



**Universidad
Tecnológica
del Perú**

Sesión 03

Métodos de Ordenamiento Avanzado & Búsqueda

Unidad I



Universidad
Tecnológica
del Perú

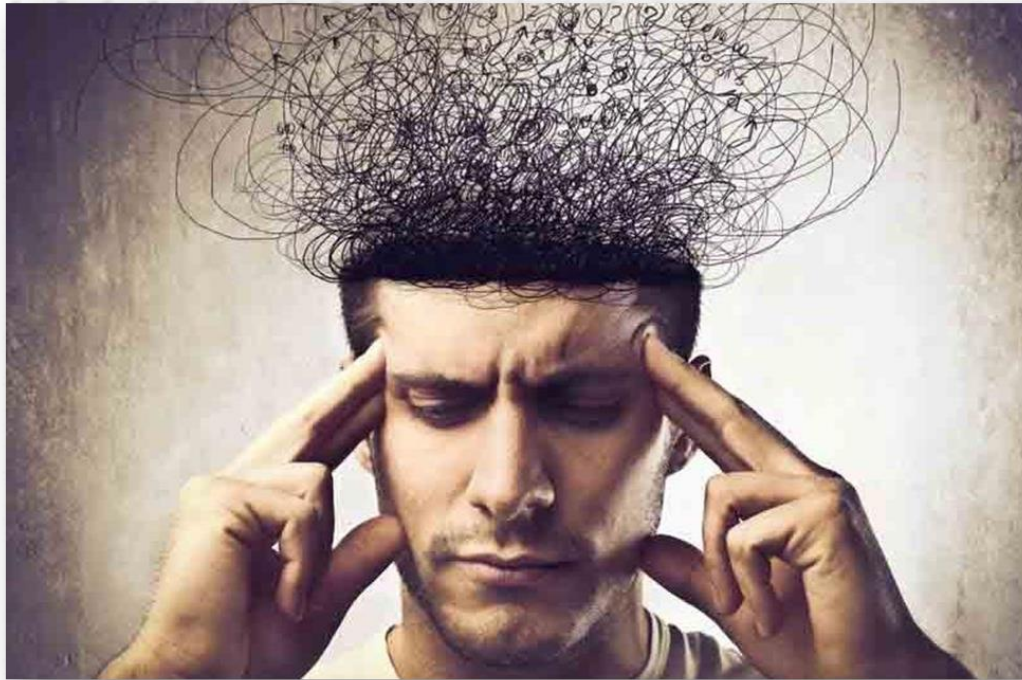
Temario:

- Repaso de algoritmos avanzados de ordenamiento
 - Quick Sort
 - Shell Sort
 - Merge Sort
- Algoritmos de Búsqueda
 - Linear search
 - Binary search
- Práctica

Pautas de trabajo

- Los días que tengamos clases debemos conectarnos a través de Zoom.
- La participación de los estudiantes se dará través del **chat de Zoom**.

Inicio:



`equals` vs `==`

`compareTo`

Merge Sort

Quick Sort

Shell Sort

Utilidad:



Mira la imagen y responde

¿Cuál es el servicio principal de Google?

¿Siempre encuentra resultados de lo que se busca?

¿Qué información devuelve?

¿Cómo podrías buscar información en Java?

Utilidad:

Desarrollar un diccionario en Java



Mira la imagen y responde

¿Qué actividad se está realizando en la imagen?

¿Los datos mostrados, deben estar en orden?

¿Qué sucede si una palabra no se encuentra en el **diccionario**?

¿En que te ayudaría en tu futuro profesional desarrollar **aplicaciones que te permitan buscar información**?

Logro de la sesión:

“Al finalizar la sesión, el estudiante resolverá problemas propuestos utilizando algoritmos ordenamiento avanzado y de búsqueda utilizando el lenguaje Java”.





