

3 Java 3

3.17 Ejercicio 17

3.17.1 Crea un array de enteros y otro array de strings y rellénalo de datos.

```
int[] aEnteros={5,6,7,8,6,9,4,2,8,9};  
String[] aString={"En","un","lugar","de","la","mancha","de","cuyo","nombre","no","quiero","acordarme"};
```

3.17.2 Sacar una posición aleatoria del array de strings y muestra el resultado por pantalla. Hazlo 10 veces.

```
public static void sacarPosN(String[] cadenas, int veces ) {  
    String txt="";  
    for(int i=0;i<veces;i++) {  
        int pos= (int)(Math.random()*(cadenas.length-2));  
        if(i<=(veces-1)) {  
            txt+=cadenas[pos]+" ";  
        }else{  
            txt+=cadenas[pos]+". ";  
        }  
    }  
    System.out.println(txt);  
}
```

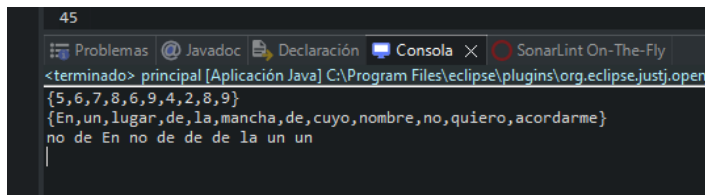
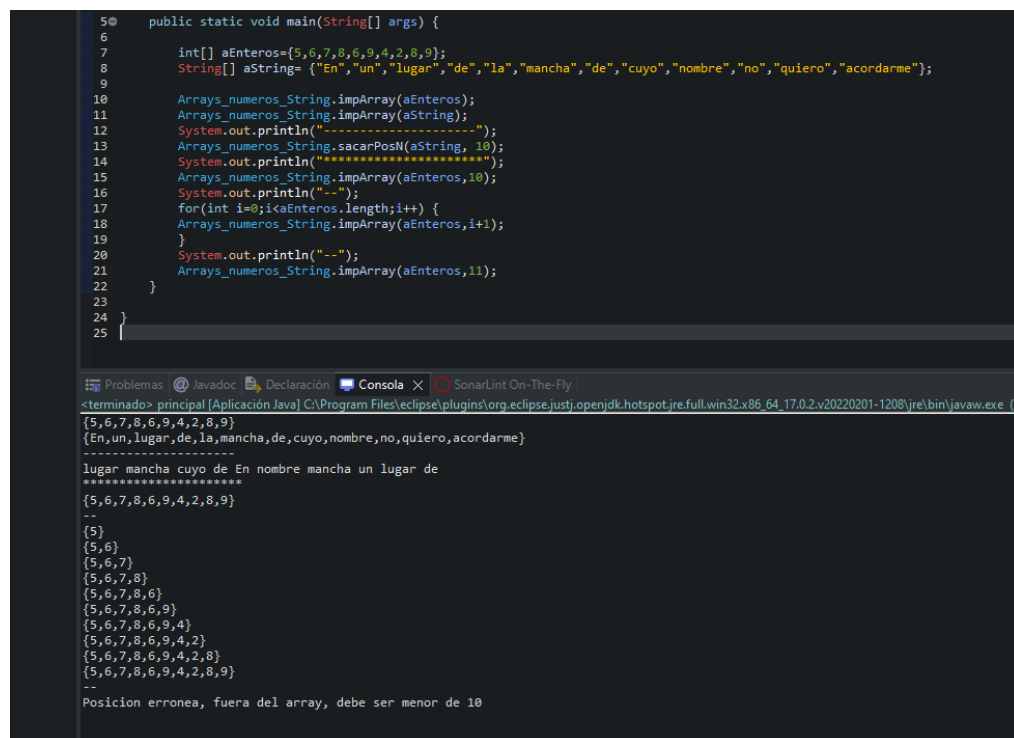


Ilustración 1 array de string aleatorio

3.17.3 Muestra hasta una determinada posición, los valores guardados en el array de enteros :

```
public static void impArray(int[] enteros,int pos) {
    String txt="{";
    if(pos<=enteros.length) {
        for(int i=0;i<pos;i++) {
            if(i!=pos-1) {
                txt+=enteros[i]+",";
            }else {
                txt+=enteros[i]+"}";
            }
        }
    }else {
        txt="Posicion erronea, fuera del array, debe ser menor de "+enteros.length;
    }
    System.out.println(txt);
}
```



```
50 public static void main(String[] args) {
51
52     int[] aEnteros={5,6,7,8,6,9,4,2,8,9};
53     String[] aString= {"En","un","lugar","de","la","mancha","de","cuyo","nombre","no","quiero","acordarme"};
54
55     Arrays_numeros_String.impArray(aEnteros);
56     Arrays_numeros_String.impArray(aString);
57     System.out.println("-----");
58     Arrays_numeros_String.sacarPosN(aString, 10);
59     System.out.println("*****");
60     Arrays_numeros_String.impArray(aEnteros,10);
61     System.out.println("---");
62     for(int i=0;i<aEnteros.length;i++) {
63         Arrays_numeros_String.impArray(aEnteros,i+1);
64     }
65     System.out.println("---");
66     Arrays_numeros_String.impArray(aEnteros,11);
67 }
68
69 }
```

Problemas Javadoc Declaración Consola SonarLint On-The-Fly

<terminado> principal [Aplicación Java] C:\Program Files\ eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (7)

```
{5,6,7,8,6,9,4,2,8,9}
{En,un,lugar,de,la,mancha,de,cuyo,nombre,no,quiero,acordarme}
-----
lugar mancha cuyo de En nombre mancha un lugar de
*****
{5,6,7,8,6,9,4,2,8,9}
--
{5}
{5,6}
{5,6,7}
{5,6,7,8}
{5,6,7,8,6}
{5,6,7,8,6,9}
{5,6,7,8,6,9,4}
{5,6,7,8,6,9,4,2}
{5,6,7,8,6,9,4,2,8}
{5,6,7,8,6,9,4,2,8,9}
--
Posicion erronea, fuera del array, debe ser menor de 10
```

