ies SERPIS
GENERALITAT VALENCIANA
CONSELLERIA D'EDUCACIÓ, INVESTIGACIÓ, CULTURA I ESPORT
Fondo Social Europeo

Programación

UD 4: Estructuras de datos estáticas

Ciclo Formativo de Grado Superior
Desarrollo de Aplicaciones Web

Francisco Pérez

1.- Librerías de clases

1.- Librerías de clases

Clases envoltorio (*wrapper*)

✓ En ocasiones es útil tratar los tipos de datos básicos como objetos.

int, byte, short, long, char, boolean, float, double

✓ Muchas funciones y clases trabajan con elementos que heredan de la clase Object.

No funcionarán directamente con estos tipos básicos.

✓ Existe una clase envoltorio por cada tipo básico.

✓ Cada una tiene un único atributo, que es del tipo básico al que “envuelven”.

1.- Librerías de clases

Clases envoltorio (*wrapper*)

Tipo básico	Clase envoltorio
int	Integer
char	Character
boolean	Boolean
long	Long
double	Double
float	Float
short	Short
byte	Byte

1.- Librerías de clases

La clase Integer

https://docs.oracle.com/javase/9/docs/api/java/lang/Integer.html

Constantes

```
int max = Integer.MAX_VALUE;
int min = Integer.MIN_VALUE;
```

Métodos

```
//Pasar de INT a String
int a1 = 45678;
String a2 = Integer.toString( a1 );

//Pasar de String a INT
String b1 = "45678";
int b2 = Integer.parseInt( b1 );
```

Ejemplo:

```
cadena = sc.next(); // Lee la siguiente cadena: “13-14”
String[] separada = cadena.split("-");
a = Integer.parseInt( separada[0] ); // Pasar el 13 de texto a número
b = Integer.parseInt( separada[1] ); // Pasar el 14 de texto a número
```

1.- Librerías de clases

La clase Character

https://docs.oracle.com/javase/9/docs/api/java/lang/Character.html

Ejemplo:

Se ha impuesto una política de contraseñas para seguridad de la información de una empresa. Debemos ayudar a escribir un programa que comprueba si la contraseña es válida (mostrando OK) o inválida (mostrando ERROR).

Los requerimientos son los siguientes:

- ✓ Al menos una letra minúscula.
- ✓ Al menos una letra mayúscula.
- ✓ Al menos un dígito.
- ✓ Al menos un símbolo del conjunto +_)(*&^%$#@!./,;{}
- ✓ Longitud mínima de 12.

```
Character.isDigit('3');          //Devuelve True
Character.isLetter('3');         //Devuelve False
Character.isUpperCase('D');      //Devuelve True
Character.toLowerCase('D');      //Devuelve 'd'
```

1.- Librerías de clases

La clase Random

https://docs.oracle.com/javase/9/docs/api/java/util/Random.html

Crear un objeto de la clase Random

```
Random r = new Random();
```

Hay cuatro funciones miembro diferentes que generan números aleatorios:

Función miembro	Descripción	Rango
r.nextInt()	Número aleatorio entero de tipo int	2^{-32} y 2^{32}
r.nextLong()	Número aleatorio entero de tipo long	2^{-64} y 2^{64}
r.nextFloat()	Número aleatorio real de tipo float	[0,1[
r.nextDouble()	Número aleatorio real de tipo double	[0,1[

1.- Librerías de clases

La clase Random

En el caso de necesitar números aleatorios enteros en un rango determinado, podemos trasladarnos a un intervalo distinto, simplemente multiplicando, aplicando la siguiente fórmula general:

```
(int) (rnd.nextDouble() * cantidad_números_rango + término_inicial_rango)
```

donde (int) al inicio, transforma un número decimal double en entero int, eliminando la parte decimal.

Por ejemplo, si deseamos números aleatorios enteros comprendidos entre [1,6], que son los lados de un dado, la fórmula quedaría así.

```
(int)(rnd.nextDouble() * 6 + 1);
```

donde 6 es la cantidad de números enteros en el rango [1,6] y 1 es el término inicial del rango.

