



FUNDACIÓN UNIVERSITARIA DE POPAYÁN

Educación de **calidad** con
responsabilidad **social**



FUNDACIÓN
UNIVERSITARIA
DE POPAYÁN

INGENIERIA DE SISTEMAS

Estructura de Datos

Adrian Danilo Astudillo

Mail: adrian.astudillo@docente.fup.edu.co



FUNDACIÓN
UNIVERSITARIA
DE POPAYÁN

Objetivo

Desarrollar programas informáticos que utilizan estructuras de datos para la solución de problemas propios de la ingeniería, integrando técnicas y herramientas de programación orientada a objetos

CONTENIDO

- ✓ Algoritmos de Ordenamiento
 - Introducción
- ✓ Algoritmo de Ordenamiento por Inserción
 - Pasos a Realizar
 - Algoritmo
- ✓ Algoritmo de Ordenamiento por Selección
 - Forma
 - Parámetros

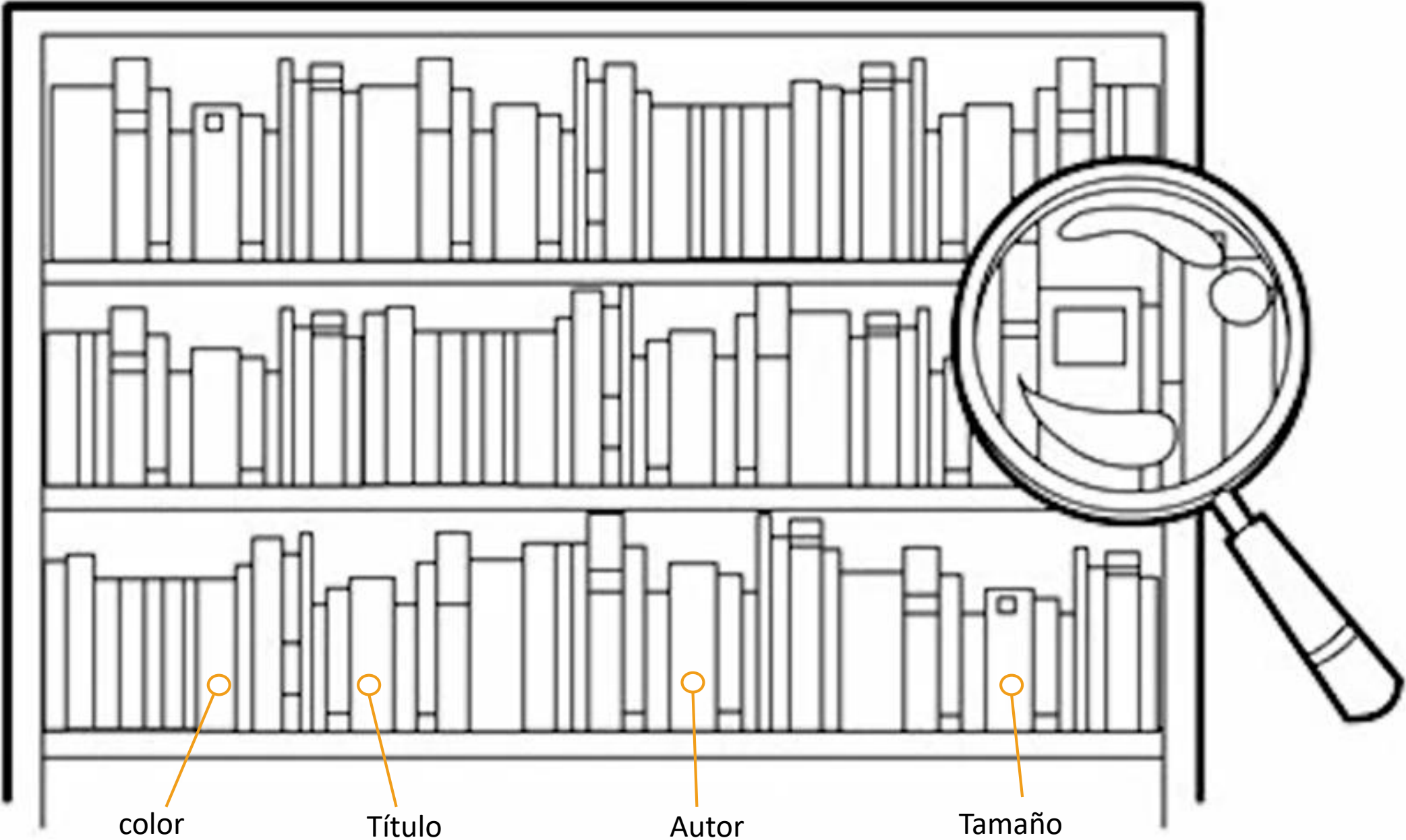


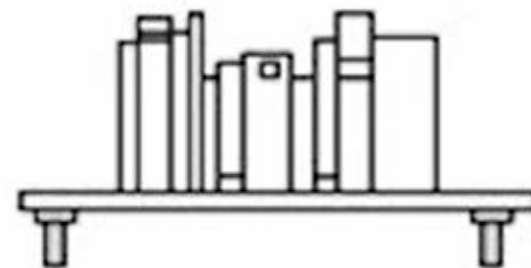
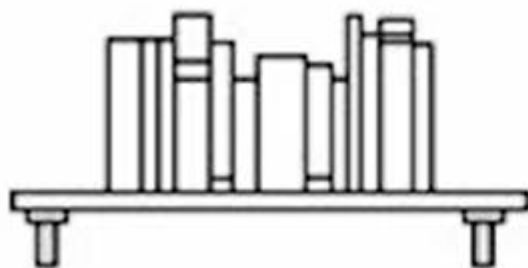
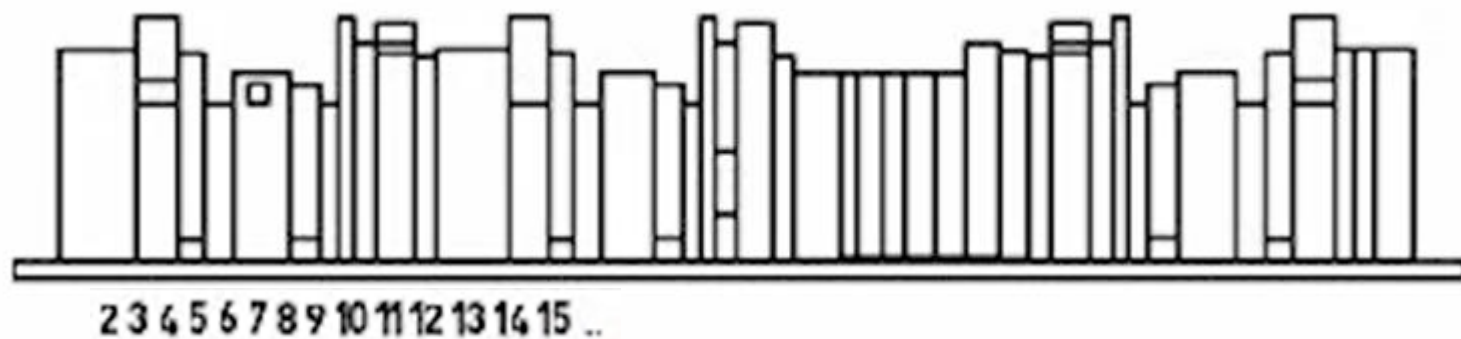
INTRODUCCIÓN