Ilustración 2 ejecución del programa

3.18 Ejercicio 18 Crea un array de enteros y cuenta cuantas veces se repite un determinado valor.

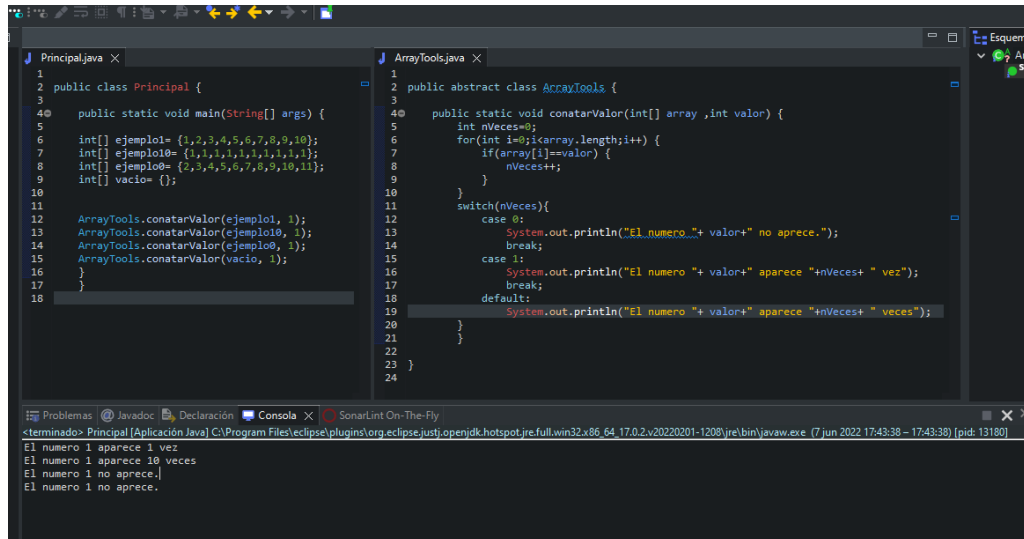


Ilustración 3 contar veces que aparece un número

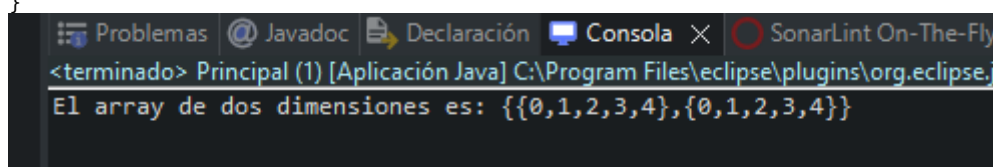
3.19 Ejercicio 19 Arrays multidimensionales

3.19.1 Crea un array bidimensional y recórrelo mostrando los datos por pantalla.

```
public abstract class ArrayDimensiones {

    public static void imprimirArray(int[][] a2D) {
        String txt="";
        for(int i=0 ;i<a2D.length;i++) {

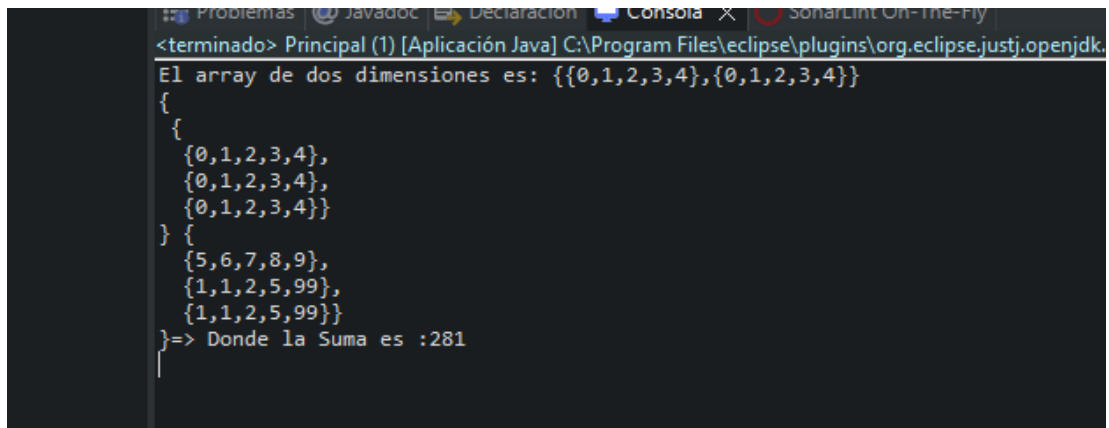
            for(int j=0;j<a2D[i].length;j++) {
                if(j==0) {
                    txt+="{"+a2D[i][j]+",";
                }else if(j==a2D[i].length-1) {
                    txt+=a2D[i][j]+"}";
                }else {
                    txt+=a2D[i][j]+",";
                }
            }
            if(i<a2D.length-1) {
                txt+=",";
            }else{
                txt+="}";
            }
        }
        System.out.println("El array de dos dimensiones es: "+txt);
    }
}
```



3.19.2 Crea un array tridimensional de enteros y recórrelo sumando todos sus valores almacenados

```
public static void imprimirArray(int[][][] a3D) {
    String txt="{\n";
    int sum=0;
    for(int i=0 ;i<a3D.length;i++) {
        txt+=" {\n";
        for(int j=0 ;j<a3D[i].length;j++) {
            for(int k=0;k<a3D[i][j].length;k++){
                sum+=a3D[i][j][k];
                if(k==0) {
                    txt+=" {"+a3D[i][j][k]+",";
                }else if(k==a3D[i][j].length-1) {
                    txt+=a3D[i][j][k]+"}";
                }else {
                    txt+=a3D[i][j][k]+",";
                }
            }
            if(j<a3D[i].length-1) {
                txt+=",\n";
            }else{
                txt+="}\n";
            }
        }
        txt+="}";
    }

    System.out.println(txt+"=> Donde la Suma es
: "+sum);
}
```



```
<terminado> Principal (1) [Aplicación Java] C:\Program Files\eclipse\plugins\org.eclipse.justi.openjdk...
El array de dos dimensiones es: {{0,1,2,3,4},{0,1,2,3,4}}
{
  {
    {0,1,2,3,4},
    {0,1,2,3,4},
    {0,1,2,3,4}}
  {
    {5,6,7,8,9},
    {1,1,2,5,99},
    {1,1,2,5,99}}
  }=> Donde la Suma es :281
```

Ilustración 4 array tridimensional

3.20 Ejercicio 20 Listas.

3.20.1 Crea una lista de personas (la clase que creaste en el ejercicio 14) y muestra por pantalla los detalles de cada una de ellas .

Las dos listas más usadas son [ArrayList](#) y [LinkedList](#)

Import java.util.List; Import java.util.ArrayList; Import java.util.LinkedList;

3.20.1.1 Java ArrayList.

La ArrayList clase es una matriz de tamaño variable, que se puede encontrar en el paquete java.util. La diferencia entre una matriz o array integrado es que el tamaño de este no se puede modificar (si se desea crear o eliminar elementos de una matriz esta se debe eliminar y crear una nueva) .En un ArrayList los elementos se pueden agregar y eliminar en cualquier momento.La sintaxis también es ligeramente diferente:

List <tipoDato> nombrelista=new ArrayList<tipoDato>();

Los elementos de un ArrayList son realmente Objetos, todos los ejemplos los vemos con String pues es un objeto no un tipo primitivo. Para usar tipos primitivos debes especificar la clase contenedora equivalente: Integer,Boolean,Character,Double etc..

