Ejercicios ArraysUtils

Explicación de ejercicios:

1. **static double[] create(int length)** --> Usamos el métido que hemos usado siempre para crear un array (**double[] arr = new double[length];**).

```
static double[] create(int length) {
        double[] arr = new double[length];
        return arr;
    }
```

```
ARRAYS UTILS LIBRARY DEMO
=================================
ArraysUtils.create() : 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
[iam2671090@j05 ~]$
```

2. **static String toString(double[] a)** --> Creamos una variable de tipo String, a la cual le añadiremos los valores del array usando el método **String.valueof()** para visualizar la separación de cada elemento, se añadirá una coma antes de insertar un elemento del array, excepto con el primer número.

```
static String toString(double[] a){
        String result = new String("");

        for (int i = 0; i < a.length; i++) {

            if (i != 0){
                result += ", ";
            }
            result += String.valueOf(a[i]);
        }

        return result;
    }
```

3. **static double[] append(double[] a, double value)** --> Creamos un array cuya longitud será la de **a** + 1 (para añadirle otro número), medinate un bucle for, rellenaremos el array con los números de **a**, después añadiremos el

último elemento (**value**) especificando la longitud del array - 1: newArray[newArray.length - 1] = value;

```
static double[] append(double[] a, double value){

        double[] newArray = ArraysUtils.create(a.length + 1);



        for (int i = 0; i < a.length; i++) {
            newArray[i] = a[i];
        }

        newArray[newArray.length - 1] = value;

        return newArray;
    }
```

```
//double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
double[] array = {10.0, 10.0, 11.0, 50.0};
//double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");

System.out.println("ArraysUtils.append() : " + ArraysUtils.toString(ArraysUtils.append(array, 5.6)));
```

```
0 0   Ln 31, Col 1  Spaces: 4  UTF-8  LF  Java
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.append() : 10.0, 10.0, 11.0, 50.0, 5.6
[iam2671090@j05 ~]$
```

4. **static boolean equals(double[] n1, double[] n2)** --> Se resuelve de la misma forma que en el ejercicio **equals** de la práctica de **Arrays Unidimensionales**.

```
static boolean equals(double[] n1, double[] n2){

        boolean salir = false;
        boolean resultado = true;
        int cont = 0;

        //Primera comprobación fuera del bucle, si es falsa, no entramos
        if (n1[cont] != n2[cont]){
```

```
            salir = true;
            resultado = false;
        }


        while(salir == false){

            cont++;

            if(n1[cont] != n2[cont]){

                resultado = false;
            }

            //Condición de salida

            if(resultado == false || cont >= n1.length -1){
                salir = true;
            }

        }

        return resultado;
    }
```

```
    //double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
    double[] array = {10.0, 10.0, 11.0, 50.0};
    //double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
    double[] array2 = {5.0, 5.2, 95.3,60.4};

    System.out.println("ARRAYS UTILS LIBRARY DEMO");
    System.out.println("================================");
    System.out.println("ArraysUtils.equals() : " + ArraysUtils.equals(array, array2));
```

```
⊗ 0 ⚠ 0                                                    Ln 32, Col 1   Spaces
    ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
    ARRAYS UTILS LIBRARY DEMO
    ================================
    ArraysUtils.equals() : false
```

```
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 10.0, 11.0, 50.0};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.equals() : " + ArraysUtils.equals(array, array2));
```

```
0 ⚠ 0                                                    Ln 32, Col 1   Spaces
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.equals() : true
[iam2671090@j05 ~]$
```

5. **static boolean sorted(double[] a)** --> Se resuelve de la misma forma que en el ejercicio **sorted** de la práctica de **Arrays Unidimensionales**.

```
static boolean sorted(double[] a){

        boolean ordenat = true;
        int index = 0;
        if(a.length > 1){

            boolean salir = false;
            while(salir == false){

                if(a[index] > a[index + 1]){
                    ordenat = false;
                }



                //Incremento
                index++;

                //Condición de salida
                if(ordenat == false || index >= a.length -1){
                    salir = true;
                }
            }
        }
```

```
        return ordenat;


    }
```

```
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 10.0, 11.0, 50.0};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.sorted() : " + ArraysUtils.sorted(array));
```

```
0 0                                                    Ln 32, Col
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.sorted() : true
[iam2671090@j05 ~]$
```

