tiempo del necesario con la E/S.**

- La entrada nos da pistas de la modelización del problema.
- Es importante **no perder más tiempo del necesario con la E/S.**
- Se recomienda llevarla apuntada o en su defecto memorizarla.



4 -> número de entradas

4 -> número de entradas

4 -> tamaño de vector

4 -> número de entradas

4 -> tamaño de vector

1 1 1 2 -> elementos de vector

4 -> número de entradas

4 -> tamaño de vector

1 1 1 2 -> elementos de vector

2

1 1

5

1 1 2 2 3

# Entrada/Salida: Procesador genérico

**Data:** num\_entradas, tam\_vector, vector

declarar variables;

**while** *num\_entradas* > 0 **do**

    leer tam;

    inicializar vector;

**for** *i* **in to** *tam\_vector* **do**

        vector[i] = leer\_entero;

**end**

    procesar;

    num\_entradas -= 1;

**end**

## Entrada/Salida: Ejemplo

```
public static void main(String[] args){
    int n, len;
    int [] arr;
    FastReader fr = new FastReader();

    n = fr.nextInt();
    while (n > 0){
        len = fr.nextInt();
        arr = new int[len];
        for (int i = 0; i < len; i++)
            arr[i] = fr.nextInt();

        System.out.println(findMajority(arr));
        n--;
    }
}
```

# Métodos algorítmicos

---

## Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.



## Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.
- Descompone problemas en subproblemas de solución más sencilla.

## Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.
- Descompone problemas en subproblemas de solución más sencilla.
- Se distinguen cuatro pasos:

# Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.
- Descompone problemas en subproblemas de solución más sencilla.
- Se distinguen cuatro pasos:
  - Hallar el caso base del problema.

# Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.
- Descompone problemas en subproblemas de solución más sencilla.
- Se distinguen cuatro pasos:
  - Hallar el caso base del problema.
  - Dividir la estructura de datos hasta encontrar el caso base.

# Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.
- Descompone problemas en subproblemas de solución más sencilla.
- Se distinguen cuatro pasos:
  - Hallar el caso base del problema.
  - Dividir la estructura de datos hasta encontrar el caso base.
  - Aplicar la solución al caso base.

# Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.
- Descompone problemas en subproblemas de solución más sencilla.
- Se distinguen cuatro pasos:
  - Hallar el caso base del problema.
  - Dividir la estructura de datos hasta encontrar el caso base.
  - Aplicar la solución al caso base.
  - Recomponer la estructura de datos ya resuelta.

# Divide y vencerás: Concepto

- Patrón de diseño de algoritmos de carácter recursivo.
- Descompone problemas en subproblemas de solución más sencilla.
- Se distinguen cuatro pasos:
  - Hallar el caso base del problema.
  - Dividir la estructura de datos hasta encontrar el caso base.
  - Aplicar la solución al caso base.
  - Recomponer la estructura de datos ya resuelta.
- Ejemplos de este método: Mergesort, quicksort, búsqueda binaria...

## Método voraz: Concepto

- Este método algorítmico se basa en la elección de soluciones locales óptimas con el fin de obtener soluciones globales óptimas.



## Método voraz: Concepto

- Este método algorítmico se basa en la elección de soluciones locales óptimas con el fin de obtener soluciones globales óptimas.
- Útil en problemas de optimización pero difícil de probar su correctitud.

## Método voraz: Concepto

- Este método algorítmico se basa en la elección de soluciones locales óptimas con el fin de obtener soluciones globales óptimas.
- Útil en problemas de optimización pero difícil de probar su correctitud.
- Consiste en los siguientes fundamentos:

## Método voraz: Concepto

- Este método algorítmico se basa en la elección de soluciones locales óptimas con el fin de obtener soluciones globales óptimas.
- Útil en problemas de optimización pero difícil de probar su correctitud.
- Consiste en los siguientes fundamentos:
  - Disponemos de un conjunto de entrada (candidatos) y de otro de salida.

## Método voraz: Concepto

- Este método algorítmico se basa en la elección de soluciones locales óptimas con el fin de obtener soluciones globales óptimas.
- Útil en problemas de optimización pero difícil de probar su correctitud.
- Consiste en los siguientes fundamentos:
  - Disponemos de un conjunto de entrada (candidatos) y de otro de salida.
  - Mediante heurísticas decidimos qué elemento entra en el conjunto de salida (y lo eliminamos de los candidatos).

## Método voraz: Concepto

- Este método algorítmico se basa en la elección de soluciones locales óptimas con el fin de obtener soluciones globales óptimas.
- Útil en problemas de optimización pero difícil de probar su correctitud.
- Consiste en los siguientes fundamentos:
  - Disponemos de un conjunto de entrada (candidatos) y de otro de salida.
  - Mediante heurísticas decidimos qué elemento entra en el conjunto de salida (y lo eliminamos de los candidatos).
  - Para decidir los siguientes elementos los sometemos a un test de factibilidad.

## Método voraz: Concepto

- Este método algorítmico se basa en la elección de soluciones locales óptimas con el fin de obtener soluciones globales óptimas.
- Útil en problemas de optimización pero difícil de probar su correctitud.
- Consiste en los siguientes fundamentos:
  - Disponemos de un conjunto de entrada (candidatos) y de otro de salida.
  - Mediante heurísticas decidimos qué elemento entra en el conjunto de salida (y lo eliminamos de los candidatos).
  - Para decidir los siguientes elementos los sometemos a un test de factibilidad.