3.20.1.1.1 Añadir Elementos Método Add:

```
ArrayList<String> palabras=new ArrayList<String>();//Creacion
palabras.add("Hola");
palabras.add("buenos");
palabras.add("dias");
```

3.20.1.1.2 Acceder a Elementos Método get(pos):

nombrelista.add(pos);

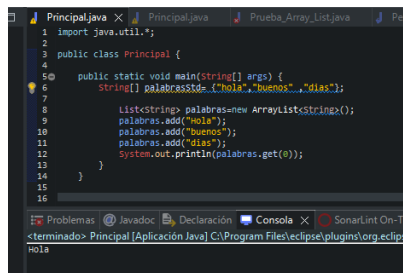


Ilustración 5 acceder a Elementos

3.20.1.1.3 Cambiar Elementos Método Set(pos,valor):

Para modificar un elemento, se utiliza el método set y la posición a modificar

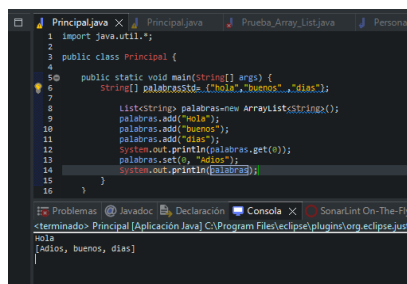


Ilustración 6 cambiar valor

3.20.1.1.4 Quitar Elementos 0 borrar todo:

Para eliminar un elemento, utilizamos el método `remove(indice)` y el número de índice. Da error si sales de la longitud. Reposiciona los índices de los elementos restantes.

```
1 import java.util.*;
2
3 public class Principal {
4
5     public static void main(String[] args) {
6         String[] palabras = {"hola", "buenos", "días"};
7
8         List<String> palabras = new ArrayList<String>();
9         palabras.add("hola");
10        palabras.add("buenos");
11        palabras.add("días");
12        System.out.println(palabras.get(0));
13        palabras.set(0, "Adios");
14        System.out.println(palabras);
15        palabras.remove(1);
16        System.out.println(palabras);
17        palabras.remove(0);
18    }
19 }
20
21
22
```

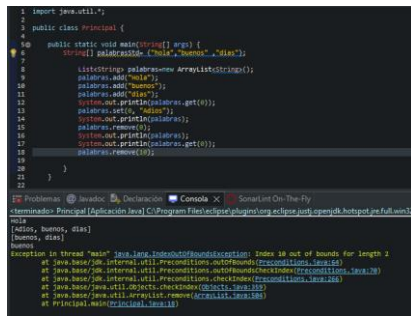


Ilustración 7 nombreArrayList.remove(pos)

Para eliminar todos los elementos del ArrayList se utiliza el método `clear()`.

```
7
8 List<String> palabras = new ArrayList<String>();
9 palabras.add("hola");
10 palabras.add("buenos");
11 palabras.add("días");
12 System.out.println(palabras.get(0));
13 palabras.set(0, "Adios");
14 System.out.println(palabras);
15 palabras.remove(0);
16 System.out.println(palabras);
17 System.out.println(palabras.get(0));
18 palabras.clear();
19 System.out.println(palabras);
20
21 }
22
```

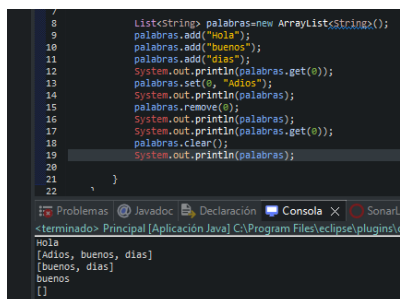


Ilustración 8 nombreArrayList.clear()

3.20.1.1.5 Tamaño del ArrayList.

Utilizamos el método `nombreArrayList.size()`;

3.20.1.1.6 Bucles para recorrer ArrayList.For y Foreach

```
1 import java.util.*;
2
3 public class Principal {
4
5     public static void main(String[] args) {
6
7         List<String> palabras = new ArrayList<String>();
8         palabras.add("hola");
9         palabras.add("buenos");
10        palabras.add("días");
11        for(int i=0; i<palabras.size(); i++) {
12            System.out.println("Pos["+i+"]="+palabras.get(i)+".");
13        }
14    }
15 }
16
17
```

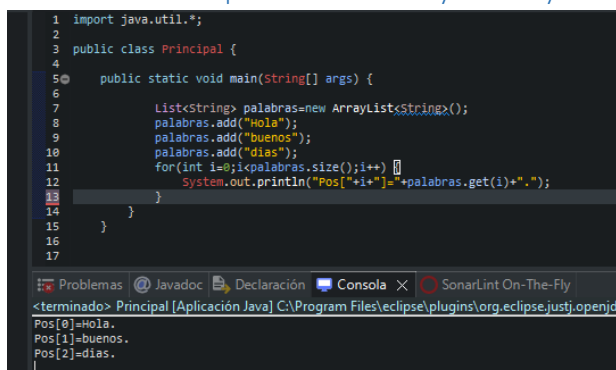


Ilustración 9 recorrer ArrayList Bucle For

```
1 import java.util.*;
2
3 public class Principal {
4
5     public static void main(String[] args) {
6
7         List<String> palabras=new ArrayList<String>();
8         palabras.add("Hola");
9         palabras.add("buenos");
10        palabras.add("dias");
11        for(String s:palabras) {
12            System.out.println(s);
13        }
14    }
15 }
16
17
```

Problemas Javadoc Declaración Consola X SonarLint On-The-Fly

<terminado> Principal [Aplicación Java] C:\Program Files\ eclipse\plugins\org.eclipse.justi.openjdk...

Hola
buenos
dias

Ilustración 10 bucle forEach

3.20.1.1.7 Ordenar un ArrayList.

Otra clase útil en java.útil es la clase Collections , que incluye sort() método que ordena listas alfabéticamente o numéricamente:

```
Principal.java X
1 import java.util.*;
2 import java.util.Collections;
3 public class Principal {
4
5     public static void main(String[] args) {
6
7         ArrayList<String> palabras=new ArrayList<String>();
8         palabras.add("hola");
9         palabras.add("buenos");
10        palabras.add("dias");
11        palabras.add("hola");
12        Collections.sort(palabras);
13        for(String s:palabras) {
14            System.out.println(s);
15        }
16    }
17 }
18
```

Problemas Javadoc Declaración Consola X SonarLint On-The-Fly

<terminado> Principal [Aplicación Java] C:\Program Files\ eclipse\plugins\org.eclipse.justi.openjdk...

hola
buenos
dias
hola

Ilustración 11 Ordenar Collections.sort()

Es sensible a Mayúsculas. Fíjate en el Hola.

Ordenamos un ArrayList de Enteros mediante la clase contenedora Integer y sort de Collection.

```
Principal.java X
1 import java.util.*;
2 import java.util.Collections;
3 public class Principal {
4
5     public static void main(String[] args) {
6
7         ArrayList<Integer> numeros=new ArrayList<Integer>();
8         numeros.add(100);
9         numeros.add(6);
10        numeros.add(1024);
11        numeros.add(-3);
12        Collections.sort(numeros);
13        for(Integer i:numeros) {
14            System.out.println(i);
15        }
16    }
17 }
18
```

Problemas Javadoc Declaración Consola X SonarLint On-The-Fly

<terminado> Principal [Aplicación Java] C:\Program Files\ eclipse\plugins\org.eclipse.justi.openjdk...