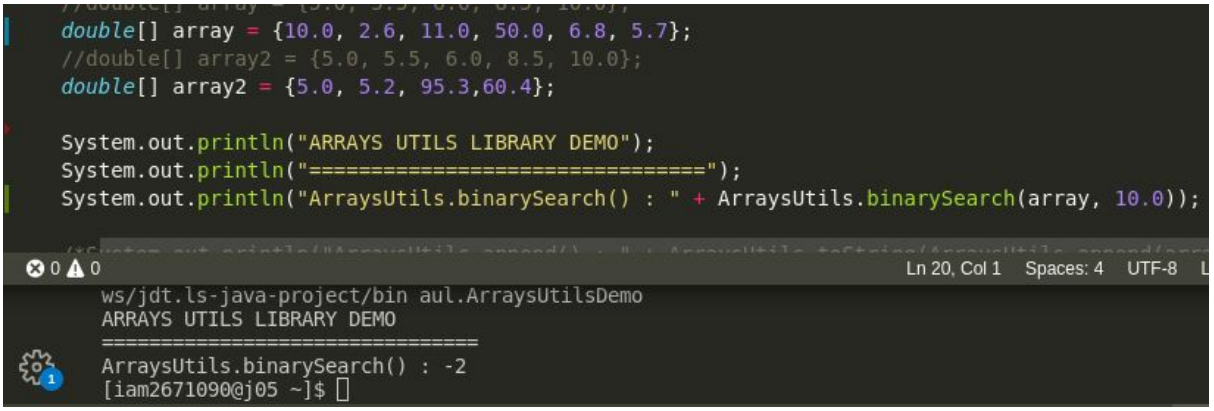
```
double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
//double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.sorted() : " + ArraysUtils.sorted(array));
```

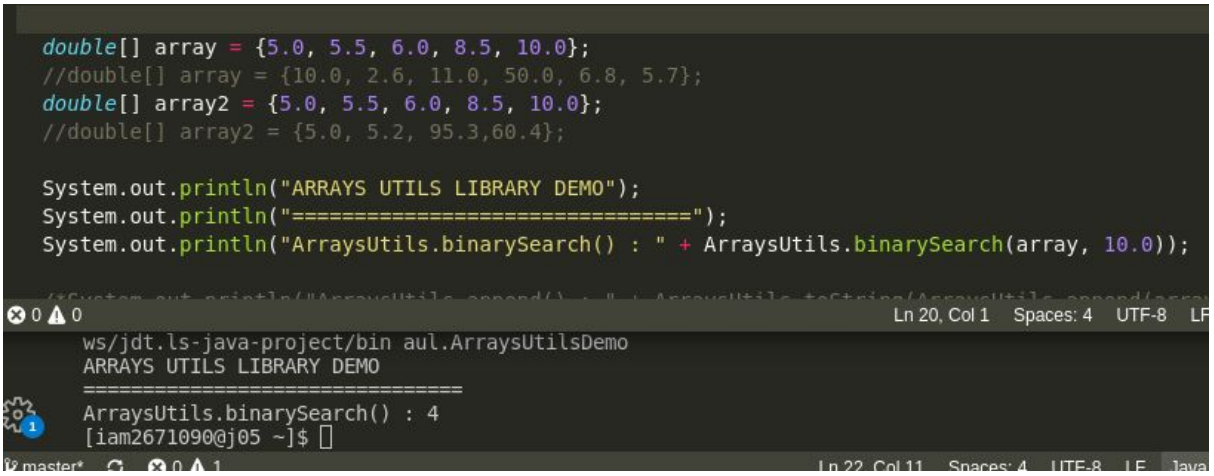
```
0 0                                                    Ln 23, Col 1
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.sorted() : false
[iam2671090@j05 ~]$
```

6. **static int binarySearch(double[] a, double key)** --> Se resuelve de la misma forma que en el ejercicio **binarySearch** de la práctica de **Arrays Unidimensionales**.

```
static int binarySearch(double[] a, double key){


        if (sorted(a)){


            int mayor = a.length -1;
            int menor = 0;
            boolean salir = false;
            boolean encontrado = false;
```

```
            int medio = 0;
            int vueltas = 0;

            while(salir == false){

                medio = (menor + mayor) / 2;

                if (key > a[medio]){

                    menor = medio + 1;

                } else if (key < a[medio]) {
                    mayor = medio - 1;

                } else {
                    encontrado = true;
                }

                vueltas++;

                if (encontrado == true || vueltas >= a.length){
                    salir = true;
                }
            }


            if (encontrado == true){
                return medio;
            }
            else {
                return -1;
            }


        }
        return -2;


    }
```