Algoritmos de ordenamiento avanzados

Métodos de ordenamiento

Merge Sort

- Sigue el enfoque “divide y vencerás”
- Subdivide el array en 2 subarrays de $n/2$ elementos cada uno de manera recursiva
- Una vez que la división de arrays ya no puede continuar, se realiza el proceso de “fusión” (merge)
- Su complejidad es $O(n \log n)$

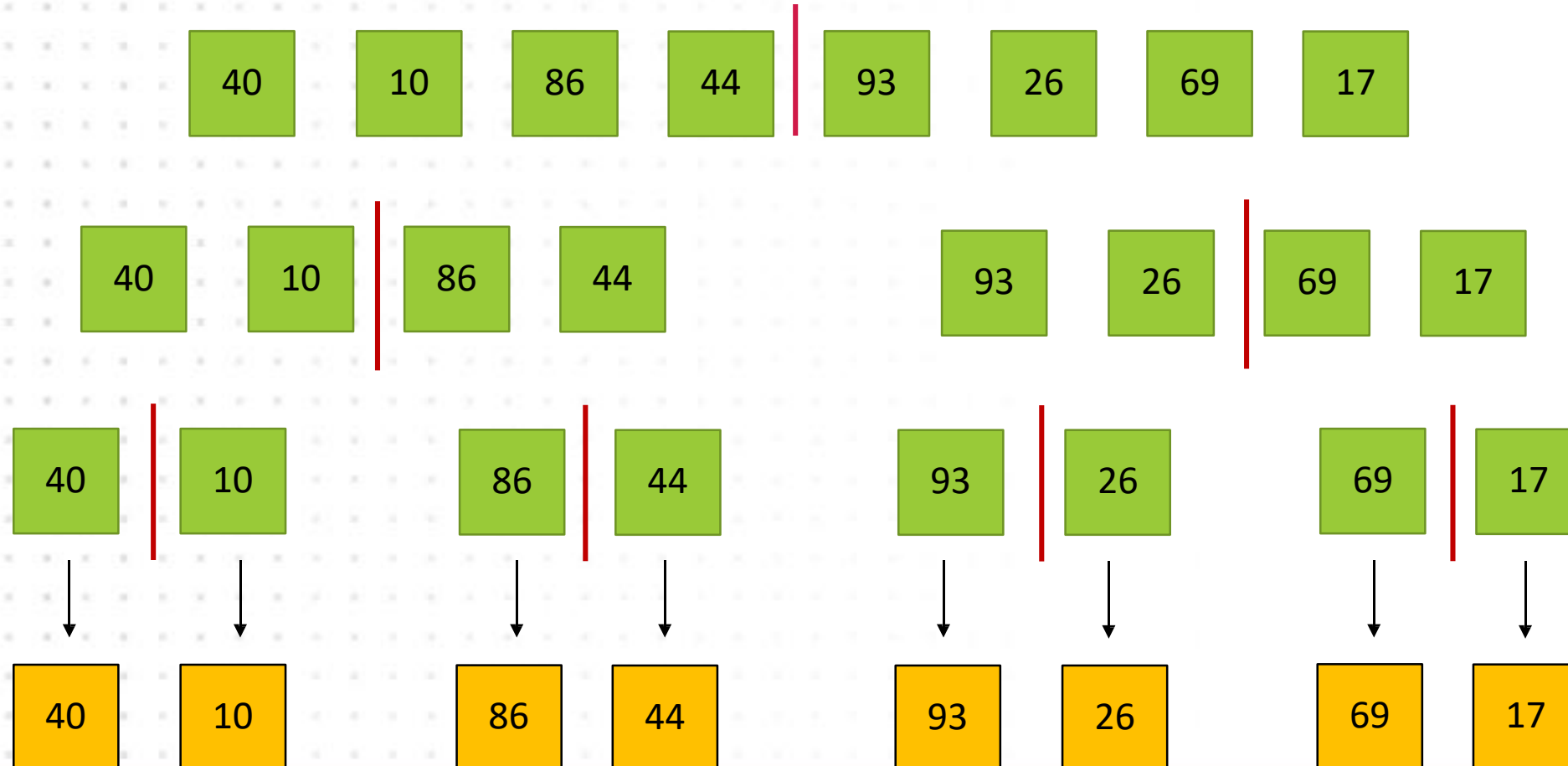


Universidad
Tecnológica
del Perú



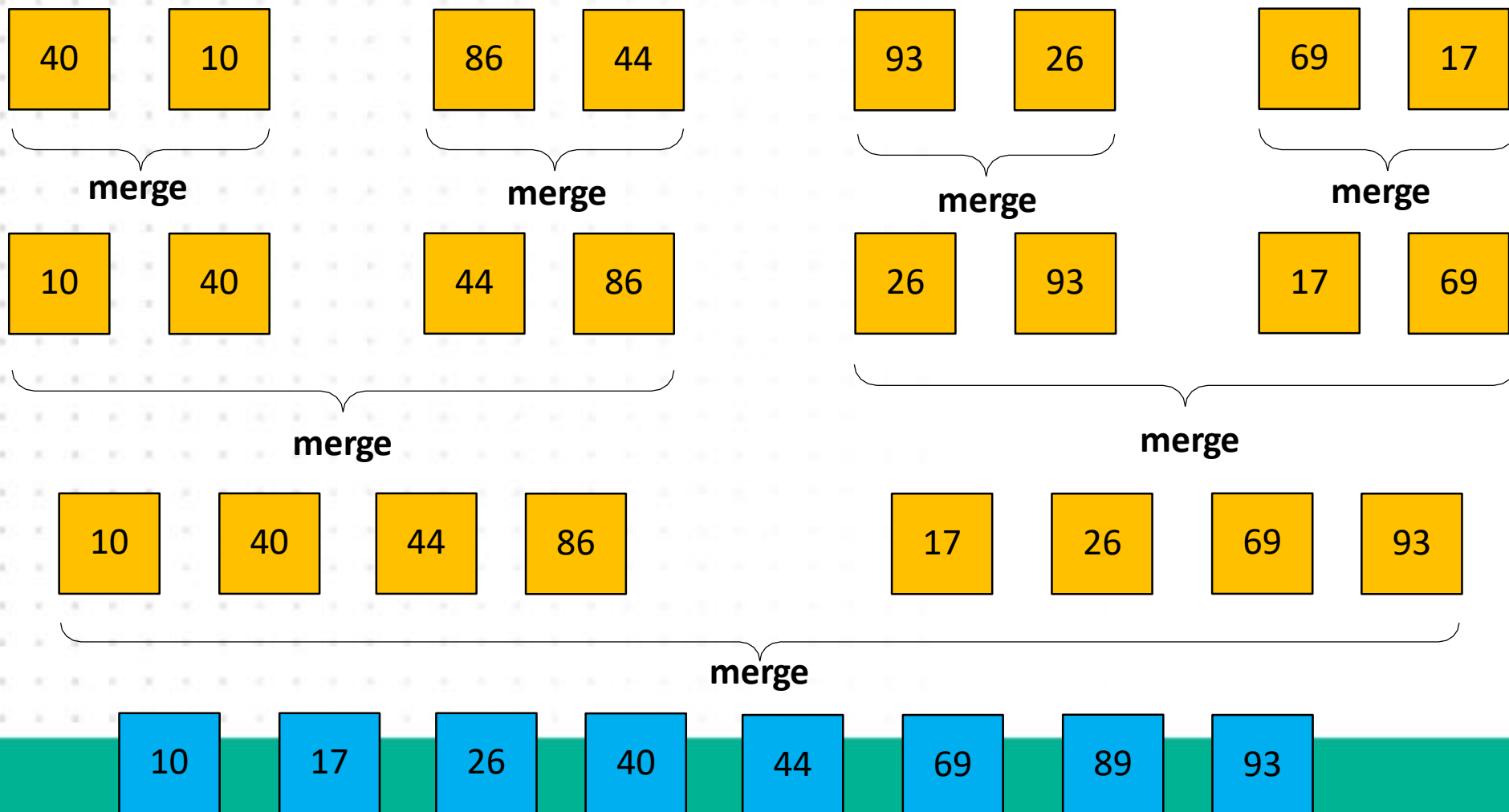
Métodos de ordenamiento

Proceso de división



Métodos de ordenamiento

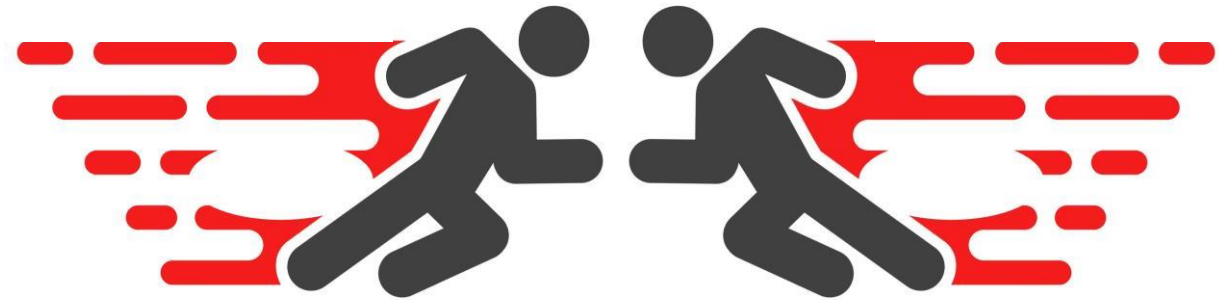
Proceso de mezclado (merge)



Métodos de ordenamiento

Quick Sort

- También sigue el enfoque “divide y vencerás”.
Subdivide el array en 2 subarrays de $n/2$ elementos cada uno de manera recursiva
- Se elige un elemento Pivot cercano a la mitad de cada subarray y se utilizan las posiciones Left y Right del array acercándolos al centro y evaluando donde corresponda el swap.
- El método va ordenando las mitades izquierdas de los subarrays, y el algoritmo se ordena de izquierda a derecha
- Su complejidad es $O(n \log n)$



Métodos de ordenamiento

Shell Sort

- Está basado en el algoritmo InsertionSort
- El algoritmo rompe el conjunto original en subconjuntos más pequeños y luego los ordena usando InsertionSort
- Utiliza un intervalo o gap para la creación de subconjuntos
- Su complejidad es $O(n \log n)$



Métodos de ordenamiento

Integrando ordenamiento a Java



Ejercicio Propuesto

Crear la clase java Curso con los siguientes atributos:

`codigo (String), ciclo (int), nombre (String), credits (int)`

Implementar la interfaz Comparable y configurar el orden natural por codigo

Crear la clase AppCurso e instanciar 5 objetos de la clase Curso en un array de manera desordenada