1.- Librerías de clases

La clase Math

https://docs.oracle.com/javase/9/docs/api/java/lang/Math.html

MÉTODO	DESCRIPCIÓN	EJEMPLO DE USO	RESULTADO
abs	Devuelve el valor absoluto de un numero.	int x = Math.abs(2.3);	x = 2;
ceil	Devuelve el entero más cercano por arriba.	double x = Math.ceil(2.5);	x = 3.0;
floor	Devuelve el entero más cercano por debajo.	double x = Math.floor(2.5);	x = 2.0;
round	Devuelve el entero más cercano.	double x = Math.round(2.5);	x = 3.0;
log	Devuelve el logaritmo natural en base e de un número.	double x = Math.log(2.71);	x = 0.9996;
max	Devuelve el mayor de dos entre dos valores.	int x = Math.max(3, 8);	x = 8;
min	Devuelve el menor de dos entre dos valores.	int x = Math.min(3, 8);	x = 3;
random	Devuelve un número aleatorio entre 0 y 1. Se pueden cambiar el rango de generación.	double x = Math.random();	x = 0.206178;
sqrt	Devuelve la raíz cuadrada de un número.	double x = Math.sqrt(9);	x = 3.0;
pow	Devuelve un número elevado a un exponente.	double x = Math.pow(2, 10);	x= 1024.0;
…	…	…	…

CONSTANTE	DESCRIPCIÓN
PI	Devuelve el valor de PI. Es un double.
E	Devuelve el valor de E (Euler). Es un double.

2.- Estructuras de datos

2.- Estructuras de datos

Estructuras de datos estáticas

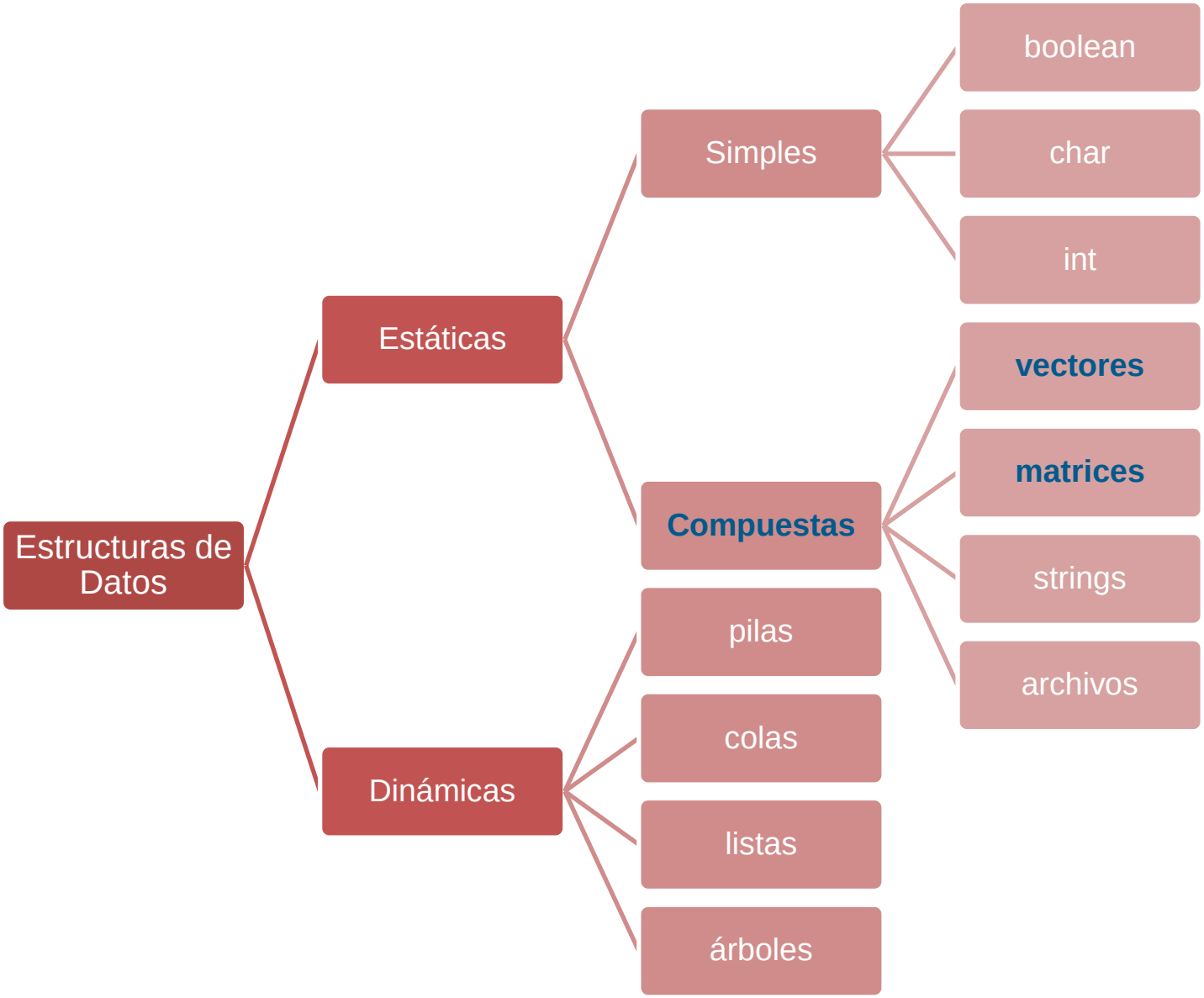
Las estructuras estáticas son aquellas en las que el tamaño ocupado en memoria se define con anterioridad a la ejecución del programa que los usa, de forma que su dimensión no puede modificarse durante la misma (p.e., un vector o una matriz) aunque no necesariamente se tenga que utilizar toda la memoria reservada al inicio (en todos los lenguajes de programación las estructuras estáticas se representan en memoria de forma contigua).

