{ Estructura de Datos }

/*

* Primer Semestre

* Semana 3.1

*/

Introducción

- Clasificación de las Estructuras de datos estáticas y dinámicas.
- Estructuras de datos estáticas simples.
  - Arreglos.
    - Unidimensionales.

Clasificación de las Estructuras de datos

- En programación, una estructura de datos es una forma de organizar un conjunto de datos con el objetivo de facilitar su manipulación.

¿?

Clasificación de las Estructuras de datos estáticas

- Estructuras de datos primitivas:
  - Son primitivas son aquellas que no están compuestas por otras estructuras de datos, por ejemplo, enteros (int, byte, short, Long), decimales (float, double), booleanos (true, false) y caracteres (char).

Enteros

```
byte numeroByte = 127;
short numeroShort = 32767;
int numeroEntero = 2147483647;
long numeroLong = 92233720368547758**07L**;
```

Decimales (punto flotante)

```
float numeroFlotante = 3.4F; //3.4 * 10^38
double numeroDoble = 1.8; //1.8 * 10^308
```

Caracteres

```
char caracter = '@';
```

Booleanos

```
boolean verdadero = true;
boolean falso = false;
```

Clasificación de las Estructuras de datos estáticas

- Estructuras de datos simples:
  - Las estructuras de datos simples se construyen a partir de estructuras de datos primitivas.
  - Por ejemplo, Cadenas (String), Arreglos (unidimensional y multidimensional)

Cadenas de caracteres

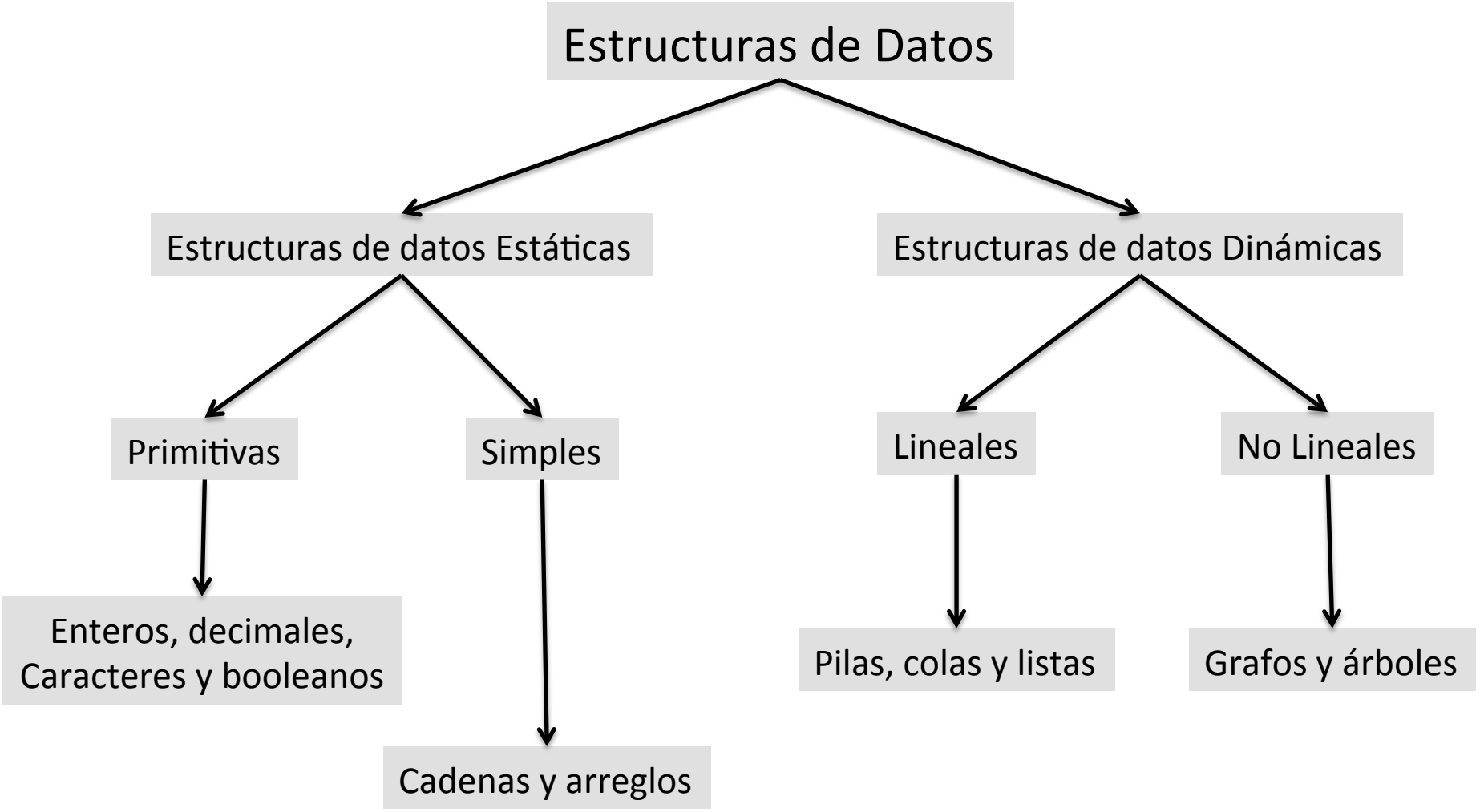
```
char[] vectorChar = new char[]{'H','O','L','A'};

String cadena = new String(vectorChar);

System.out.println("Cadena: " + cadena);
```