7. **static int binarySearch(double[] a, int fromIndex, int toIndex, double key)**
   --> Se resuelve de la misma forma que el ejercicio anterior, pero se inicia la variable **mayor** a **toIndex** y **menor** a **fromIndex**.

```
static int binarySearch(double[] a, int fromIndex, int toIndex, double key){



        if (sorted(a)){


            int mayor = toIndex;
            int menor = fromIndex;
            boolean salir = false;
            boolean encontrado = false;
            int medio = 0;
            int vueltas = 0;


            while(salir == false){


                medio = (menor + mayor) / 2;
```

```
            if (key > a[medio]){

                menor = medio + 1;

            } else if (key < a[medio]) {
                mayor = medio - 1;

            } else {
                encontrado = true;
            }

            vueltas++;

            if (encontrado == true || vueltas >= a.length){
                salir = true;
            }
        }


        if (encontrado == true){
            return medio;
        }
        else {
            return -1;
        }


    }
    return -2;
}
```

```java
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.binarySearch() : " + ArraysUtils.binarySearch(array, 0, 2, 5));
```

```
0 ⚠ 0                                   Ln 20, Col 1  Spaces: 4  UTF-8  LF  J
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.binarySearch() : 0
[iam2671090@i05 ~]$
```

8. **static double[] union(double[] a, double[] b)** --> Se crea un array cuya longitud será el resultado de sumar las longitudes de los arrays pasados por parámetro (**a.length + b.length**). Hacemos un primer recorrido con un bucle for desde 0 hasta la longitud del primer array, así habremos rellenado la primera parte del array. Después hacemos otro recorrido desde 0 hasta la longitud del segundo array, pero esta vez, usaremos otro contador además de **i**, usaremos el contador para acceder a los elementos del nuevo array creado y usaremos **i** para acceder a los elementos del segundo array pasado por parámetro.

```java
static double[] union(double[] a, double[] b){

        double[] c = new double[a.length + b.length];

        //Rellenamos la primera parte del array
        for (int i = 0; i < a.length; i++) {

            c[i] = a[i];
        }

        int contador = a.length;
        for (int i = 0; i < b.length; i++) {

            c[contador] = b[i];
            contador++;

        }

        return c;
    }
```

```
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.union() : " + ArraysUtils.toString(ArraysUtils.union(array,array2)));
```

```
0 0   Ln 20, Col 1  Spaces: 4  UTF-8  LF  Java
    ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
    ARRAYS UTILS LIBRARY DEMO
    ================================
    ArraysUtils.union() : 5.0, 5.5, 6.0, 8.5, 10.0, 5.0, 5.5, 6.0, 8.5, 10.0
    [iam2671090@j05 ~]$
```

9. **static double[] insert(double[] a, int index, double value)** --> Primero comprobamos si **index** es mayor que la longitud del array, osea que no está en el rango de elementos de **a**. Si no está, crearemos un nuevo array, con un solo elemento, el cual será -1, y devolveremos su valor. Si **index** está en el rango de **a**, accederemos a ese índice y lo cambiaremos por **value**: **a[index] = value;**.

```
static double[] insert(double[] a, int index, double value){

        if (index > a.length - 1){

            double[] c = new double[1];
            c[0] = -1.0;
            return c;

        } else {
            a[index] = value;
        }

        return a;
    }
```

```
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.insert() : " + ArraysUtils.toString(ArraysUtils.insert(array,2,3.3)))
```

```
0 0   Ln 20, Col 1  Spaces: 4  UTF-8  LF  Java
    ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
    ARRAYS UTILS LIBRARY DEMO
    ================================
    ArraysUtils.insert() : 5.0, 5.5, 3.3, 8.5, 10.0
    [iam2671090@j05 ~]$
```

10. **static double[] remove(double[] a, int index)** --> Se realiza lo mismo que en el ejercicio anterior pero en vez de asignar **a[index] = value,** cambiaremos **value** por 0.

```java
static double[] remove(double[] a, int index){

        if (index > a.length - 1){

            double[] c = new double[1];
            c[0] = -1.0;
            return c;

        } else {
            a[index] = 0;
        }
        return a;

    }
```

```java
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.remove() : " + ArraysUtils.toString(ArraysUtils.remove(array, 3)));
```

```
0 ⚠ 0                                   Ln 20, Col 1  Spaces: 4  UTF-8  LF  Java
    ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
    ARRAYS UTILS LIBRARY DEMO
    ================================
    ArraysUtils.remove() : 5.0, 5.5, 6.0, 0.0, 10.0
```

11. **static void fill(double[] a, double val)** --> Se hace un bucle for desde 0 hasta el último número del array, y a cada elemento de éste le asignaremos **val**.

```java
static void fill(double[] a, double val){



        for (int i = 0; i < a.length; i++) {
            a[i] = val;
        }
```

```
        System.out.println(toString(a));


    }
```

```
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.fill(): ");
ArraysUtils.fill(array, 9.9);
```

```
0 0
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.fill():
9.9, 9.9, 9.9, 9.9, 9.9
[iam2671090@i05 ~]$
```