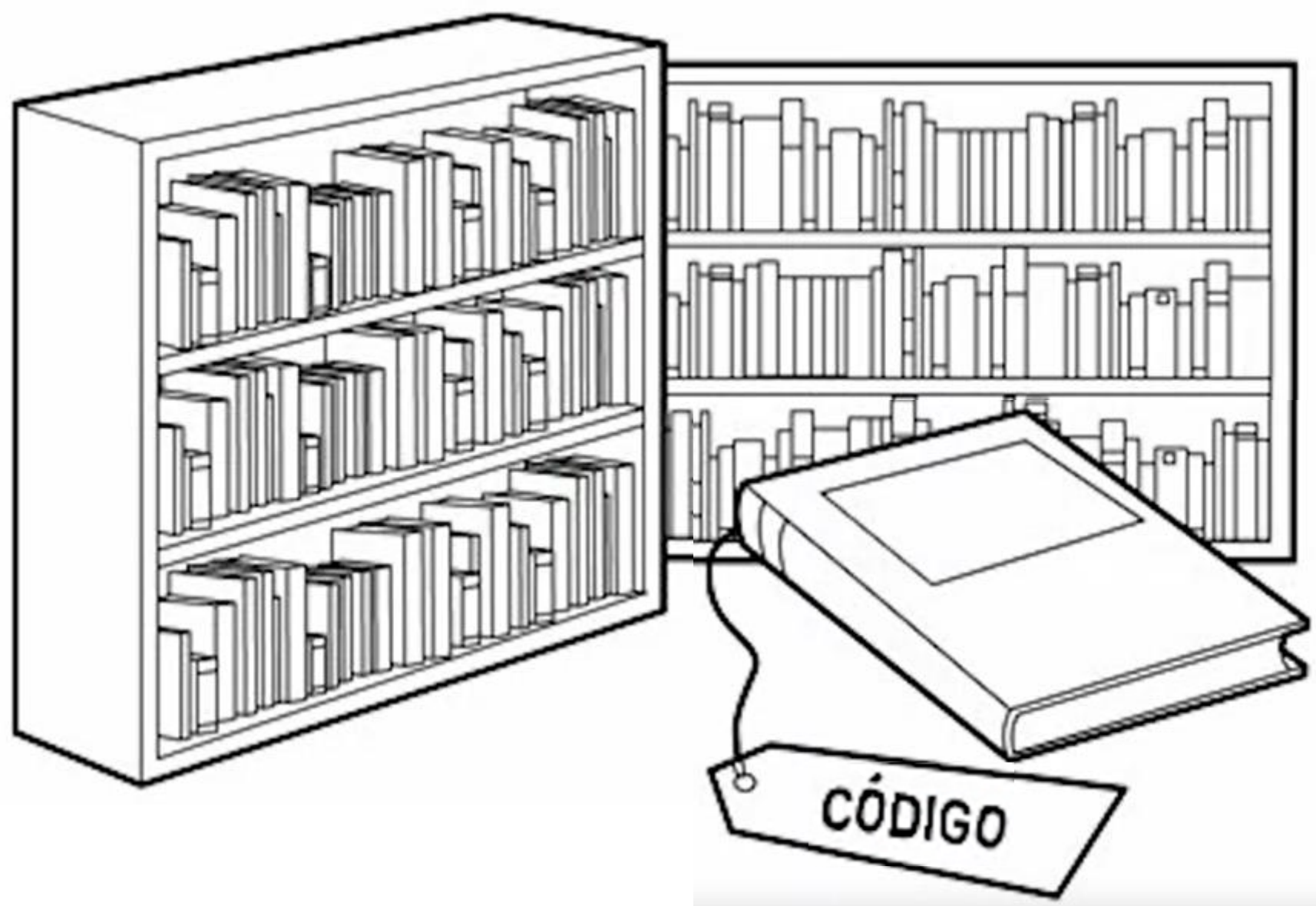
Imaginemos que nos encontramos en nuestra habitación desordenada, imaginemos que somos orgullosos propietarios de una extensa colección de libros. Si la colección es extensa la tarea de buscar en esa estantería sería complicada. Así que el siguiente paso es ordenar los libros

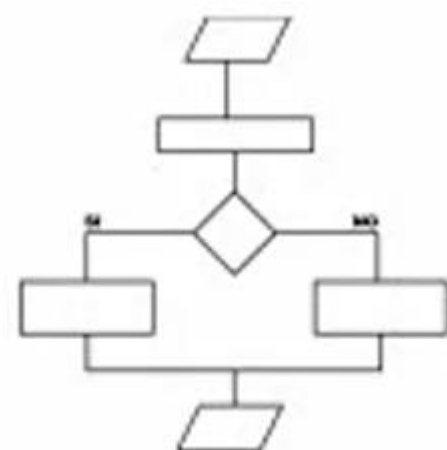


ESTANTERÍA

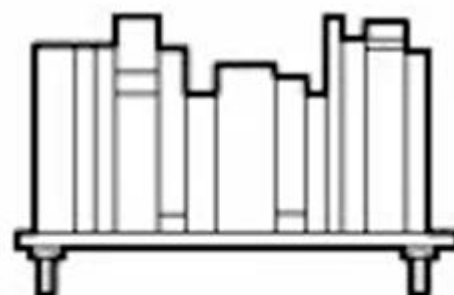




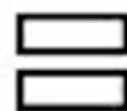




ALGORITMO



ORDENAMIENTO



MUNDO REAL



MUNDO DE LA PROGRAMACIÓN

Por qué Ordenar?