Clasificación de las Estructuras de datos dinámicas

- Estructuras de datos lineales y no lineales:
  - Las estructuras de datos **lineales** son: Las pilas (Stack), las colas o filas (Queue) y las listas lineales.
  - Las estructuras de datos **no lineales** son: Los grafos (Graphs)y los árboles (Trees).

¿?

Resumen


Estructuras de Datos
Estructuras de datos Estáticas
Estructuras de datos Dinámicas
Primitivas
Simples
Lineales
No Lineales
Enteros, decimales,
Caracteres y booleanos
Cadenas y arreglos
Pilas, colas y listas
Grafos y árboles

Estructuras de Datos Estáticas Simples

- *¿Qué pasa si tenemos una gran cantidad de datos que guardan una relación entre sí*?
  - Para cada uno de esos datos se debería utilizar una variable distinta.
  - Para resolver estas dificultades se agrupan los datos en un mismo conjunto, bajo un nombre común, que se pueden tratar como una sola unidad.

Edades, nombres, listas de precios, etc.

Estructuras de Datos Estáticas Simples

- Arreglos (o Arrays):
  - Un arreglo es una estructura de datos con un conjunto de elementos homogéneos, es decir, del mismo tipo, numérico o alfanumérico, reconocidos por un nombre común, y que residen en la *memoria del computador*.
  - A cada elemento se accede por la posición (o índice) que ocupa dentro del conjunto de datos.

Estructuras de Datos Estáticas Simples

- Arreglos:
  - Antes de utilizarlos, hay que reservar una zona de la memoria para su uso, así como definir el numero de parámetros necesarios para acceder a cada elemento de la estructura, es decir, *dimensionarlos*.
  - Según el numero de parámetros necesarios para dimensionar los arreglos, se pueden clasificar en los siguientes tipos:
    - Unidimensionales.
    - Bidimensionales.
    - Multidimensionales.

Estructuras de Datos Estáticas Simples

- Arreglos Unidimensionales o vectores:
  - Se trata de un **conjunto ordenado** de elementos por posición (de 0 a n) y es **homogéneo**, por que todos sus elementos son del mismo tipo [de dato].

