Argentina programa 4.0

Programas y Archivos

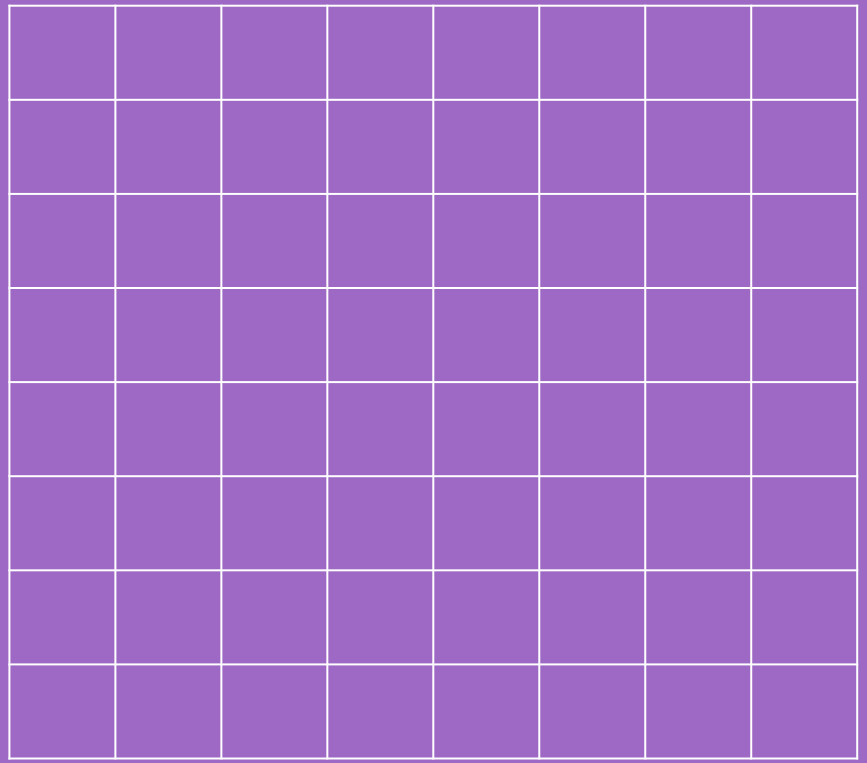
"Desarrollador Java Inicial"

Agenda

Argentina programa 4.0

- Programa
  - Punto de entrada al programa
  - Programas de consola
  - Función main
- Funciones
  - Declaración
  - Usos
- Archivos
  - Sistema de archivos
  - Formatos de Archivo
  - Lectura y Escritura
  - Charsets
- Entradas en un programa
  - Argumentos
  - Lectura de la consola

Programa



Programa – Programas de Consola

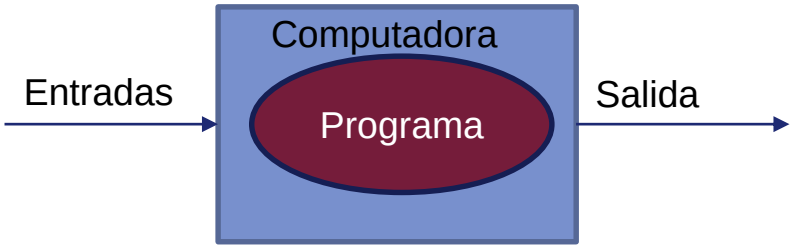
En el desarrollo de curso venimos trabajando con diferentes algoritmos que son “programados” en un archivo.
De aquí en adelante podemos hacer referencia al concepto de ***Programa*** :

“Un programa es un conjunto de instrucciones o algoritmos diseñados para ser interpretadas y ejecutadas por una computadora. “

De manera similar a cómo entendíamos un algoritmo (que podía representarse mediante diagrama de flujo por ejemplo) el programa posee instrucciones o sentencias, pero que son leídas o interpretadas por la computadora.

El programa es la representación del algoritmo en código que se termina ejecutando.

Programa – Programas de Consola



Entradas: por ejemplo el *teclado* o mouse, micrófono

Salidas: ejemplo la *pantalla*, sonido (parlantes, auriculares)

La computadora “ejecuta” un programa, o sea el programa funciona dentro de la computadora.