Ordenar los datos utilizando algún algoritmo de ordenamiento avanzado

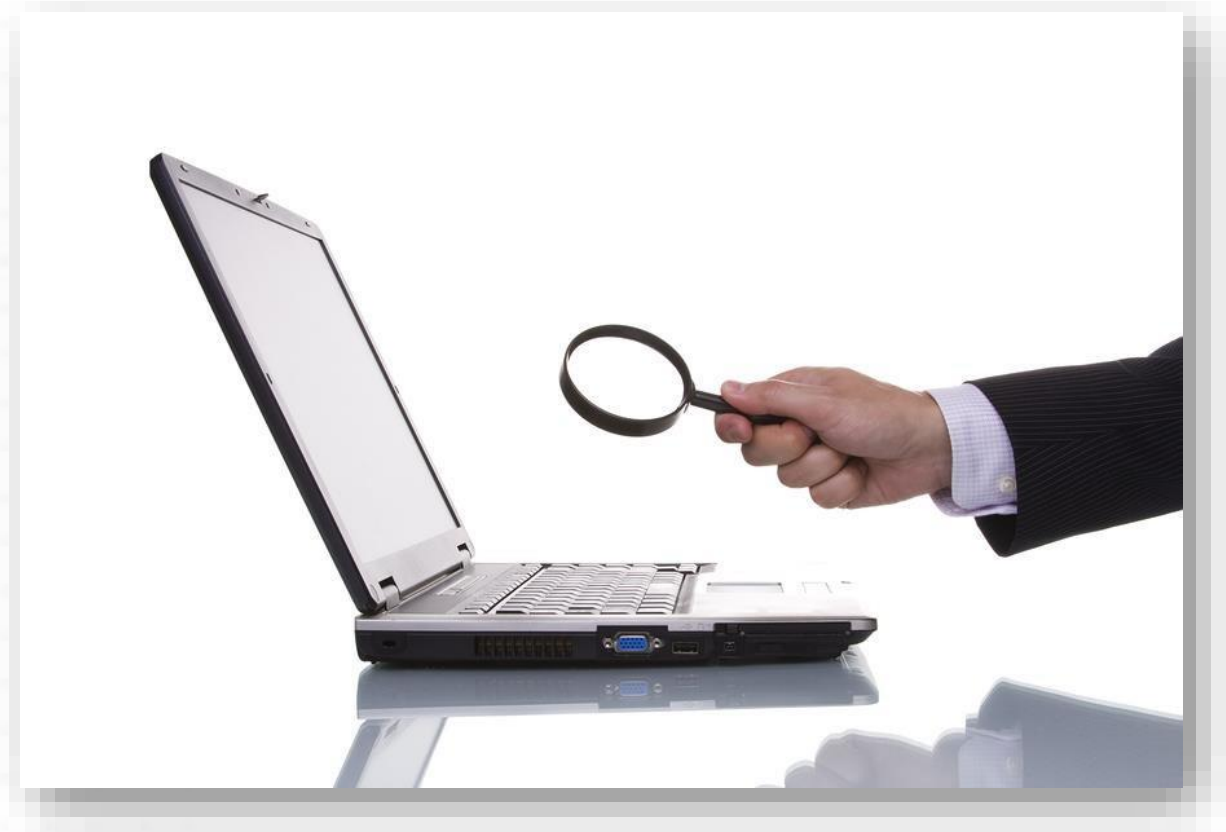


Algoritmos de Búsqueda

Métodos de Ordenamiento

¿Qué es un algoritmos de búsqueda?

- Es un algoritmo encargado de recuperar información
- La búsqueda es una de las aplicaciones más importante de la informática y se le conoce como **searching**
- La búsqueda se realiza en un almacén de datos denominados tablas **lookup** cuya información suele estar emparejada
- Una vez encontrado el elemento buscado, se devuelve la información o la ubicación de la información para su posterior procesamiento.



Algoritmos de búsqueda básicos

Algoritmos de búsqueda básicos

Linear Search

Búsqueda secuencial o lineal

Binary Search

Búsqueda binaria

Algoritmos de búsqueda básicos

Linear Search

Buscar KEY **14**

3

IDX	KEY	VALUE
0	54	LINUX
1	72	WINDOWS
2	26	MAC OS
3	14	FREEBSD
4	98	ANDROID
5	37	IOS

- La búsqueda lineal, recorre todas las claves (**KEY**) del **lookup** y devuelve el índice (**IDX**) si la clave ha sido encontrada.
- El **IDX** servirá para referenciar al valor buscado (**VALUE**)

Algoritmos de búsqueda básicos

Linear Search

Lookup

- Si la clave (KEY) buscada no se encuentra, el algoritmo **devuelve el valor -1**
- Después de llamar a la función de búsqueda, debemos comprobar si la clave (KEY) fue encontrada