-3
6
100
1024

Ilustración 12 sort de una clase Integer

3.20.1.2 Java LinkedList.

Puede contener muchos objetos del mismo tipo como el ArrayList. La LinkedList tiene todos los mismos métodos que la clase ArrayList porque ambos métodos implementan la interfaz List. Esto quiere decir que todo lo anterior es válido, borrar, acceder etc...

Se puede usar de la misma manera pero se construyen de manera muy diferente.

Como funciona ArrayList.

La clase ArrayList tiene una matriz regular dentro de ella. Cuando se agrega un elemento, se coloca en la matriz. Si la matriz no es lo suficientemente grande, se crea una nueva matriz más grande para reemplazar la anterior y se elimina la anterior.

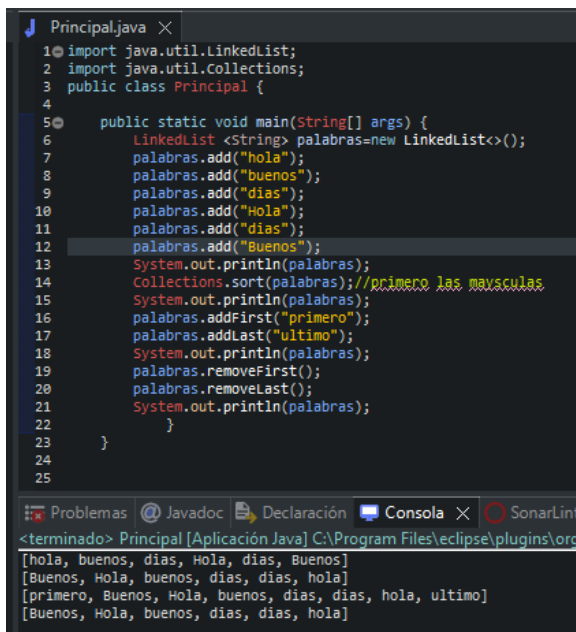
Como funciona LinkedList.

El LinkedList almacena sus artículos en “contenedores”. La lista tiene un enlace al primer contenedor y cada contenedor tiene un enlace al siguiente contenedor de la lista. Para agregar un elemento a la lista, el elemento se coloca en un nuevo contenedor y ese contenedor se vincula a uno de los contenedores de la lista.

Como regla Use [ArrayList](#) para [almacenar y acceder a datos](#), y [LinkedList](#) para [manipular datos](#).

Los métodos de LinkedList:

- [addFirst\(\)](#)
- [addLst\(\)](#)
- [removeFirst\(\)](#);
- [getFirst\(\)](#);
- [getLast\(\)](#);



```
Principal.java
1 import java.util.LinkedList;
2 import java.util.Collections;
3 public class Principal {
4
5     public static void main(String[] args) {
6         LinkedList<String> palabras=new LinkedList<>();
7         palabras.add("hola");
8         palabras.add("buenos");
9         palabras.add("dias");
10        palabras.add("Hola");
11        palabras.add("dias");
12        palabras.add("Buenos");
13        System.out.println(palabras);
14        Collections.sort(palabras); //primero las mayusculas
15        System.out.println(palabras);
16        palabras.addFirst("primero");
17        palabras.addLast("ultimo");
18        System.out.println(palabras);
19        palabras.removeFirst();
20        palabras.removeLast();
21        System.out.println(palabras);
22    }
23 }
24
25
```

Problemas Javadoc Declaración Consola SonarLint

<terminado> Principal [Aplicación Java] C:\Program Files\eclipse\plugins\org

[hola, buenos, dias, Hola, dias, Buenos]
[Buenos, Hola, buenos, dias, dias, hola]
[primero, Buenos, Hola, buenos, dias, dias, hola, ultimo]
[Buenos, Hola, buenos, dias, dias, hola]

Ilustración 13 métodos LinkedList.

- Listas
- Estos son los métodos que tienen las listas
- Las listas más comunes:
 - ArrayList
 - LinkedList

```
public static void main(String[] args) {
    List<String> customList = new ArrayList<>();
    customList.add("Libreta");
    customList.add("Bolígrafo");
    customList.add("Lápiz");

    for (String s: customList) {
        System.out.println(s + " está na posición " + customList.indexOf(s) );
    }
}
```

Libreta está na posición 0
Bolígrafo está na posición 1
Lápiz está na posición 2

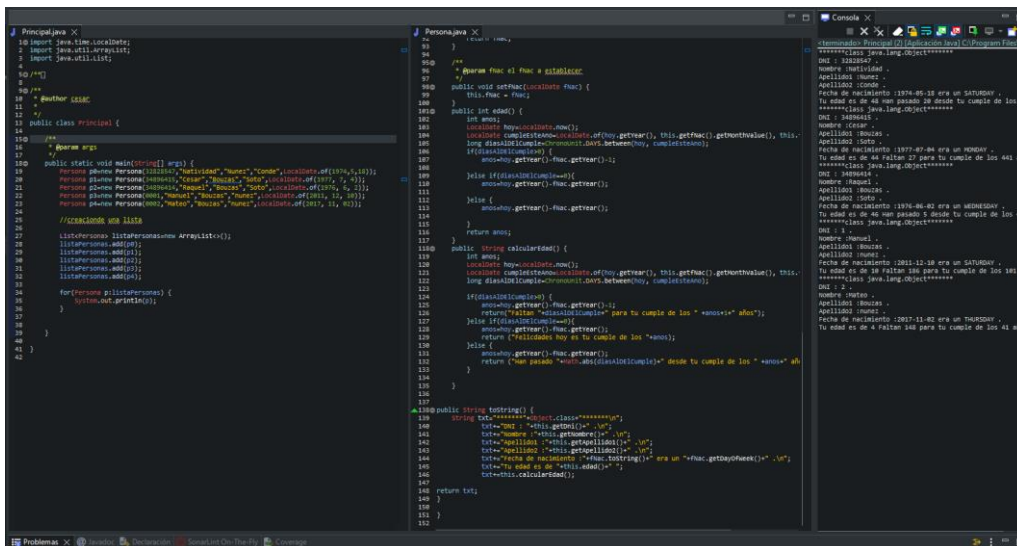
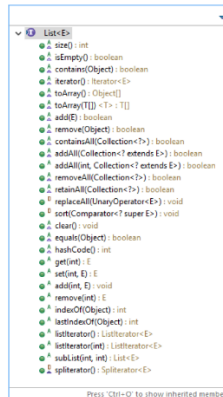


Ilustración 14 ArrayList personas

3.21 Ejercicio 21:

- 3.21.1 Crea un conjunto de strings (añade y elimina los que quieras), mete duplicados y muestra por pantalla lo que te queda del conjunto.

Los conjuntos no admiten como las listas elementos repetidos. Los conjuntos más comunes son HashSet y TreeSet.

