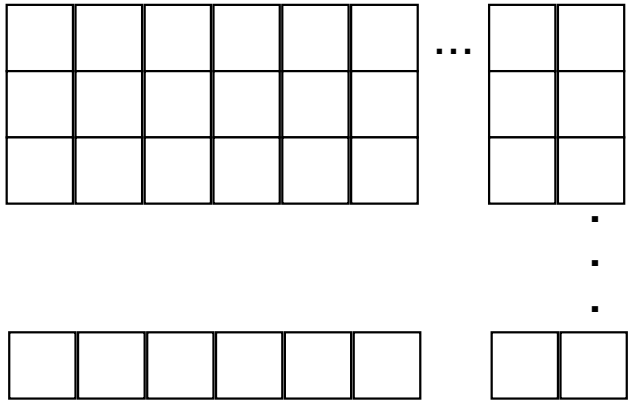
ARRAYS

Prácticas Fundamentos de programación I

Curso 2021/22

ARRAYS

- Estructuras de datos primitiva
- De TAMAÑO FIJO
- Agrupa datos del MISMO TIPO
- MULTIPLES DIMENSIONES



ARRAYS

- Definición de un array unidimensional (vector), sintaxis:
  type [ ] array_name = new type [array_length];
  El nombre del array representa la dirección de memoria (referencia) al objecto.
- Es posible definir una referencia vacía a un array.
  type [] name = null;
- Examplo, definición de un vector con 20 enteros:
  int [ ] results = new int [20];
  - By default,
  - Por defecto, Java inicializa todo el array con valores cero.
- También es posible:
  type[ ] array_name;
  array_name =new tipo [array_length];
- Para acceder a un elemento concreto del array:
  - results [3] //Recuerda que el primer elemento de un array empieza en la posición 0

ARRAYS

- En Java el tamaño de un array puede ser leído a través de la propiedad "length".

```
length = lista.length;
```

- Por otro lado, en Java una lista de elementos puede ser usada para crear e inicializar un array, Su sintaxis es:

```
type [ ] array_name = {lista de elementos};
```

- La lista de elementos es un conjunto de elementos separados por coma. Por ejemplo:

```
int [ ] valores = {22, 56, 1, 39, 88};
```

ARRAYS

- Un índice no puede apuntar a un elemento que esté fuera de los límites del vector. En Java se lanzaría la excepción -> ArrayOutOfBoundsException.
- Diferencia entre:
  - Dimensionamiento Estático. (En tiempo de compilación). Es como hemos explicado hasta ahora.
  - Dimensionamiento Dinámico. (En tiempo de ejecución).
- El siguiente es un ejemplo de dimensionamiento dinámico. El tamaño del vector es leído a través del teclado.

ARRAYS. RECORRIDO SECUENCIAL DE ARRAYS.

- Podemos acceder a todos los elementos desde el primero al último elemento del vector usando un bucle.
- Primero inicializamos la variable índice i a 0, que es el primer elemento en los vectores de Java.
- El bucle se repetirá mientras sea menor que la longitud del vector. i< v.length
- Iremos avanzando el índice de 1 en 1. i++.
- Imprimimos el valor del elemento del vector. v[i].

```
public class Vector01 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int[] v = {31,24,23,4,35};
        for(int i=0;i<v.length;i++) {
            System.out.println(i+": "+v[i]);
        }
    }

}
```

EJEMPLO ARRAY UNIDIMENSIONAL O VECTOR

```java
public class Vector01 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int[] v = {31,24,23,4,35};
        for(int i=0;i<v.length;i++) {
            System.out.println(i+": "+v[i]);
        }
    }

}
```

ARRAYS

- Arrays Multidimensionales o Matrices.
  - Ejemplo de array bidimensional:

```
int [ ] [ ] dosD = new int [4][5];
```

- Los arrays multidimensionales son arrays de arrays. En el siguiente ejemplo inicializamos una lista de arrays. Necesitamos usar la propiedad length para controlar el acceso a todos los elementos:

```
int [ ] [ ] tabla ={ {1}, {2,3}, {4,5,6}, {7,8,9,10} };
```

- Thus we have an array of arrays:

Fila		
0	1	Esta fila es un vector de un elemento.
1	2 3	Esta fila es un vector de 2 elementos.
2	4 5 6	Esta fila es un vector de 3 elementos.
3	7 8 9 10	Esta fila es un vector de 4 elementos.

RECORRIDO DE ARRAYS MULTIDIMENSIONALES.