Punto de Entrada – Función main

Argentina programa 4.0

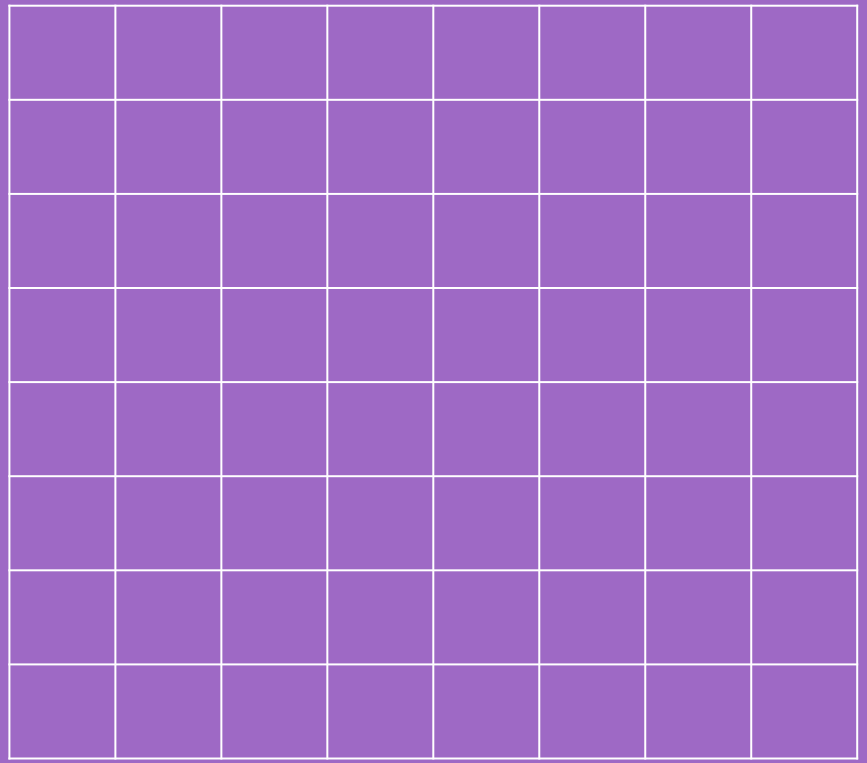
En todas las aplicaciones de consola debemos crear al menos un archivo que posea la función llamada “main” y declarada de la siguiente manera:

```
public static void main(String[] args) {
}
```

Este método o función es muy importante ya que es el ***punto de entrada del programa***, es decir, lo primero que Java (más precisamente el compilador) busca en nuestro programa para ejecutarse.

Si no tenemos eso, no podemos ejecutar el programa creado.

Funciones / Métodos

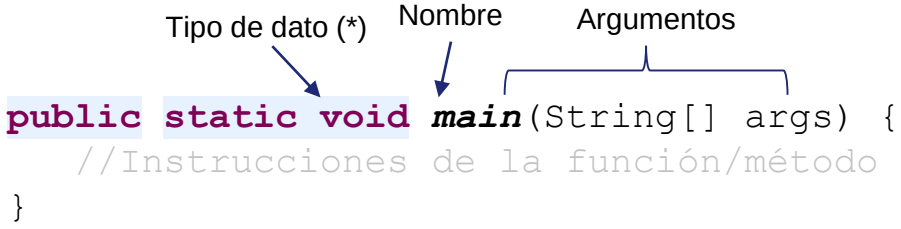


Funciones y Métodos

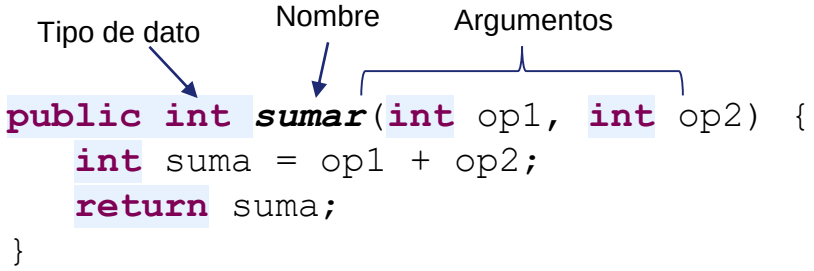
Una función es un bloque de código que sirve para organizar y “modularizar” el mismo. En Java (a diferencia de otros lenguajes que no son de tipo Orientado a Objetos) todas las funciones deben incluirse dentro de una clase y se llaman **métodos**. Siempre tiene un nombre identificatorio, además de tener:

Tipo de retorno: tipo de dato/valor que “devuelve” la función.
Argumentos: datos de entrada necesarios de la función que se usará dentro del código creado en la función.
Ejemplos:

Función/método main



Otra función que definimos llamada “sumar”



* En Java y otros lenguajes *void* es una palabra que indica que el método no devuelve nada

Mirando una función principal o “main”

Argentina programa 4.0

Cuando desarrollamos, es muy común que necesitemos reutilizar parte de nuestros programas. Por ejemplo, “*dado un vector de números, y un número X, que sume todos los números > X y retorne el resultado*”, podríamos plantear algo así:

```
public class Sumatoria {
    public static void main(String[] args) {
        int numeros[] = new int[] { 1, 37, 16 };
        int resultado = 0;
        for (int i = 0; i < numeros.length; i++) {
            resultado = resultado + numeros[i];
        }
        System.out.println(resultado);
    }
}
```

Por un lado, si quiero cambiar lo que debo calcular, tengo que modificar la variable números (entrada), por el otro, el resultado (salida) se imprime por pantalla y se “pierde” al final del programa.

Primer función reutilizable

Argentina programa 4.0

Ahora, vamos a suponer que quiero calcular un promedio. Una forma de hacerlo es volver a escribir el "main":

```
public class Promedio {
	public static void main(String[] args) {
		int numeros[] = new int[] { 1, 37, 16 };
		int resultado = 0;
		for (int i = 0; i < numeros.length; i++) {
			resultado = resultado + numeros[i];
		}
		resultado = resultado / numeros.length;
		System.out.println(resultado);
	}
}
```

Pero, suponiendo que no quiero perder el trabajo que ya tengo, conceptualmente sería usar el codigo que use para sumar y dividirlo por su largo

```
public class Promedio {
	public static void main(String[] args) {
		int numeros[] = new int[] { 1, 37, 16 };
		int resultado = sumatoria(numeros);
		resultado = resultado / numeros.length;
		System.out.println(resultado);
	}
}
```

La pregunta es como representamos esa "`sumatoria(numeros)`", para utilizarla en nuestro programa de promedios.

Primer función reutilizable

Argentina programa 4.0

Entonces queda algo así:

```
public class Promedio {
	public static void main(String[] args) {
		int numeros[] = new int[] { 1, 37, 16
	};
		int resultado = sumatoria(numeros);
		resultado = resultado / numeros.length;
		System.out.println(resultado);
	}

	private static int sumatoria(int[]
numerosASumar)
	{
		int suma = 0;
		for (int i = 0; i <
		numerosASumar.length; i++) {
			suma = suma + numerosASumar[i];
		}
		return suma;
	}
}
```

Vemos entonces la función/método sumatoria, que recibe la variable **números** (puede llamarse con cualquier valor) y retorna un valor en la variable **suma** que se almacena en la variable "resultado" dentro de **main.**

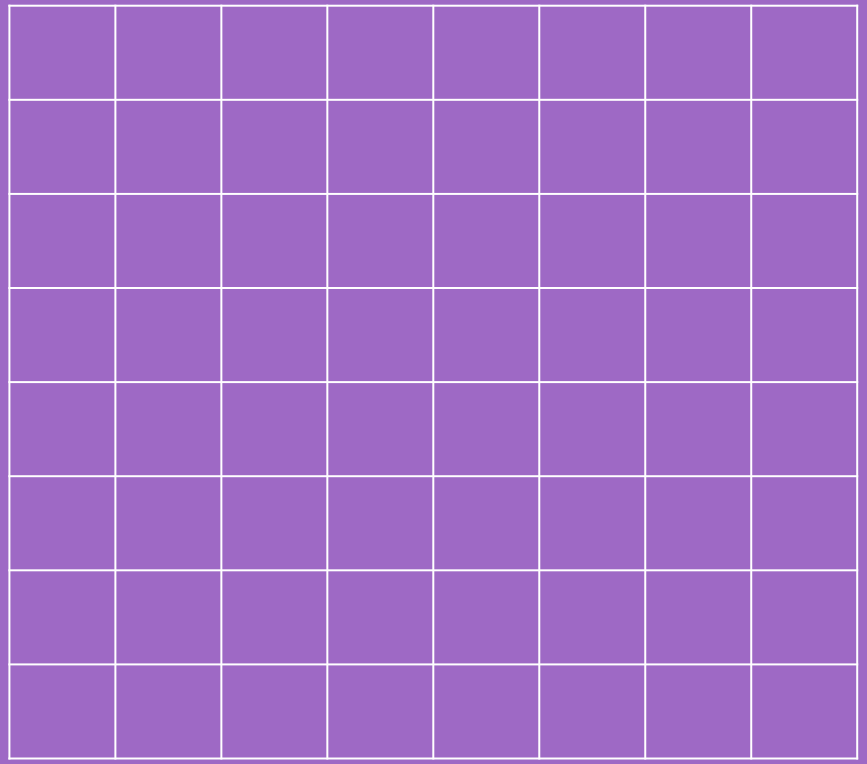
Recordemos que las entradas de las funciones se denominan parámetros o argumentos, en el caso de **sumatoria**, el único es **numerosASumar**. Las funciones pueden tener ninguno o más parámetros.