La interfaz Set define una colección que no puede contener los elementos duplicados. Se encarga de gestionar conjuntos. Esta interfaz contiene, únicamente los métodos heredados de Collection añadiendo la restricción del almacenamiento de elementos duplicados.

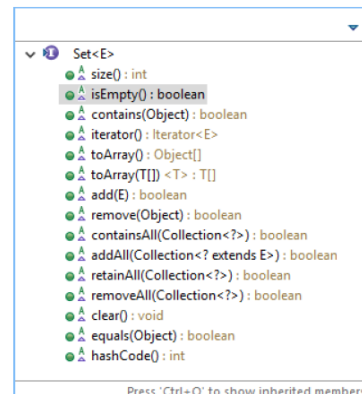
HashSet



Estructuras de almacenamiento: Colecciones

- Conjuntos
- Estos son los métodos que tienen los conjuntos
- Los conjuntos más comunes:
 - HashSet
 - TreeSet

```
public static void main(String[] args) {  
  
    Set<String> customSet = new HashSet<>();  
    customSet.add("Libreta");  
    customSet.add("Bolígrafo");  
    customSet.add("Lápiz");  
    customSet.remove("Bolígrafo");  
  
    for (String s: customSet) {  
        System.out.println(s);  
    }  
}
```



Libreta
Lápiz

3.21.1.1 Conjuntos HashSet.

Un HashSet es una colección de elementos donde cada elemento es único y se encuentra en java.util.

- Almacena los elementos en una tabla hash
- No garantiza ningún orden a la hora de realizar iteraciones
- Se debe definir el tamaño inicial.

- Es la implementación de Set con mejor rendimiento.

3.21.1.1.1 Creación de un objeto

HashSet<String> nombres=new HashSet <String>();

3.21.1.1.2 Añadir elementos add().

Los HashSet tienen muchos métodos y de ellos es add() para agregar elementos a la colección.

3.21.1.1.3 Remover Elemento remove(elemento).

Para eliminar use remove() , por ejemplo palabras.remove("cesar");

3.21.1.1.4 Eliminar todos los elementos método clear().

palabras.remove()

3.21.1.1.5 Tamaño del conjunto de HashSet. Método size();

Palabras.size();

3.21.1.1.6 Recorrer un HashSet.

```
for(String s:palabras){
```

```
    System.out.println(s);
```

```
}
```

3.21.1.1.7 Otros tipos .

Como en el caso anterior los tipos primitivos deben utilizar la clase contenedora ,Integer, Boolean,Character ,Double etc..

3.21.1.1.8 Realizo el ejercicio.

The screenshot shows a Java IDE with a file named 'conjuto.java'. The code defines a HashSet named 'palabras' and performs several operations: adding 'Cesar', 'Avion', and 'Tractor'; checking if 'Cesar' is present; printing the hash codes of all elements; creating an ArrayList from the set, sorting it, and printing it; clearing the set; and adding elements from the sorted ArrayList back to the set. The console output shows the results of these operations, including the hash codes and the sorted list of words.

```
17 System.out.println(palabras.add("Cesar")); //devuelve false ya existe Cesar
18 System.out.println( palabras.contains("Cesar")); //devuelve true ya que contiene cesar
19 System.out.println(palabras.add("Avion")); //devuelve true pues añade Avion
20 palabras.add("Tractor");
21
22 for(String s:palabras) {
23     System.out.println(s+"hashCode()->"+s.hashCode()+"");
24 }
25
26
27
28
29 ArrayList <String> listaPalabras=new ArrayList<>(palabras);
30 Collections.sort(listaPalabras);
31 System.out.println(listaPalabras);
32
33
34 palabras.clear(); //lo vacío
35 System.out.println(palabras); //lo imprimo vacío
36 palabras.addAll(listaPalabras); //lo meto todos las palabras ordenadas
37 System.out.println(palabras); //NO las muestra ordenada por el cálculo del hash de cada string vuelve a ser el mismo
38 System.out.print("Mateo".hashCode());
39 }
40
41
```

Console Output:

```
<terminado> Conjuto [Aplicación Java] C:\Program Files\ eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\
false
true
true
Mateohascode()->74115562,
Raquelhascode()->-1854313318,
Natividadhascode()->-995858796,
Avionhascode()->-63648659,
Pedrohascode()->-76998316,
Cesarhascode()->-64998434,
Manuelhascode()->-1997548446,
Tractorhascode()->-597266967,
[Avion, Cesar, Manuel, Mateo, Natividad, Pedro, Raquel, Tractor]
[]
[Mateo, Raquel, Natividad, Avion, Pedro, Cesar, Manuel, Tractor]
74115562
```

Ilustración 15 uso de HashSet e intento de ordenación

NO LO ORDENA!!! Fila 22 a 28. Por el cálculo del hashcode

3.21.1.2 TreeSet.

- Almacena los elementos ordenándolos en función de sus valores.
- Los elementos almacenados deben implementar la interfaz comparable
- Es la implementación con un rendimiento más lento.

3.22 Ejercicio 22: Crea una lista de strings y muestra por pantalla el elemento y la posición en la que se encuentra.

```
lista_String_Ordenada.java X
1 import java.util.LinkedList;
2 public class lista_String_Ordenada {
3
4
5     public static void main(String[] args) {
6         LinkedList<String> palabras=new LinkedList<>();
7         palabras.add("aaa");
8         palabras.add("trac");
9         palabras.add("tor");
10        palabras.add("vol");
11        palabras.add("ador");
12        palabras.add("aaa");
13        int i=0;
14        System.out.println("Con index.of()----->Con un contador----->>>HashCode");
15        for(String s:palabras) {
16            System.out.print("Pos["+s+palabras.indexOf(s)+"]="+"s+----->> ");
17            System.out.println("Pos["+s+(i++)+""]="+"s+----->>hashCode----->>s.hashCode());
18        }
19
20    }
21
22
23
24 }
25 }
```

Problemas Javadoc Declaración Consola SonarLint On-The-Fly

<terminado> lista_String_Ordenada [Aplicación Java] C:\Program Files\Eclipse\plugins\org.eclipse.justi.openjdk.hotspot...

```
Con index.of()----->Con un contador----->>>HashCode
Pos[0]= aaa ----->> Pos[0]= aaa ----->>hashCode ----->>96521
Pos[1]= trac ----->> Pos[1]= trac ----->>hashCode ----->>9568416
Pos[2]= tor ----->> Pos[2]= tor ----->>hashCode ----->>115031
Pos[3]= vol ----->> Pos[3]= vol ----->>hashCode ----->>116947
Pos[4]= ador ----->> Pos[4]= ador ----->>hashCode ----->>2989382
Pos[0]= aaa ----->> Pos[5]= aaa ----->>hashCode ----->>96521
```

Ilustración 16 lista de string con contador

POSICION OTRA MANERA sin contador???

3.23 Ejercicio 23: Crear una dupla clave-valor de Personas (la clase que creaste en el ejercicio 14), añade y quita elementos y muestra por pantalla los elementos que quedan.