- Es parecido al recorrido de Arrays.
- Es necesario usar dos bucles anidados.
- Con el bucle más externo recorremos el array de vectores.
- Con el más interno los elementos de cada array.

```
public class Vector02 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int [ ] [ ] tabla ={ {1}, {2,3}, {4,5,6}, {7,8,9,10} };
        for(int i = 0;i<tabla.length;i++) {
            for(int j = 0;j<tabla[i].length;j++) {
                System.out.println("Elemento ["+i+","+j+"] = "+tabla[i][j]);
            }
            System.out.println();
        }

    }

}
```

Bucle para recorrer el array de arrays

Bucle para recorrer cada array de elementos

RECORRIDO ARRAYS MULTIDIMENSIONALES.

```java
public class Vector02 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int [ ] [ ] tabla ={ {1}, {2,3}, {4,5,6}, {7,8,9,10} };
        for(int i = 0;i<tabla.length;i++) {
            for(int j = 0;j<tabla[i].length;j++) {
                System.out.println("Elemento ["+i+","+j+"] = "+tabla[i][j]);
            }
            System.out.println();
        }

    }

}
```

```
Elemento [0,0] = 1

Elemento [1,0] = 2
Elemento [1,1] = 3

Elemento [2,0] = 4
Elemento [2,1] = 5
Elemento [2,2] = 6

Elemento [3,0] = 7
Elemento [3,1] = 8
Elemento [3,2] = 9
Elemento [3,3] = 10
```

EJERCICIO. TOMAR EL MAYOR

- Dado dos vectores de igual tamaño que almacena números enteros, generar un vector resultado del mismo tamaño en el que se ponga el valor mayor de cada índice de los vectores de entrada.

EJERCICIO. TOMAR EL MAYOR

```
int[] vectorA = { 1, 3, 5, 7, 9 };
int[] vectorB = { 9, 7, 5, 3, 1 };
int[] vectorResultado = new int[5];

for (int i = 0; i < vectorA.length; i++) {
   if (vectorA[i] > vectorB[i]) {
         vectorResultado[i] = vectorA[i];
   } else {
         vectorResultado[i] = vectorB[i];
   }
   // vectorResultado[i] = Math.max(vectorA[i], vectorB[i]);
}

System.out.print("\nResultado = | ");
for (int i = 0; i < vectorResultado.length; i++) {
         System.out.print(vectorResultado[i] + " | ");
}
```

EJERCICIO – String [] args

- Tomar varios valores enteros desde los parámetros del programa
- Considerar varios valores separados por espacios, y un separador especiales con el carácter '#'.
- Sumar todos los números enteros entre cada símbolo especial, y poner el resultado en un vector de resultado.
- En cada ejecución se puede meter un número variable de valores
- Como parámetro se puede acabar en un número o símbolo especial

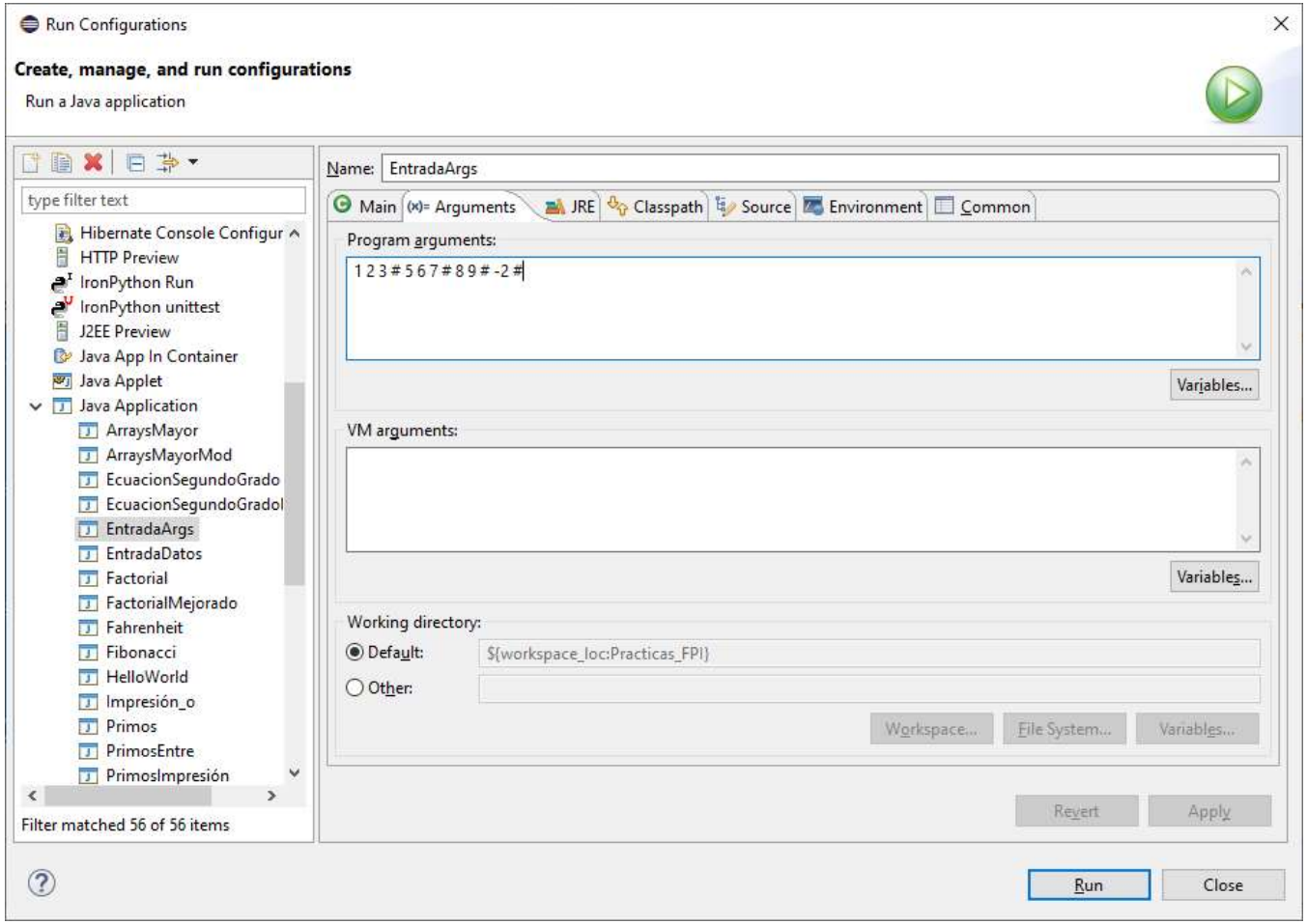
EJERCICIO – String [] args

```
Scanner lee = new Scanner(System.in);
int[] vectorSuma = new int [args.length/2+1]; //vector resultado
int suma = 0; //la suma parcial antes de cada caracter especial de separación
int contadorSumas = 0;

for (int i = 0; i < args.length; i++) {
      String valor = args[i];
      if (!valor.equals("#")) {
            int valorEntero = Integer.parseInt(valor);
            suma += valorEntero;
      } else {
            vectorSuma[contadorSumas] = suma; // se pone la suma acumulada
            suma = 0; // se resetea la suma
            contadorSumas++; // se avanza el contador
      }
}
vectorSuma[contadorSumas] = suma; //se pone la suma acumulada

System.out.print("\nResultado = | ");
for (int i = 0; i <= contadorSumas; i++) {
   System.out.print(vectorSuma[i] + " | ");
}
```