12. **static void fill(double[] a, int fromIndex, int toIndex, double val)** --> Se resuelve como en el ejercicio anterior, pero esta vez el bucle for será desde **fromIndex** hasta **toIndex**. Sin embargo antes de hacer el bucle se comprobará lo siguiente:
    -Si el rango de inicio es mayor que el final.
    -Si el rango de inicio está al final del array.
    -Si el rango final está al principio del array.
    -Si el índice final es mayor que la longitud del array.

```
static void fill(double[] a, int fromIndex, int toIndex, double val){

        String result = "";

        if (fromIndex > toIndex){

            result = "ERROR: El rango de inicio es mayor que el final";

        } else if(fromIndex >= a.length - 1 && toIndex != a.length - 1){
            result = "ERROR: El rango de inicio está al final del array";
        }

        else if(toIndex == a[0] && fromIndex != a[0]){
```

```
            result = "ERROR: El rango final está al principio del array";
        }

        else if(toIndex > a.length - 1){
            result = "ERROR: El índice final es mayor que la longitud del array";
        }
        else {
            for (int i = fromIndex; i <= toIndex; i++){
                a[i] = val;
            }

            result = toString(a);
        }

        System.out.println(result);

    }
```

13. **static double[] copyOf(double[] original, int newLength)** --> Primero comparamos la longitud del nuevo array con la del array pasado por parámetro, si **newLenght** es mayor que la longitud de **original**, se creará un array cuya longitud será **newLength**, haremos un bucle for para rellenar el array con los elementos de **original**, y después otro bucle for para rellenar el resto con 0. Si **newLength** es menor que la longitud de **original**, haremos un bucle for rellenando todos los elementos del nuevo array con los números que nos alcancen de **original**, si **newLength** es igual a la longitud de **original**, el nuevo array creado será igual a **original**.

```
static double[] copyOf(double[] original, int newLength){

        double[] copy = new double[1];

        if (newLength > original.length){

            copy = new double[newLength];

            //Rellenamos con el array original
```

```
            for (int i = 0; i < original.length; i++) {
                copy[i] = original[i];
            }

            for (int i = original.length; i < copy.length; i++) {
                copy[i] = 0;
            }

        } else if(newLength < original.length){

        copy = new double[newLength];

        for (int i = 0; i < copy.length; i++) {

            copy[i] = original[i];
        }



        } else { //Tienen la misma longitud;
            copy = original;
        }

        return copy;
    }
```

```
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.copyOf() : " + ArraysUtils.toString(ArraysUtils.copyOf(array, 10)));
```

```
0 ⚠ 0                                   Ln 20, Col 1  Spaces: 4  UTF-8  LF  Java
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.copyOf() : 5.0, 5.5, 6.0, 8.5, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0
[iam2671090@i05 ~]$
```

14. **static double[] copyOfRange(double[] original, int from, int to)** --> Antes de hacer algo se harán las mismas comprobaciones que en el ejercicio 12 respecto a los rangos. Si el rango es correcto, crearemos un array cuya longitud será obtenida al calcular la diferencia entre **to** y **from** ((to - from) + 1) después haremos un bucle for para rellenar el array, y como en el ejercicio 8,

usaremos **i** para acceder a los elementos de **original** y un contador para acceder a los elementos del nuevo array.

```
static double[] copyOfRange(double[] original, int from, int to){

        double[] copy = new double[1];

        if (from > to){

            //El rango de inicio es mayor que el final
            copy[0] = -1;

        } else if(from >= original.length - 1 && to != original.length - 1){

            //El rango de inicio está al final del array
            copy[0] = -2;
        }

        else if(to == original[0] && from != original[0]){
            //El rango final está al principio del array
            copy[0] = -3;
        }

        else if(to > original.length - 1){

            //El índice final es mayor que la longitud del array
            copy[0] = -4;
        }

        else {

            copy = new double[(to - from) + 1];
            int counter = 0;

            for (int i = from; i <= to; i++){

                copy[counter] = original[i];
                counter++;
            }
```

```
        }
        return copy;
    }
```

```
] array = {5.0, 5.5, 6.0, 8.5, 10.0};
e[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
e[] array2 = {5.0, 5.2, 95.3,60.4};

out.println("ARRAYS UTILS LIBRARY DEMO");
out.println("================================");
out.println("ArraysUtils.copyOfRange() : " + ArraysUtils.toString(ArraysUtils.copyOfRange(array, 2, 4))
```

```
0 0                                   Ln 20, Col 1  Spaces: 4  UTF-8  LF  Java
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.copyOfRange() : 6.0, 8.5, 10.0
```