Facilitar Procesos de Búsqueda

Algunos conceptos que solo tienen sentido cuando los elementos están ordenados

- **Mediana**

4. Repetir Hasta Ordenar Todo



CONTENIDO

- ✓ Algoritmos de Ordenamiento
 - ✓ Por qué Ordenar
 - ✓ **Algoritmo de Ordenamiento por Selección**
 - Forma
 - Parámetros
- ✓ Algoritmo de Ordenamiento por Inserción
 - Pasos a Realizar
 - Algoritmo
- ✓ Algoritmo de Ordenamiento por Burbuja
 - Pasos a Realizar
 - Algoritmo



Algoritmo de Ordenamiento por Selección

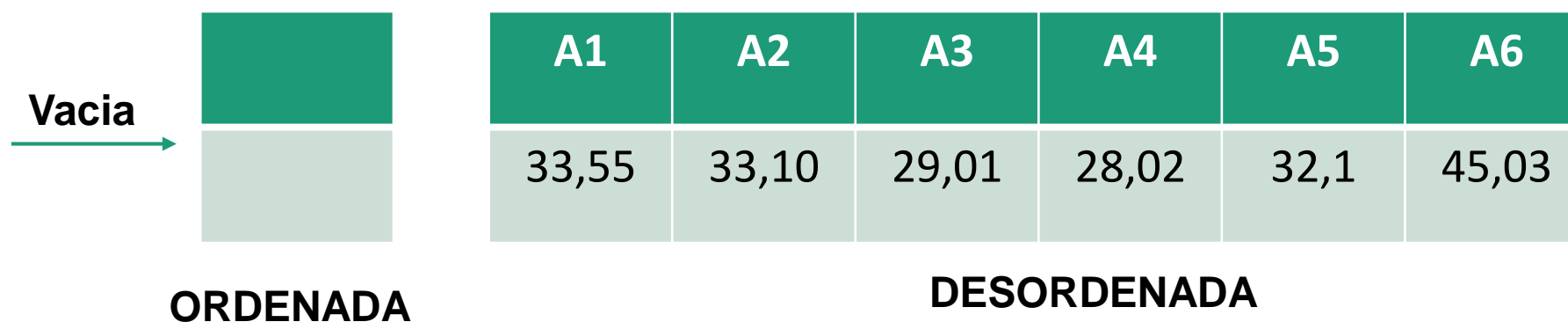
1. Definir una Parte Ordenada
2. Buscar el Menor(o Mayor) de la parte Desordenada
3. Intercambiarlo con el primero de la parte desordenada
4. Repetir Hasta Ordenar Todos