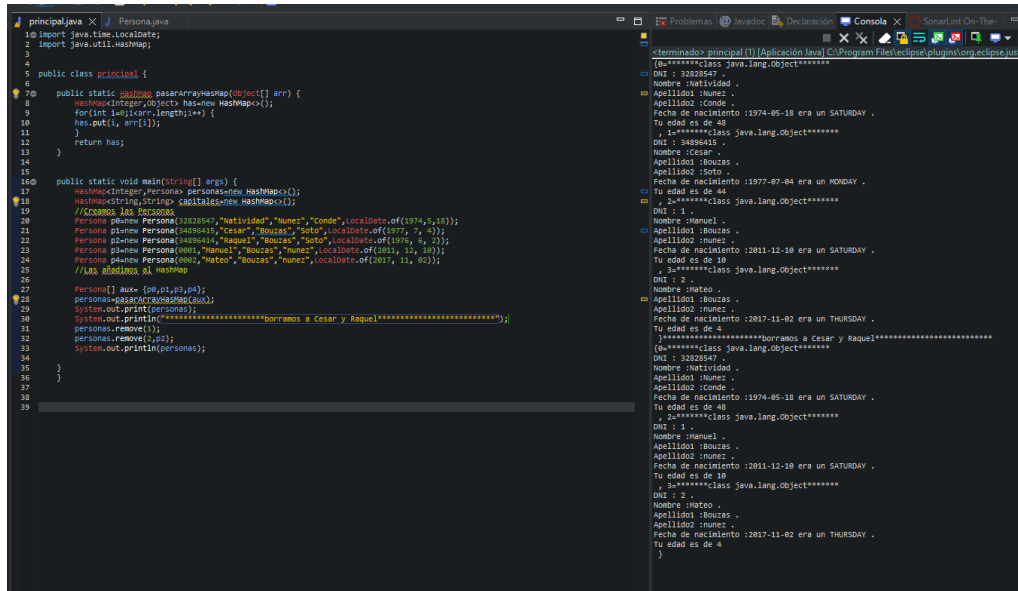
La clase `Map` permite definir colecciones de elementos que poseen pares de datos Clave-Valor. Cada Elemento tiene asociado una clave y un valor. La clave es utilizada para acceder de forma muy rápida a un elemento.

Java permite que la clave sea cualquier tipo de objeto pero se suele utilizar Integer o String.

Los mapas no permiten insertar objetos nulos ya que povocarían excepciones de tipo `NullPointerException`

3.23.1 Declaración.

HashMap<Integer,Persona> personas=new HashMap<>();



```
1 import java.time.LocalDate;
2 import java.util.HashMap;
3
4 public class principal {
5
6     public static void main(String[] args) {
7         HashMap<Integer,Persona> personas=new HashMap<>();
8         for(int i=0;i<arr.length;i++)
9             has.put(i, arr[i]);
10    }
11    return has;
12 }
13
14 public static void main(String[] args) {
15     HashMap<Integer,Persona> personas=new HashMap<>();
16     //Creamos las personas
17     Persona p0=new Persona(32828547,"Natividad","Nunez","Conde",LocalDate.of(1974,5,18));
18     Persona p1=new Persona(34896416,"Cesar","Bouzas","Soto",LocalDate.of(1977,7,4));
19     Persona p2=new Persona(34896416,"Raquel","Bouzas","Soto",LocalDate.of(1976,6,2));
20     Persona p3=new Persona(8001,"Manuel","Bouzas","Nunez",LocalDate.of(2011,11,18));
21     Persona p4=new Persona(8001,"Mateo","Bouzas","Nunez",LocalDate.of(2017,11,8));
22     //Las añadimos al HashMap
23     personas.put(p0.getId(),p0);
24     personas.put(p1.getId(),p1);
25     personas.put(p2.getId(),p2);
26     personas.put(p3.getId(),p3);
27     personas.put(p4.getId(),p4);
28     System.out.println(personas);
29     System.out.println("*****borramos a Cesar y Raquel*****");
30     personas.remove(p1.getId());
31     personas.remove(p2.getId());
32     System.out.println(personas);
33 }
34 }
35 }
36 }
37 }
38 }
39 }
```

Console Output:

```
Terminador principal () [Aplicación Java] C:\Program Files\ eclipse\plugins\org.eclipse.jdt...
[*****class java.lang.Object*****
ONT : 32828547
Nombre :Natividad
Apellido01 :Nunez
Apellido02 :Conde
Fecha de nacimiento :1974-05-18 era un SATURDAY
Tu edad es de 48
*****class java.lang.Object*****
ONT : 34896416
Nombre :Cesar
Apellido01 :Bouzas
Apellido02 :Soto
Fecha de nacimiento :1977-07-04 era un MONDAY
Tu edad es de 44
*****class java.lang.Object*****
ONT : 1
Nombre :Manuel
Apellido01 :Bouzas
Apellido02 :Nunez
Fecha de nacimiento :2011-12-18 era un SATURDAY
Tu edad es de 18
*****class java.lang.Object*****
ONT : 2
Nombre :Mateo
Apellido01 :Bouzas
Apellido02 :Nunez
Fecha de nacimiento :2017-11-02 era un THURSDAY
Tu edad es de 6
*****borramos a Cesar y Raquel*****
[*****class java.lang.Object*****
ONT : 32828547
Nombre :Natividad
Apellido01 :Nunez
Apellido02 :Conde
Fecha de nacimiento :1974-05-18 era un SATURDAY
Tu edad es de 48
*****class java.lang.Object*****
ONT : 1
Nombre :Manuel
Apellido01 :Bouzas
Apellido02 :Nunez
Fecha de nacimiento :2011-12-18 era un SATURDAY
Tu edad es de 18
*****class java.lang.Object*****
ONT : 2
Nombre :Mateo
Apellido01 :Bouzas
Apellido02 :Nunez
Fecha de nacimiento :2017-11-02 era un THURSDAY
Tu edad es de 6
]
}
```

3.23.2 Métodos destacados del HashMap:

- `Int size()`;
- `Boolean isEmpty()`;
- `Void put(k,v)` :inserta el elemento v con la clave k;
- `V get(K clave)`:Devuelve el valor de la clave que se pasa como parámetro , null si la clave no existe.
- `V remove(k clave)`:borrar el par clave valor;
- `Boolean containsKey(k clave)`:Devuelve true si existe la clave en el map
- `Boolean containsValue(v valor)`: Devuelve true si es map hay un Valor V;
- `Collection<v>values`: Devuelve una Collection con los valores del objeto Map.
- `Set<k> keySet()`:Devuelve un set con las claves del objeto Map.

3.23.3 Condiciones del Map(k,v)

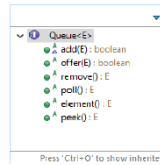
El conjunto de claves no puede repetir la clave es decir una estructura de tipo Set por lo que para detectar si la clave esta repetida deben definir correctamente `hashCode()` y `equals()`

3.24 Ejercicio 24: Colas.

Las colas son listas en la que los elementos se introduce y eliminan por diferentes extremos .