Estructuras de datos dinámicas

Por el contrario, ciertas estructuras de datos pueden crecer o decrecer en tamaño, durante la ejecución, dependiendo de las necesidades de la aplicación, sin que el programador pueda o deba determinarlo previamente: son las llamadas estructuras dinámicas. Las estructuras dinámicas no tienen teóricamente limitaciones en su tamaño, salvo la única restricción de la memoria disponible en el computador.

NOTA: Las estructuras de datos dinámicas se estudian en la UD7

2.- Estructuras de datos



3.- Creación de arrays

3.- Creación de arrays

Los arrays permiten almacenar una colección de objetos o datos del mismo tipo. Son muy útiles y su utilización es muy simple:

Declaración del array: La declaración de un array consiste en decir “esto es un array” y sigue la siguiente estructura:

```
tipo[] nombre;
```

El tipo será un tipo de variable o una clase ya existente, de la cual se quieran almacenar varias unidades.

Creación del array: La creación de un array consiste en decir el tamaño que tendrá el array, es decir, el número de elementos que contendrá, y se pone de la siguiente forma:

```
nombre = new tipo[dimension]
```

Donde dimensión es un número entero positivo que indicará el tamaño del array.

Una vez creado el array este no podrá cambiar de tamaño.

Declaración y creación:

```
tipo[] nombre = new tipo[dimension]
```

3.- Creación de arrays

Ejemplos:

```java
//Declarar un vector
int[] numeros;

//Declarar y crear un vector de tamaño 7 (0..6)
String[] dias = new String[7];

//Crear e Inicializar un vector
int[] inicializado = {1, 2, 3, 4, 5};

//Modificar el valor de una posición del vector
dias[0] = "lunes";
dias[3] = "jueves";

//Acceder al valor de una posición del vector
int valor = inicializado[2];

//valor = 3
System.out.println( valor );
```