Aclaraciones importantes

Argentina programa 4.0

- En realidad, como comentábamos Java no tiene funciones, sino “métodos”, ya que están asociados a un objeto o clase, pero recién la próxima clase podremos apreciar la diferencia.
- Ya venimos usando Métodos, por ejemplo:
    - println : `System.out.println("un texto")`
    - split: `"hola que tal?".split(" ")`

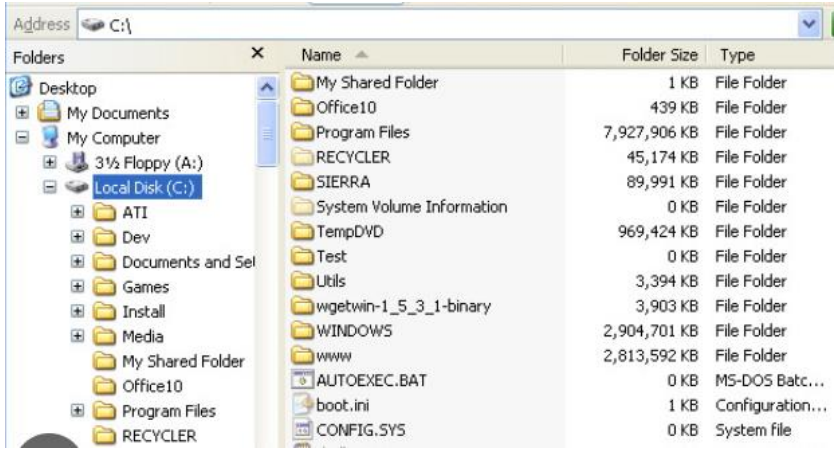
Sistema de Archivos



Sistema de Archivos

En general, la información duradera dentro de una computadora, se almacena en una estructura de archivos (files) y directorios o carpetas(folders). Las carpetas contienen archivos y otras carpetas, mientras que los archivos contienen información, cuya interpretación depende de su formato (por ejemplo texto, video, sonido, pdf, programa ejecutable, etc.)

En el Sistema Operativo Windows, el padre de todas las carpetas es la unidad de almacenamiento o disco rígido en uso, por ejemplo C: , D: etc...



También en Windows, en general, el formato del archivo viene acompañado de una extensión, por ejemplo pdf, mp3, .ext, .png, etc.
Por ejemplo, el código “fuente” de java se almacena en archivos .java y como veremos, su versión “compilada” se almacena en archivos .class.

Sistema de Archivos

Argentina programa 4.0

Cuando se quiere acceder a un archivo o directorio, debe conocerse su “ruta”, es decir dentro de qué directorios está lo que queremos buscar. En general dicha ruta se expresa con un String y el carácter “/” (dividido) se usa para separar los directorios y archivos (en Windows también se puede usar el carácter “\” contrabarra, pero como vimos en la unidad anterior, al ser el carácter de escape, hay que utilizarlo 2 veces para que se represente a sí mismo).