3.24.1 FIFO->Cola

- Colas
- Las colas son listas en las que los elementos se introducen y eliminan por diferentes extremos
- FIFO -> First In - First Out

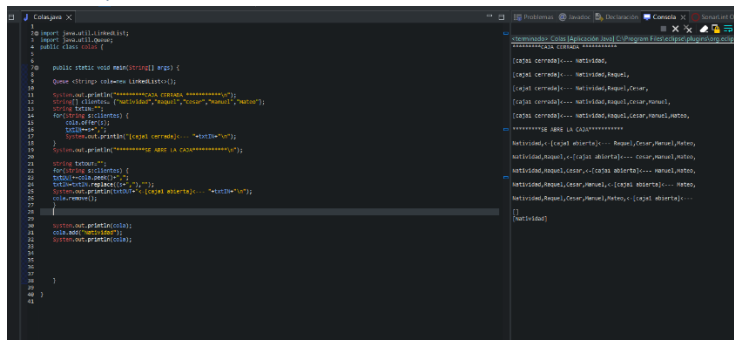


```
public static void main(String[] args) {  
    Queue<String> customQueue= new LinkedList<>();  
  
    customQueue.offer("Smith");  
    customQueue.offer("Montessori");  
    customQueue.offer("Peralta");  
    customQueue.offer("House");  
  
    System.out.println(customQueue.peek()+"\n");  
  
    while (!customQueue.isEmpty()) {  
        System.out.println(customQueue.poll());  
    }  
}
```

Smith
Smith
Montessori
Peralta
House

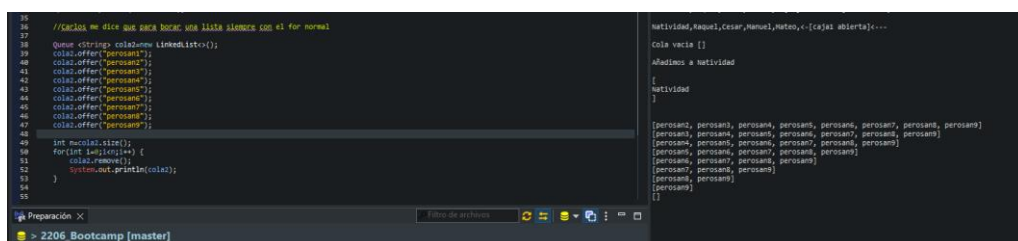
- Add() añade un elemento a la lista (posibles errores si tenemos limitación de tamaño)
- Offer() añade un elemento al final de la lista si es posible , preferible al uso de add.
- Remove() Borrar el primer elemento de la cola, que lleva mas tiempo esperando.
-

3.24.2 Crea una cola, añade/quita elementos y muestra lo que quede por pantalla .



3.24.3 Recorre la cola y elimina todos sus elementos.

Mejor recorrer con el for normal, el foreach se vuelve loco.



Se puede hacer sin sacar fuera el size() con un for???

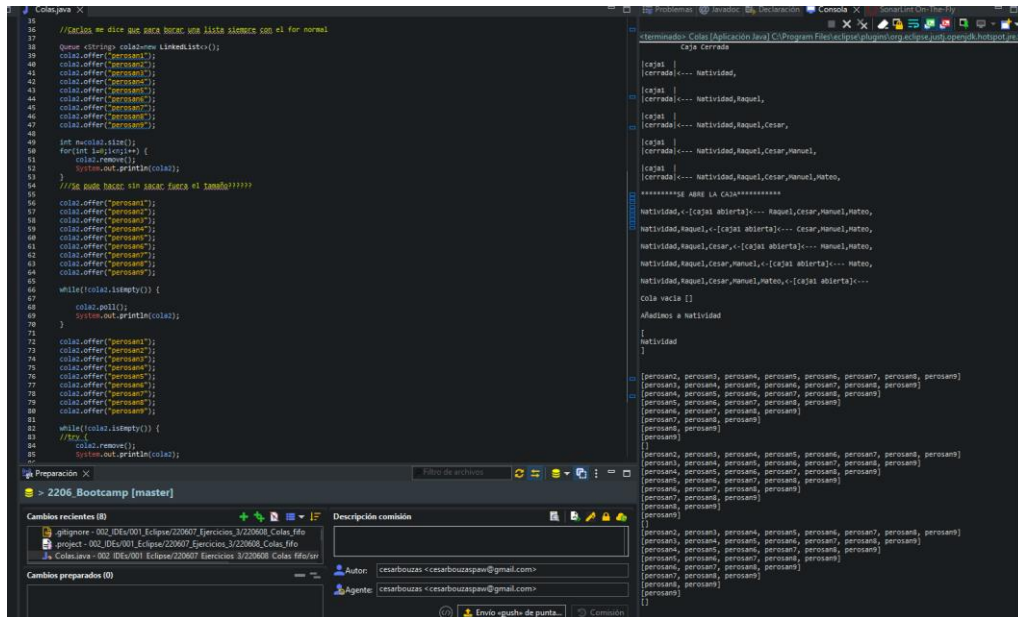


Ilustración 17 modos de borrar una cola con while

3.25 Ejercicio 25: Pilas

3.25.1 Crea una pila, añade/quita elementos y muestra lo que quede por pantalla .

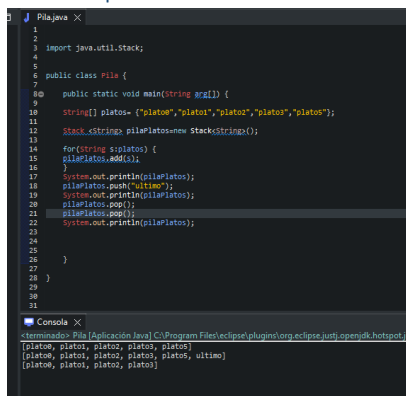
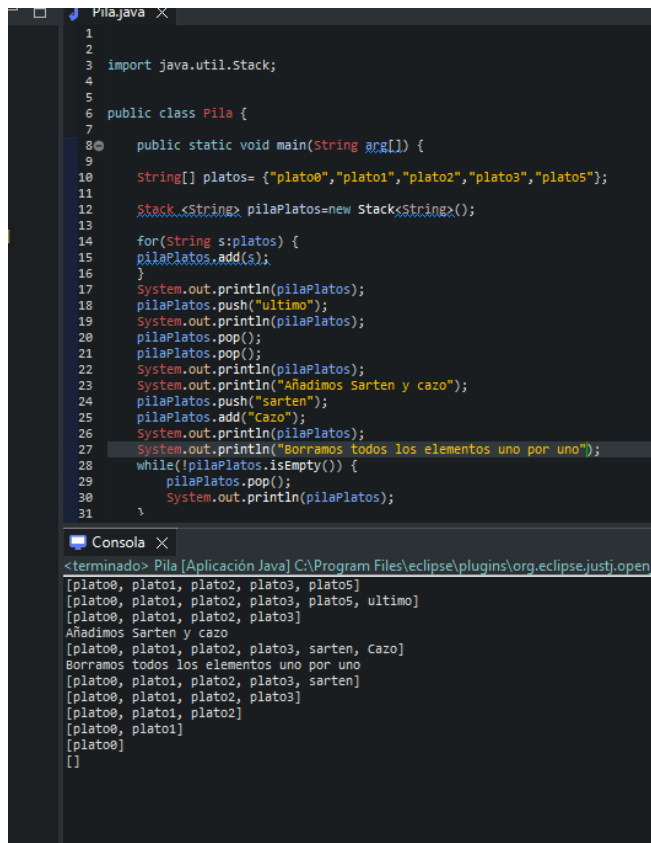