4. Repetir Hasta Ordenar Todo



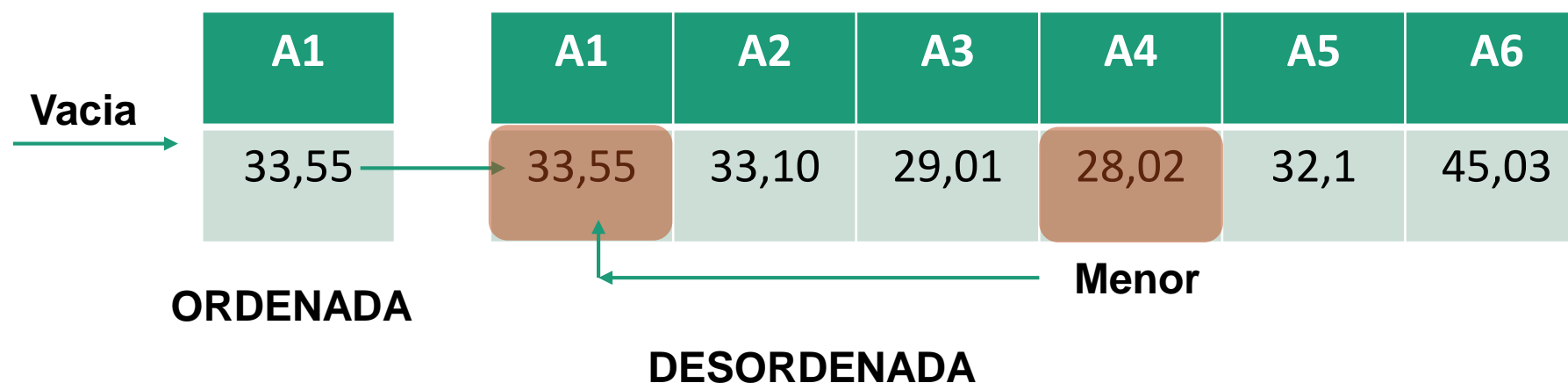
Algoritmo de Ordenamiento por Selección

1. Definir una Parte Ordenada



Algoritmo de Ordenamiento por Selección

2. Definir una Parte Ordenada



Algoritmo de Ordenamiento por Selección

3. Intercambio con el Primero de la parte Desordenada

A1	A2	A3	A4	A5	A6
28,02	33,10	29,01	33,55	32,1	45,03

Menor Intercambio

DESORDENADA

Algoritmo de Ordenamiento por Selección

4. Repetir Hasta Ordenar

A1	A2	A3	A4	A5	A6
28,02	33,10	29,01	33,55	32,1	45,03
↑			↑		
28,02	29,01	33,10	33,55	32,1	45,03
28,02	29,01	32,1	33,55	33,10	45,03
28,02	29,01	32,1	33,10	33,55	45,03

CONTENIDO

- ✓ Algoritmos de Ordenamiento
 - ✓ Por qué Ordenar
- ✓ Algoritmo de Ordenamiento por Selección
 - Forma
 - Parámetros
- ✓ **Algoritmo de Ordenamiento por Inserción**
 - Pasos a Realizar
 - Algoritmo
- ✓ Algoritmo de Ordenamiento por Burbuja
 - Pasos a Realizar
 - Algoritmo

Algoritmo de Ordenamiento por Inserción

1. Definir una Parte Ordenada
2. Tomar el Primero de la Parte Desordenada
3. Agregarlo Ordenadamente en la Parte Ordenada
4. Repetir Hasta Ordenar Todo

Algoritmo de Ordenamiento por Inserción

1. Definir una Parte Ordenada

A1	A2	A3	A4	A5	A6
33,55	33,10	29,01	28,02	32,1	45,03

ORDENADA

DESORDENADA

Algoritmo de Ordenamiento por Inserción

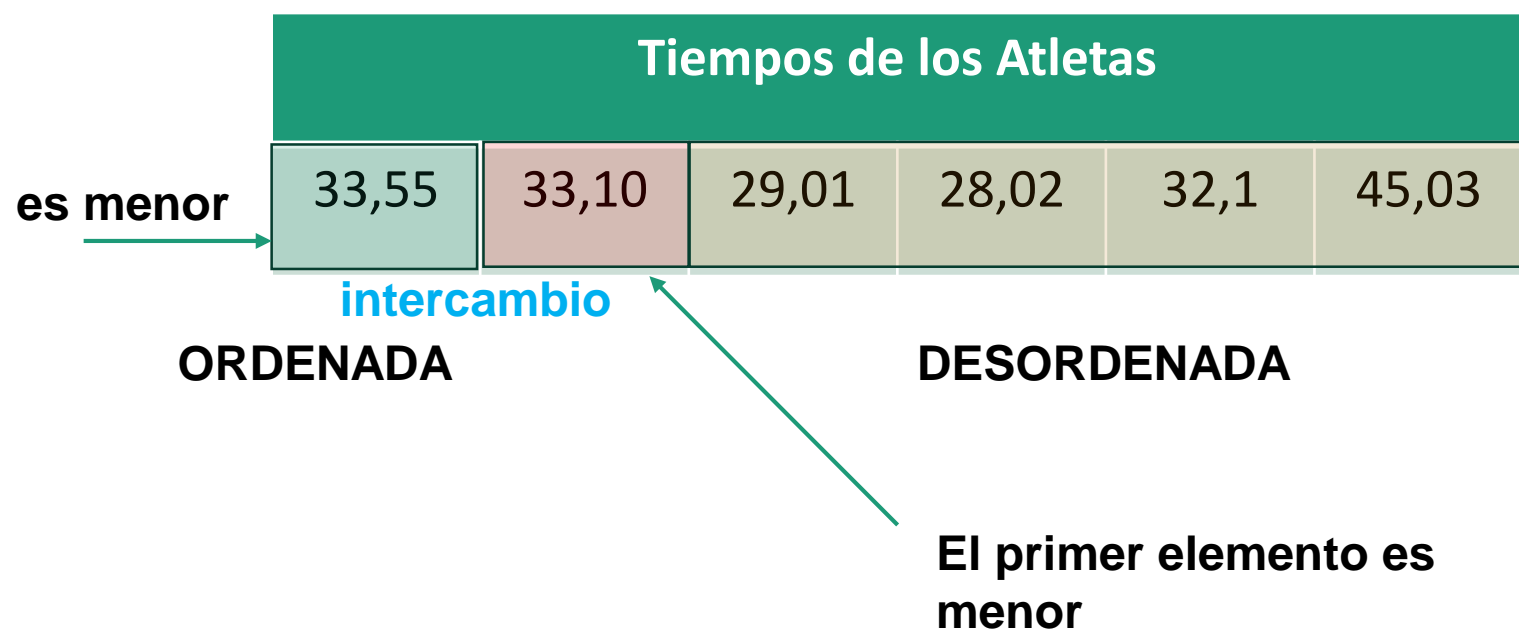
2. Tomar el Primero de la parte Desordenada