15. **static double[] prepend(double[] a, double value)** --> Crearemos un nuevo array cuya longitud será la de **a** + 1. Al primer valor le asignamos **value** y después realizaremos un bucle for para rellenar los siguientes elementos.

```
static double[] prepend(double[] a, double value){

        double[] newArray = ArraysUtils.create(a.length + 1);

        newArray[0] = value;

        for (int i = 0; i < a.length; i++) {
            newArray[i + 1] = a[i];
        }

        return newArray;
    }
```

```
double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
//double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.prepend() : " + ArraysUtils.toString(ArraysUtils.prepend(array, 536.6
```

0 ⚠ 0 Ln 20, Col 1 Spaces: 4 UTF-8 LF Java

```
ws/jdt.ls-java-project/bin aul.ArraysUtilsDemo
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.prepend() : 536.6, 5.0, 5.5, 6.0, 8.5, 10.0
[iam2671090@j05 ~]$
```

master* 0 ⚠ 1 Ln 28, Col 114 Spaces: 4 UTF-8 LF Java

16. **static void sort(double[] a)** --> Se resuelve de la misma forma que en el ejercicio **bubbleSort** de la práctica de **Arrays Unidimensionales**.

```
static void sort(double[] a){

        double aux = 0.0;
        boolean salir = false;
        int cont = 0;

        while(salir == false){

            cont = 0;

            for(int i = 0; i < a.length -1; i++){

                if (a[i] > a[i + 1]){

                    aux = a[i];
                    a[i] = a[i + 1];
                    a[i + 1] = aux;

                } else {
                    cont++;
                }

            }

            //Condición de salida
            if (cont >= a.length - 1){
```

```
                salir = true;
            }
        }

        System.out.println(toString(a));
    }
```

```
double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
//double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.sort(): ");
//ArraysUtils.sort(array, 3, 6);
ArraysUtils.sort(array);

/*System.out.println("ArraysUtils.append() : " + ArraysUtil
```

```
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.sort():
2.6, 5.7, 6.8, 10.0, 11.0, 50.0
[iam2671090@j05 ~]$
```

17. **static void sort(double[] a, int fromIndex, int toIndex)** --> Antes de hacer algo se harán las mismas comprobaciones que en el ejercicio 12 respecto a los rangos. Si el rango es correcto, realizaremos el método del ejercicio anterior, pero el bucle for no se hará desde 0 hasta el último número del array, sino de **fromIndex** hasta **toIndex**.

```
static void sort(double[] a, int fromIndex, int toIndex){

        String result = "";

        if (fromIndex > toIndex){

            result = "ERROR: El rango de inicio es mayor que el final";

        } else if(fromIndex >= a.length - 1 && toIndex != a.length - 1){
            result = "ERROR: El rango de inicio está al final del array";
        }
```

```
else if(toIndex == a[0] && fromIndex != a[0]){
    result = "ERROR: El rango final está al principio del array";
}

else if(toIndex > a.length - 1){
    result = "ERROR: El índice final es mayor que la longitud del array";
}

else {

    double aux = 0.0;
    boolean salir = false;
    int cont = 0;

    while(salir == false){

        for(int i = fromIndex; i < toIndex; i++){

            /*System.out.println(i);
            System.out.println(a[i]);
            System.out.println("contador = " + cont);*/

            if (a[i] > a[i + 1]){

                aux = a[i];
                a[i] = a[i + 1];
                a[i + 1] = aux;

            } else {
                cont++;
            }
        }
        //Condición de salida
        if (cont >= toIndex){
            salir = true;
        }
    }
```

```
            result = toString(a);
        }

        System.out.println(result);
    }
```

```
//double[] array = {5.0, 5.5, 6.0, 8.5, 10.0};
double[] array = {10.0, 2.6, 11.0, 50.0, 6.8, 5.7};
//double[] array2 = {5.0, 5.5, 6.0, 8.5, 10.0};
double[] array2 = {5.0, 5.2, 95.3,60.4};

System.out.println("ARRAYS UTILS LIBRARY DEMO");
System.out.println("================================");
System.out.println("ArraysUtils.sort(): ");
ArraysUtils.sort(array, 2, 4);
```

```
0 0
ARRAYS UTILS LIBRARY DEMO
================================
ArraysUtils.sort():
10.0, 2.6, 6.8, 11.0, 50.0, 5.7
[iam2671090@j05 ~]$
```