Por ejemplo:

C:\\Users\\JuanPablo\\unArchivo.txt

Establece que: dentro del disco rígido que tenemos configurado como **C:** , en el directorio **Users**, se encuentra el directorio **JuanPablo** y dentro de ese, esta el archivo de texto, **unArchivo.txt**

Archivos - Operaciones - Utilidades

Argentina programa 4.0

Por su importancia, prácticamente todos los lenguajes de programación poseen la forma de acceder al sistema de archivos, en el caso de Java, en esta unidad hablaremos de las clases “Files” y “Path”, nos enfocaremos sobre los archivos de texto y su lectura secuencial.

Nuevamente, si bien no llegamos a ver el concepto de clase, los vamos a tratar como utilidades para cumplir nuestros objetivos.

Primero que nada, antes de usar estas clases, es necesario importarlas, para que sean más fáciles de referenciar, para eso, antes de la declaración de nuestra clase principal, vamos a importarlas:

```
import java.nio.file.Files;

import java.nio.file.Path;

public class MiPrograma {

    public static void main(String[] args) {

    //...
```

Archivos - Operaciones - Lectura

Argentina programa 4.0

Lectura de las líneas de un archivo

```
Files.readAllLines(Paths.get("C:\\Users\\JuanPablo\\unArchivo.txt")
```

- De esto hay varios comentarios para hacer

1) `Paths.get("...")` prepara la ruta ingresada para que la pueda aceptar el método readAllLines

2) readAllLines recibe la ruta generada por Paths.get y retorna una lista de Strings. Aún no vimos ese tipo de dato, pero sabemos que puede ser iterado. Por ejemplo:

```
String archivo = "C:\\Users\\JuanPablo\\unArchivo.txt";
for (String linea : Files.readAllLines(Paths.get(archivo))){
    System.out.println(linea);
}
```

1) Esta forma es útil siempre y cuando el archivo no sea muy grande (unos pocos megas)

Archivos - Operaciones - Escritura

Escritura de una línea sobre un archivo

```
Files.write(Paths.get("/home/eze/nuevoArch"), "hola\n que tal?\n".getBytes());
```

`write` recibe la ruta generada y como segundo parámetro los bytes de un String con contenido.
Nuevamente, este método es efectivo, cuando el String a escribir no es muy grande.

Archivos - Operaciones – Crear un archivo

Argentina programa 4.0

Creación de un archivo si no existe:

```
String rutaArchivo = "C:\\miarchivo.txt";

if(!Files.exists(rutaArchivo)) {

    Files.createFile(rutaArchivo);

}
```

Forma alternativa usando la clase `File` :

```
File archivo = new File("C:\\miarchivo.txt");

if(!archivo.exists()) {

    archivo.createNewFile();

}
```

Manipulación de archivos, creación e información extra

Argentina programa 4.0

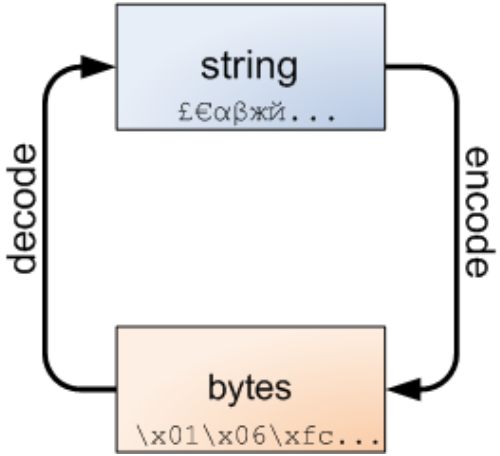
```
Path unArchivo = Paths.get("../recursos/algo.txt");
System.out.println(unArchivo.toAbsolutePath());

FileTime lastModifiedTime = Files.getLastModifiedTime(unArchivo);
System.out.println(lastModifiedTime);

Path otroArch = Paths.get("../recursos2/algoQueNoExiste.txt");
System.out.println(Files.exists(otroArch));
Files.createFile(otroArch);
System.out.println(Files.exists(otroArch));

Path unTemporal = Files.createTempFile("unPrefijo",".txt");
System.out.println(unTemporal);

System.out.println( Files.isDirectory(unTemporal) );
Files.copy(arch1, arch2);
Files.move(arch1, arch3);
```

https://docs.oracle.com/en/java/javase/13/docs/api/java.base/java/nio/file/Paths.html