A1	A2	A3	A4	A5	A6
33,55	33,10	29,01	28,02	32,1	45,03

ORDENADA elemento DESORDENADA

Algoritmo de Ordenamiento por Inserción

3. Agregarlo ordenadamente en la parte ordenada



Java – Algoritmo de Inserción

```
public class Competencia{
    private ArrayList<Atleta> atletas;

    public void ordenarAtletasPorTiempo() {
        boolean agregado=false;
        for(int j=1; j>0 && !agregado;j--){
            Atleta uno = atletas.get(j);
            Atleta dos = atletas.get(j-1);
            if(uno.getTiempo( )< dos.getTiempo( )){
                atletas.set(j, dos);
                atletas.set(j-1, uno);
            }else
                agregado=true;
        }
    }
}
```

Java – Algoritmo de Inserción

```
public class Competencia{
    private ArrayList<Atleta> atletas;

    public void ordenarAtletasPorTiempo() {
        for(int i=1; i<atletas.size();i++){
            boolean agregado=false;
            for(int j=1; j>0 && !agregado;j--){
                Atleta uno = atletas.get(j);
                Atleta dos = atletas.get(j-1);
                if(uno.getTiempo( )< dos.getTiempo( )){
                    atletas.set(j, dos);
                    atletas.set(j-1, uno);
                }else
                    agregado=true;
            }
        }
    }
}
```

CONTENIDO

- ✓ Algoritmos de Ordenamiento
 - ✓ Por qué Ordenar
- ✓ Algoritmo de Ordenamiento por Selección
 - Forma
 - Parámetros
- ✓ Algoritmo de Ordenamiento por Inserción
 - Pasos a Realizar
 - Algoritmo
- ✓ **Algoritmo de Ordenamiento por Burbuja**
 - Pasos a Realizar
 - Algoritmo

Algoritmo de Ordenamiento Burbuja

1. Definir una Parte Ordenada
2. Tomar un Par de Elementos Consecutivos y Ordenarlos
3. Repetir Hasta Llegar al Final de la Parte Desordenada
4. Repetir Hasta Ordenar Todos

Ordenamiento Burbuja

Ahora que puedes hacer malabarismos con los elementos de los arreglos, es hora de aprender como ordenarlos. Se han inventado muchos algoritmos de clasificación, que difieren mucho en velocidad, así como en complejidad. Vamos a mostrar un algoritmo muy simple, fácil de entender, pero desafortunadamente, tampoco es muy eficiente. Se usa muy raramente, y ciertamente no para listas extensas.

Digamos que una lista se puede ordenar de dos maneras:

Ascendente (o más precisamente, no descendente): si en cada par de elementos adyacentes, el primer elemento no es mayor que el segundo.

Descendente (o más precisamente, no ascendente): si en cada par de elementos adyacentes, el primer elemento no es menor que el segundo.

En las siguientes secciones, ordenaremos la lista en orden ascendente, de modo que los números se ordenen de menor a mayor.

Aquí está la lista:

8	10	6	2	4
---	----	---	---	---

Ordenamiento Burbuja

Intentaremos utilizar el siguiente enfoque: tomaremos el primer y el segundo elemento y los compararemos; si determinamos que están en el orden incorrecto (es decir, el primero es mayor que el segundo), los intercambiaremos; Si su orden es válido, no haremos nada. Un vistazo a nuestra arreglo confirma lo último: los elementos 01 y 02 están en el orden correcto, así como $8 < 10$.