3.- Creación de arrays

Recorrido Ascendente

```
public static void imprimirVectorAscendente(int[] miVector){
        for (int i = 0; i < miVector.length; i++) {
                System.out.printf("%4d", miVector[i]);
        }
        System.out.printf("\n");
}
```

Recorrido Descendente

```
public static void imprimirVectorDescendente(int[] miVector){
        for (int i = miVector.length-1; i >= 0; i--) {
                System.out.printf("%4d", miVector[i]);
        }
        System.out.printf("\n");
}
```

3.- Creación de arrays

La clase Arrays

La biblioteca de clases de Java incluye una clase auxiliar llamada java.util.Arrays que incluye como métodos algunas de las tareas que se realizan más a menudo con vectores:

Arrays.sort(v) ordena los elementos del vector.

Arrays.equals(v1, v2) comprueba si dos vectores son iguales.

Arrays.fill(v, val) rellena el vector v con el valor val.

Arrays.toString(v) devuelve una cadena que representa el contenido del vector.

Arrays.binarySearch(v, k) busca el valor k dentro del vector v (que previamente ha de estar ordenado).

4.- Arrays multidimensionales

4.- Arrays multidimensionales

```
//Declarar una matriz
int[][] matriz;


//Declarar e Inicializar una matriz
int[][] miMatriz = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};


//Obtener el número de filas y columnas de la Matriz
int filas    = miMatriz.length;
int columnas = miMatriz[0].length;


//Recorrer e Imprimir todos los elementos de la Matriz
for (int i = 0; i < filas; i++){

        for(int j = 0; j < columnas; j++){

                //Recuperar el valor de la posición (i,j) de la Matriz
                System.out.printf("%4d", miMatriz[i][j]);
        }
        System.out.printf("\n");
}
```

4.- Arrays multidimensionales

```
//Declarar y crear una matriz vacía

int[][] puntuaciones = new int[3][3];


//Asignar valores a todas las posiciones de la matriz

puntuaciones[0][0] = 50;
puntuaciones[0][1] = 100;
puntuaciones[0][2] = 75;
puntuaciones[1][0] = 0;
puntuaciones[1][1] = 84;
puntuaciones[1][2] = 17;
puntuaciones[2][0] = 369;
puntuaciones[2][1] = 8;
puntuaciones[2][2] = 44;
```

	[0]	[1]	[2]
[0]	50	100	75
[1]	0	84	17
[2]	369	8	44

6.- Cadenas de caracteres

6.- Cadenas de caracteres

La clase String

https://docs.oracle.com/javase/9/docs/api/java/lang/String.html

Son parte del lenguaje (no hay que importarlos)

Se crean:	String s = new String(“Hola Mundo”);
pero esto se puede resumir con	String s = “Hola Mundo”;
Tamaño de un String:	int i = s.length();
k-esimo carácter:	char c = s.charAt(k);
Subsecuencias:	String sub = s.substring(k);
	String sub = s.substring(inicio, fin);
Búsqueda de subsecuencias:	int i = s.indexOf(“hola”);
	int i = s.indexOf(String str, int inicio);

6.- Cadenas de caracteres

La clase String

Comparacion (boolean): s1.equals(s2);

Comparacion (entero): int i = s1.compareTo(s2);

0 si s1 == s2,

>0 si s1 > s2,

<0 si s1 < s2

int compareToIgnoreCase(String otra)

Pasar a mayúsculas: String.toUpperCase()

Pasar a minúsculas: String.toLowerCase()

Quitar espacios: String.trim()

Separar cadenas: String[] separada = cadena.split("-");

6.- Cadenas de caracteres

La clase String

```
String cadena = "Esto es un ejemplo";

System.out.println(cadena.charAt(2));          //t
System.out.println(cadena.indexOf("es"));      //5
System.out.println(cadena.toLowerCase());      //esto es un ejemplo
System.out.println(cadena.toUpperCase());      //ESTO ES UN EJEMPLO
System.out.printf("|%s|", " cadena ".trim()); //|cadena|
```