Un vector de tipo numérico con una dimensión de 5 espacios

5	20	-1	11	45

Un vector de tipo alfanumérico con una dimensión de 5 espacios

VEGETTA777	elrubiusOMG	sTaXxCraft	Fernanfloo	LoL Esports

Estructuras de Datos Estáticas Simples

- Arreglos Unidimensionales o vectores:
  - Al igual que una variable, un arreglo debe tener un nombre:

Notas:

5.5	6.8	6.9	5.7	6.3

SistemasOperativos:

openSuse	Fedora	Kali Linux	Debian	Ubuntu

Estructuras de Datos Estáticas Simples

- Arreglos Unidimensionales o vectores:
  - Los elementos que están en el **Notas** y en el **SistemasOperativos** ocupan una determinada posición:

Notas:

0	1	2	3	4
5.5	6.8	6.9	5.7	6.3

SistemasOperativos:

0	1	2	3	4
openSuse	Fedora	Kali Linux	Debian	Ubuntu

Estructuras de Datos Estáticas Simples

- Arreglos Unidimensionales o vectores:

  1.- Declaración de Arreglos:

```
String[] Nombres;
```

  2.- Instanciación de Arreglos:

```
Nombres = new String[10];
```

  3.-Inicialización de Arreglos:

```
Nombres[0] = “Richard Stallman”;
```

Estructuras de Datos Estáticas Simples

- Arreglos Unidimensionales o vectores:

  1.- Declaración e instanciación:

```
String[] distribuciones = new String[5];
```

  2.-Declaración e inicialización:

```
String[] distribuciones = {"Debian","Fedora"};
```

  3.-Declaración, instanciación e inicialización:

```
String[] distribuciones = new String[]{"Debian","Fedora"};
```

Ejercicios

Cree una clase con el nombre de **Principal**, la cual deberá tener el método **main** y cree otra clase con el nombre **Ejercicios**, en esta última resuelva los siguientes problemas en métodos diferentes.

1. Declare e inicialice un vector de 10 elementos con nombres de personas.
2. Imprima los elementos situados en las posiciones 5, 9 y 2 del vector anterior.
3. Declare, instancie e inicialice un vector con los números del 1 al 5 y despliegue sus elementos con un bucle *for*.
4. Escriba un programa que llene automáticamente un vector con los numeros del cero al nueve, utilice un bucle *for* y despliegue sus elementos con un *for-each*.
5. Escriba un programa similar al anterior, pero utilice el bucle *while* tanto para el llenado como para el despliegue de los elementos.
6. Escriba un programa similar al anterior, pero utilice el bucle *do-while* tanto para el llenado como para el despligue de elementos.
7. Escriba un programa que llene de forma automática un vector de 10 elementos y sume sus elementos, despliegue tanto los elementos como la suma de estos.
8. Escriba un programa que llene de forma automática un vector con los 20 primeros números pares (utilice el bucle *for*) y despliegue sus elementos.

Ejercicios

1. Escriba un programa que llene de forma automática un vector con los 20 primeros números impares (utilice el bucle *while*), sume y despliegue sus elementos.
2. Escriba un programa que llene de forma automática un vector con los múltiplos de 4 comprendidos entre 0 y 100, sume, cuente y despliegue dichos elementos.
3. Escribir un programa que llene un vector con cinco números consecutivos y haga una copia de ese vector en otro.
4. Escriba un programa similar al anterior, pero multiplique sus elementos por tres y copie sus elementos en otro vector.
5. Cree un vector de 7 elementos con el nombre notas ingrese sus datos por teclado, sume y calcule el promedio.
6. Escriba un programa similar al anterior, pero si su promedio es mayor que 4.0 muestre un mensaje de aprobado y de reprobado en caso contrario.
7. Cree un vector de 5 elementos ingrese sus datos por teclado y determine cuál es el mayor.
8. Escriba un programa similar al anterior pero determine cual es el menor.
9. Escriba un programa similar a los dos anteriores, pero además de determinar cual es el mayor y el menor, muestre la posición o índice de estos.
10. Con dos vectores cree una (simple) agenda telefónica, la búsqueda deberá realizarla con los índices de los vectores.

Fin;