8	10	6	2	4
---	----	---	---	---

Ahora observa el segundo y el tercer elemento. Están en las posiciones equivocadas. Tenemos que intercambiarlos:

8	6	10	2	4
---	---	----	---	---

Ordenamiento Burbuja

Vamos más allá y observemos los elementos tercero y cuarto. Una vez más, esto no es lo que se supone que es. Tenemos que intercambiarlos:

8	6	2	10	4
---	---	---	----	---

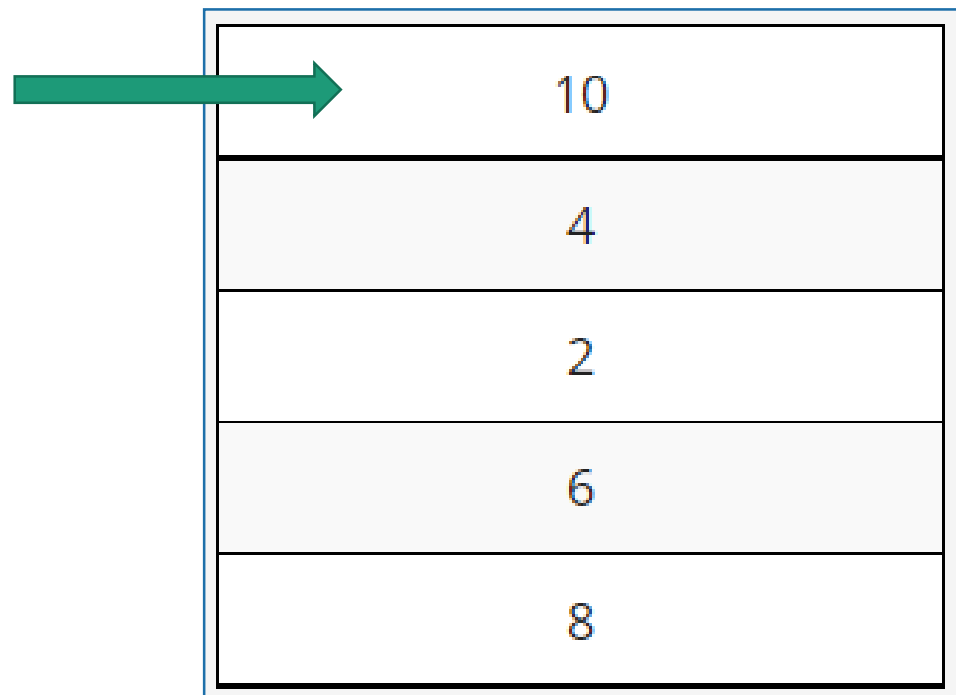
Ahora comprobemos los elementos cuarto y quinto. Si, ellos también están en las posiciones equivocadas. Ocurre otro intercambio:

8	6	2	4	10
---	---	---	---	----

Ordenamiento Burbuja

El primer paso a través del arreglo ya está terminado. Todavía estamos lejos de terminar nuestro trabajo, pero algo curioso ha sucedido mientras tanto. El elemento más grande, 10, ya ha llegado al final de la lista. Ten en cuenta que este es el lugar deseado para él. Todos los elementos restantes forman un lío pintoresco, pero este ya está en su lugar.

Ahora, por un momento, intenta imaginar arreglo de una manera ligeramente diferente, es decir, de esta manera:



10
4
2
6
8

Observa - El 10 está en la parte superior. Podríamos decir que flotó desde el fondo hasta la superficie, al igual que las burbujas en una copa de champán. El método de clasificación deriva su nombre de la misma observación: se denomina ordenamiento de burbuja.

Ordenamiento Burbuja

Ahora comenzamos con el segundo paso a través del arreglo. Miramos el primer y el segundo elemento, es necesario un intercambio:

6	8	2	4	10
---	---	---	---	----

Tiempo para el segundo y tercer elemento: también tenemos que intercambiarlos:

6	2	8	4	10
---	---	---	---	----

Ahora el tercer y cuarto elementos, y la segunda pasada, se completa, ya que 8 ya está en su lugar:

6	2	4	8	10
---	---	---	---	----

Ordenamiento Burbuja

Comenzamos el siguiente pase inmediatamente. Observe atentamente el primer y el segundo elemento: se necesita otro cambio:

2	6	4	8	10
---	---	---	---	----

Ahora 6 necesita ir a su lugar. Cambiamos el segundo y el tercer elemento:

2	4	6	8	10
---	---	---	---	----

La lista ya está ordenada. No tenemos nada más que hacer. Esto es exactamente lo que queremos.