Encoding

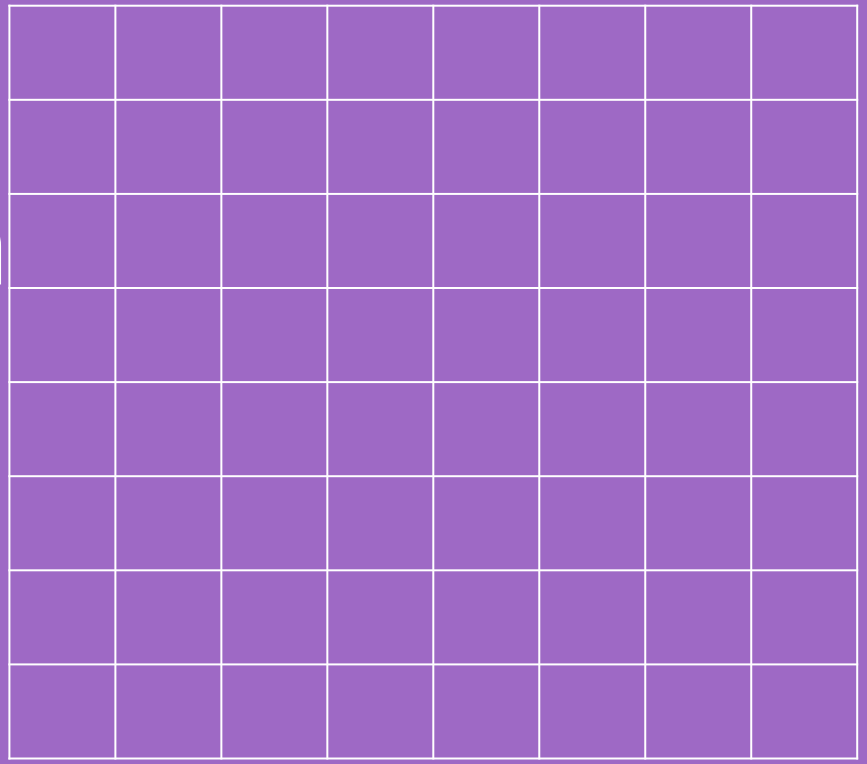
Argentina programa 4.0

```
Path utfFile = Files.createTempFile("some", ".txt");

Files.write(utfFile, "un texto \n con eñes ".getBytes()); // UTF 8

Files.write(utfFile, (System.lineSeparator() +
"y acentos á ").getBytes(),StandardOpenOption.APPEND);

System.out.println(Files.readAllLines(utfFile) .size());

for (String line: Files.readAllLines(utfFile) ) {
System.out.println(line);
}

Path iso88591File = Files.createTempFile("some", ".txt");

Files.write(iso88591File, "otro texto con eñes".getBytes(StandardCharsets.ISO_8859_1));
```



StandardOpenOption y StandardCharsets son Enums o valores enumerados, por ahora solo diremos que son fijos y no se pueden cambiar.

Entradas de un programa



Parámetros de comando

Argentina programa 4.0

Cuando se inicia un programa cualquiera, al mismo se le pueden pasar parámetros. En el caso de Java, si al programa se la pasan, este los leerá desde la variable **args** en **main**. Los parámetros pueden ser cualquier string y en cualquier cantidad.

```
public class Sumatoria {
    public static void main(String[] args) {
        int resultado = 0;
        for (int i = 0; i < args.length; i++) {
            int numero = Integer.parseInt(args[i]);
            resultado = resultado + numero;
        }
        System.out.println(resultado);
    }
}
```

Leer de consola

```java
import java.util.Scanner;

public class Ejercicio1 {

	public static void main(String[] args) {
		Scanner scn = new Scanner(System.in);
		System.out.println(
			"Ingrese números separados por UN
		espacio");
		String numeros = scn.nextLine();
		int resultado = 0;
		for (String numeroStr : numeros.split(" ")) {
			int numero = Integer.parseInt(numeroStr);
			resultado = resultado + numero;
		}
		System.out.println(resultado);
	}
}
```

Otra forma de ingresar datos a un programa es mediante la consola, en este ejemplo el programa nos pedirá que escribamos algo y apretemos la tecla “enter” para terminar

Referencias

Argentina programa 4.0

- https://www.baeldung.com/java-nio-2-file-api
- https://www.marcobehler.com/guides/java-files
- http://www.javalearningacademy.com/streams-in-java-concepts-and-usage/

Argentina
programa
4.0

Gracias!