Buscar KEY **55**

-1

IDX	KEY	VALUE
0	54	LINUX
1	72	WINDOWS
2	26	MAC OS
3	14	FREEBSD
4	98	ANDROID
5	37	IOS

No se encontró

Algoritmos de búsqueda básicos



Linear Search

Ejercicio Propuesto

Se tiene el siguiente conjunto de elementos: 100, 25, 61, 98, 205, 18, 2

Realizar la representación gráfica de un Linear Search para la búsqueda de las siguientes keys:

205

61

15

Algoritmos de búsqueda básicos

Linear Search en Java

Pseudocódigo

```
Para i desde 0 hasta i < longitud(array)
    Si (arreglo[i] == clave){
        retornar i
    }
    retornar -1
FinPara
```

```
public static int linearSearch(int[] data, int key){
    for (int i = 0; i < data.length; i++) {
        if (data[i] == key) return i;
    }
    return -1;
}
```


Algoritmos de búsqueda básicos

Binary Search

Lookup

Buscar KEY **40**

4

L=0
R=5
M=2

$$M = (R + L) / 2$$

L=3
R=5
M=4

IDX	KEY	VALUE
0	10	LINUX
1	20	WINDOWS
2	25	MAC OS
3	30	FREEBSD
4	40	ANDROID
5	50	IOS

- La búsqueda binaria divide el lookup en 2 partes (LEFT y RIGHT)
- Luego busca a KEY en el punto medio (MIDDLE). Si la clave buscada es menor a KEY, se reinicia la búsqueda en la sección LEFT, caso contrario en RIGHT

Algoritmos de búsqueda básicos

Binary Search

Importante: Los datos deben estar ordenados

Buscar KEY **10**

Paso 1: $10 == 27$? False
 $10 < 27$? True

0	1	2	3	4	5	6	7	8	9	10
10	11	12	20	25	27	30	35	40	45	50

$L=0, R=10, M=(0+10)/2=5$

Paso 2: $10 == 12$? False
 $10 < 12$? True

0	1	2	3	4	5	6	7	8	9	10
10	11	12	20	25	27	30	35	40	45	50

$L=0, R=4, M=(0+4)/2=2$

Paso 3: $10 == 10$? True
Devolver **0**

0	1	2	3	4	5	6	7	8	9	10
10	11	12	20	25	27	30	35	40	45	50

$L=0, R=1, M=(0+1)/2=0$

Algoritmos de búsqueda básicos

Binary Search

Importante: Los datos deben estar ordenados

Buscar KEY **50**

Paso 1: $50 == 27$? False
 $50 > 27$? True

0	1	2	3	4	5	6	7	8	9	10
10	11	12	20	25	27	30	35	40	45	50
					M					

$L=0, R=10, M=(0+10)/2=5$

Paso 2: $50 == 40$? False
 $50 > 40$? True

0	1	2	3	4	5	6	7	8	9	10
10	11	12	20	25	27	30	35	40	45	50
								M		

$L=6, R=10, M=(6+10)/2=8$

Paso 3: $50 == 45$? False
 $50 > 45$? True

0	1	2	3	4	5	6	7	8	9	10
10	11	12	20	25	27	30	35	40	45	50
									M	

$L=9, R=10, M=(9+10)/2=9$

Paso 4: $50 == 50$? True
Devolver **10**

0	1	2	3	4	5	6	7	8	9	10
10	11	12	20	25	27	30	35	40	45	50
										M

$L=10, R=10, M=(10+10)/2=10$

Algoritmos de búsqueda

Binary Search

Ejercicio Propuesto

Se tiene el siguiente conjunto de elementos: 10, 20, 30, 40, 50, 60, 70, 80

Realizar la representación gráfica de un Binary Search para la búsqueda de las siguientes keys:

20

60

15

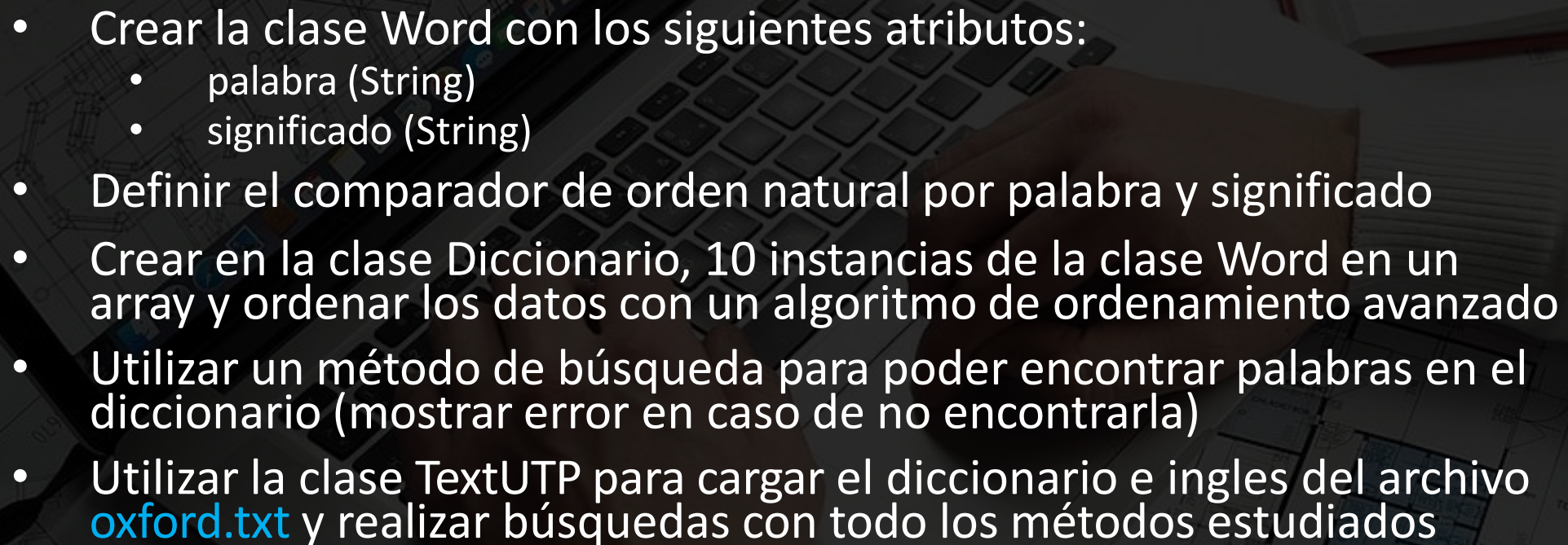
Consultas



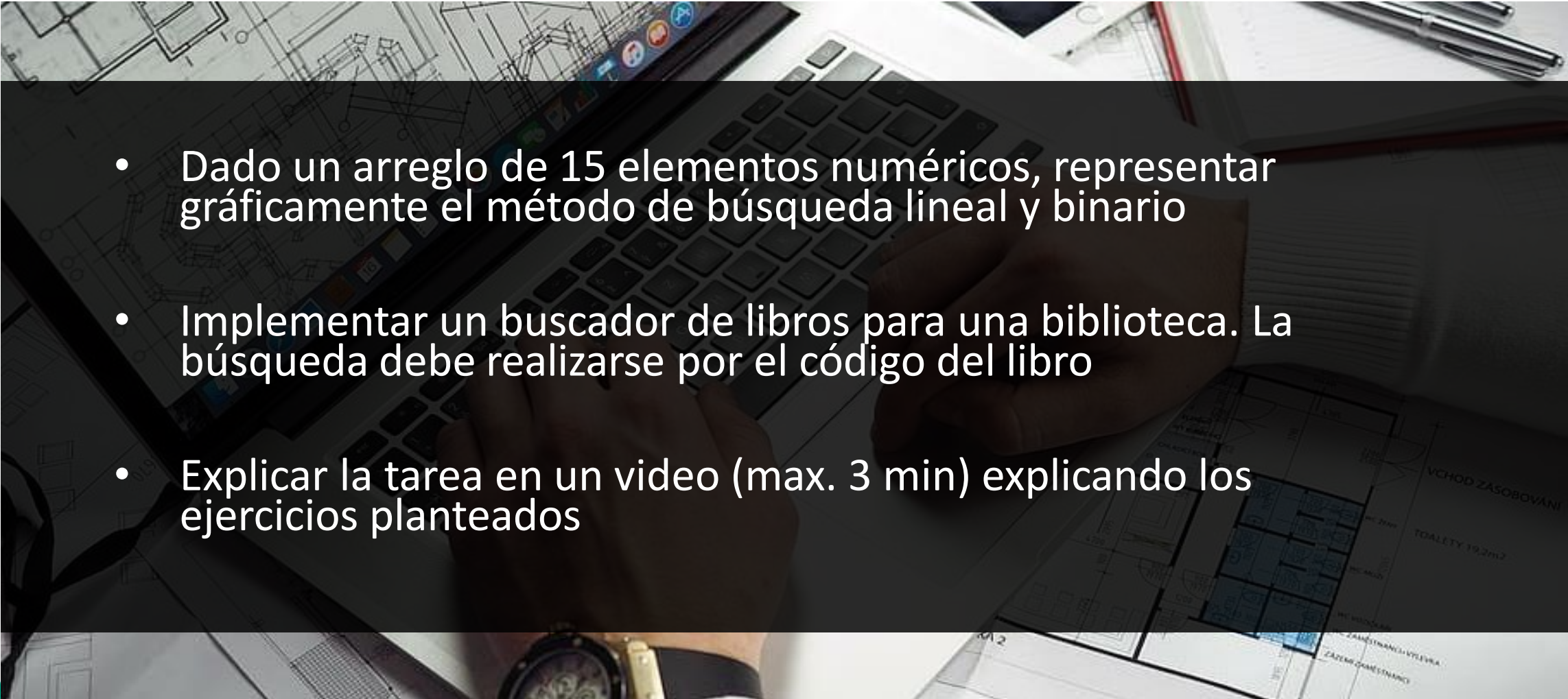
Practiquemos

- Incorporar los algoritmos de búsqueda en la clase AppCurso
- Separar los datos de entrada en el archivo `cursos.txt` que contenga el siguiente formato por cada línea de texto:
`codigo,ciclo,nombre,creditos`
- Utilizar la clase TextUTP para cargar el archivo `cursos.txt` y realizar búsquedas

Practiquemos

- 
- Crear la clase Word con los siguientes atributos:
 - palabra (String)
 - significado (String)
 - Definir el comparador de orden natural por palabra y significado
 - Crear en la clase Diccionario, 10 instancias de la clase Word en un array y ordenar los datos con un algoritmo de ordenamiento avanzado
 - Utilizar un método de búsqueda para poder encontrar palabras en el diccionario (mostrar error en caso de no encontrarla)
 - Utilizar la clase TextUTP para cargar el diccionario e ingles del archivo [oxford.txt](#) y realizar búsquedas con todo los métodos estudiados

Tarea

- 
- Dado un arreglo de 15 elementos numéricos, representar gráficamente el método de búsqueda lineal y binario
 - Implementar un buscador de libros para una biblioteca. La búsqueda debe realizarse por el código del libro
 - Explicar la tarea en un video (max. 3 min) explicando los ejercicios planteados

¿Que hemos aprendido hoy?



- ¿Para que sirven los algoritmos de búsqueda?
- ¿Cuáles son los algoritmos de búsqueda básicos?
- Si los datos están desordenados. ¿Se puede encontrar información?
- ¿Qué algoritmos de ordenamiento pueden ayudar a ordenar la información?

Bibliografía

- Tanenbaum & Van Steen (2008). Algoritmos y Estructuras de Datos - Principios y Paradigmas, 2da Edición. Pearson Education
- Khalid A. Mughal & Rolf W. Rasmussen (2017). A Programmer's guide to Java SE 8 Oracle Certified Associate



**Universidad
Tecnológica
del Perú**