Como puedes ver, la esencia de este algoritmo es simple: comparamos los elementos adyacentes y, al intercambiar algunos de ellos, logramos nuestro objetivo.

Ordenamiento Burbuja – Código Java

```
// Ordenamiento Burbuja Java
public class BubbleSort {
    public static void main(String[] args) {
        int[] numeros = {8, 10, 6, 2, 4};

        for (int i = 0; i < numeros.length - 1; i++) {
            for (int j = 0; j < numeros.length - i - 1; j++) {
                if (numeros[j] > numeros[j + 1]) {
                    // Intercambio de elementos
                    int temp = numeros[j];
                    numeros[j] = numeros[j + 1];
                    numeros[j + 1] = temp;
                }
            }
        }
        System.out.println("Arreglo ordenado:");
        for (int numero : numeros) {
            System.out.print(numero + " ");
        }
    }
}
```

Ordenamiento Burbuja – Explicación

El bucle externo (índice i) controla cuántas veces se repite el proceso de comparación e intercambio. El bucle interno (índice j) recorre el arreglo y compara pares de elementos adyacentes. Si el elemento en la posición j es mayor que el elemento en la posición $j + 1$, se realiza un intercambio.

Con cada iteración del bucle externo, el elemento más grande "flota" hacia la última posición. Después de completar todas las iteraciones, el arreglo estará ordenado de manera ascendente.


Recuerda que, aunque el ordenamiento de burbuja es útil para aprender sobre algoritmos de ordenamiento, en aplicaciones del mundo real, a menudo se prefieren algoritmos más eficientes como el ordenamiento rápido o el ordenamiento de mezcla para conjuntos de datos más grandes.

Algoritmo de Ordenamiento Burbuja

1. Definir una Parte Ordenada
2. Tomar un Par de Elementos Consecutivos y Ordenarlos
3. Repetir Hasta Llegar al Final de la Parte Desordenada
4. Repetir Hasta Ordenar Todos

Algoritmo de Ordenamiento de Burbuja

Ordenar los tiempos de los Atletas una competencia según su tiempo
De menor a Mayor



Atleta

A1	A2	A3	A4	A5	A6
33,55	33,10	29,01	28,02	32,1	45,03

Tiempos

Algoritmo de Ordenamiento de Burbuja

2. Definir una Parte Ordenada

A1	A2	A3	A4	A5	A6
33,55	33,10	29,01	28,02	32,1	45,03

DESORDENADA

A1
33,55

ORDENADA

Java – Ejercicios

Ejercicio 1: Ordenamiento por Selección

Descripción:

- ✓ En una competencia de atletismo, se registran los tiempos de llegada de los corredores en una lista desordenada. Utilizando el algoritmo de selección, tu tarea es ordenar los tiempos de llegada en orden ascendente. Esto te permitirá determinar rápidamente quién fue el corredor más rápido y quiénes ocuparon los siguientes lugares.

Java – Ejercicios

Ejercicio 2: Ordenamiento por Inserción

Descripción:

- ✓ En una biblioteca, se tiene una pila de libros que se recibieron recientemente y necesitan ser ordenados en los estantes. Utilizando el algoritmo de inserción, tu tarea es ordenar los libros alfabéticamente por título y colocarlos en los estantes correspondientes. Este enfoque te permitirá organizar rápidamente la biblioteca y facilitar el acceso a los libros.

Java – Ejercicios

Ejercicio 3: Ordenamiento de Burbuja

Descripción:

En un juego de cartas, se tiene una mano desordenada de cartas que se necesitan organizar para poder jugar. Utilizando el algoritmo de burbuja, tu tarea es ordenar las cartas por valor y palo para que puedas identificar rápidamente las combinaciones y estrategias ganadoras. Este método te permitirá organizar eficientemente tu mano y tomar decisiones informadas durante el juego.



FUNDACIÓN
UNIVERSITARIA
DE POPAYÁN

Gracias