EJERCICIO – String [] args



EJERCICIO – String [] args

- Entrada:

  1 2 3 # 5 6 7 # 8 9 # -2 #

- Salida:

  Resultado = | 6 | 18 | 17 | -2 | 0 |

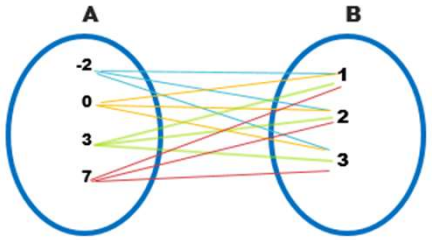
EJERCICIOS A ENTREGAR

EJERCICIO A ENTREGAR – MAYOR

- Diseñar y codificar un programa para determinar el número entero mayor de cada índice comparando dos vectores.
- Se pide realizar una mejora sobre el ejemplo visto en clase en el que:
  - El tamaño de los vectores puede ser diferente.
  - El vector resultado será del tamaño del mayor vector que se comparan.
  - Además, tanto el tamaño de cada vector de entrada, como los valores concretos se pedirán al usuario por teclado.
  - El fichero para entregar se llamará: ArrayMayor.java

EJERCICIO A ENTREGAR – MULTIPLICADOR

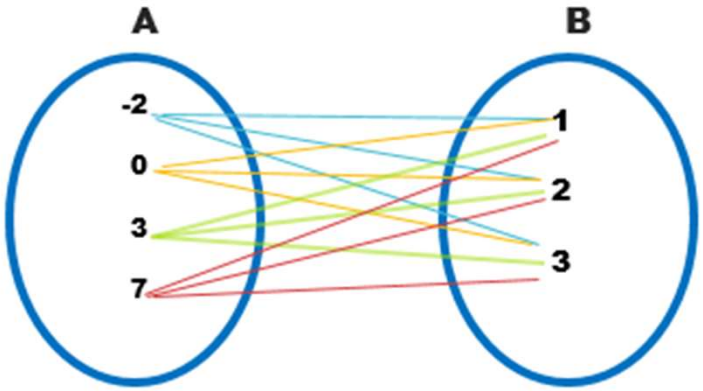
- Diseñar y codificar un programa para calcular el producto cartesiano de dos vectores con la siguiente especificación:
  - El tamaño de los dos vectores será pedido al usuario y ambos vectores pueden ser de tamaño diferente.
  - Los vectores se rellenarán de forma automática con valores *double* aleatorios entre 0 y 9,99.
  - El resultado se guardará en un matriz de resultados
  - El resultado se mostrará por pantalla en forma matricial.
    - La primera fila representa los valores aleatorios de uno de los vectores.
    - La primera columna son los valores del segundo valor.
    - El resto de celdas representa la multiplicación del valor de la fila por la columna correspondiente.



EJERCICIO A ENTREGAR – MULTIPLICADOR

- Ejemplo de entrada y salida:

```
Tamaño Vector A
2
Tamaño Vector B
3
      2,95  4,95  9,62
10,90 32,12 54,01 104,92
10,64 31,34 52,69 102,35
```



- El fichero para entregar se llamará: ArrayMultiplicador.java

DUDAS, PREGUNTAS?

manuel.decastro@uclm.es