Ilustración 18 Uso de Pila

3.25.2 Recorre la pila y elimina todos sus elementos.



```
1
2
3 import java.util.Stack;
4
5 public class Pila {
6
7     public static void main(String arg[]) {
8
9         String[] platos= {"plato0","plato1","plato2","plato3","plato5"};
10
11         Stack<String> pilaPlatos=new Stack<String>();
12
13         for(String s:platos) {
14             pilaPlatos.add(s);
15         }
16         System.out.println(pilaPlatos);
17         pilaPlatos.push("ultimo");
18         System.out.println(pilaPlatos);
19         pilaPlatos.pop();
20         pilaPlatos.pop();
21         System.out.println(pilaPlatos);
22         System.out.println("Añadimos Sarten y cazo");
23         pilaPlatos.push("sarten");
24         pilaPlatos.add("cazo");
25         System.out.println(pilaPlatos);
26         System.out.println("Borramos todos los elementos uno por uno");
27         while(!pilaPlatos.isEmpty()) {
28             pilaPlatos.pop();
29             System.out.println(pilaPlatos);
30         }
31     }
```

Console Output:

```
<terminado> Pila [Aplicación Java] C:\Program Files\eclipse\plugins\org.eclipse.justj.openj
[plato0, plato1, plato2, plato3, plato5]
[plato0, plato1, plato2, plato3, plato5, ultimo]
[plato0, plato1, plato2, plato3]
Añadimos Sarten y cazo
[plato0, plato1, plato2, plato3, sarten, cazo]
Borramos todos los elementos uno por uno
[plato0, plato1, plato2, plato3, sarten]
[plato0, plato1, plato2, plato3]
[plato0, plato1, plato2]
[plato0, plato1]
[plato0]
[]
```

3.26 Ejercicio 26 XML y JSON

3.26.1 Crea un xml de los componentes de un coche, indicando la cantidad de cada uno de ellos .

El lenguaje XML es un lenguaje de marcado extensible (eXtensible Markup Language).Es legible tanto por humanos como por máquinas independientemente de la plataforma , sirve para intercambiar información entre sistemas.

3.26.2 Crea un json de los componentes de un coche, indicando la cantidad de cada uno de ellos .

Json es un formato ligero de intercambio de datos JavaScript Object Notation. Es simple de interpretarse y generarse por máquinas.

Formado por pares Claver-Valor.

Puede utilizarse independientemente del lenguaje Javascript.

Se usa a menudo cuando los datos se envían desde un servidor a una página web.

Es independiente del idioma.

Es autodescritpivo y fácil de entender.

Los datos están separados por comas.

Las llaves sostienen objetos.

Los corchetes contienen matrices.

Ejemplo JSON

```
{
  "employees": [
    { "firstName": "John", "lastName": "Doe" },
    { "firstName": "Anna", "lastName": "Smith" },
    { "firstName": "Peter", "lastName": "Jones" }
  ]
}
```

3.26.2.1 Datos JSON : un nombre y un valor

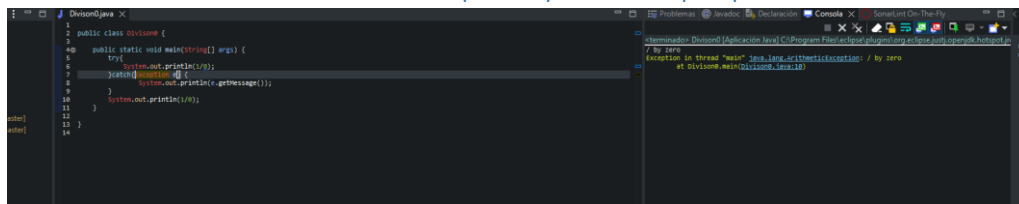
Los datos Json se escriben como nombre/valor , al igual que las propiedades de objeto de JavaScript.

Un par de nombre /valor consta de un nombre de campos (entre comillas dobles) . seguido de dos puntos, seguido de un valor:

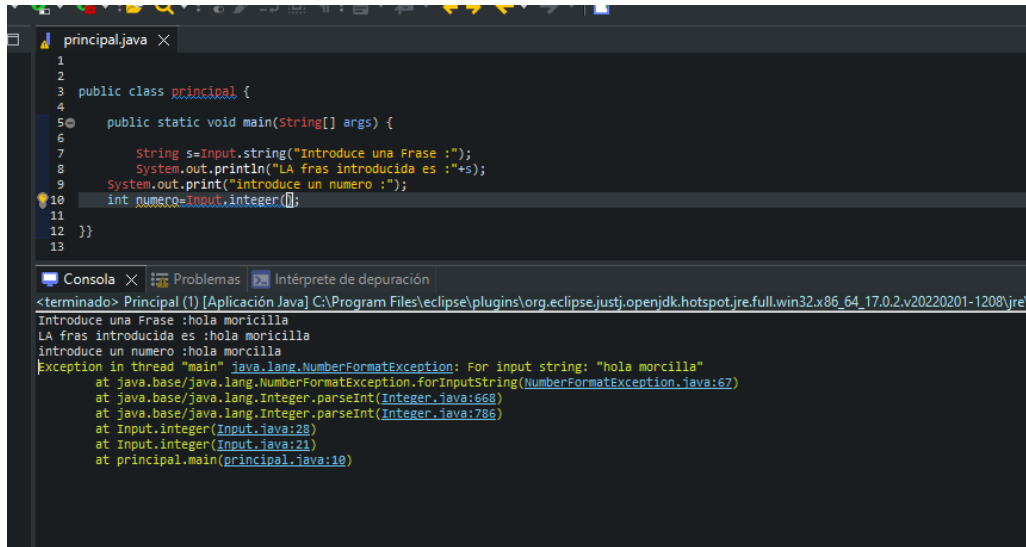
“firstName” : “Jhon”

Los nombres de Json requieren comillas dobles . Los nombres de JavaScript no.

3.27 Intenta hacer una división por 0 y mira lo que pasa .



3.28 Solicita por pantalla un número (con la clase Input) y mira qué pasa si lo que introduces es un string.



```
principal.java ×
1
2
3 public class principal {
4
5     public static void main(String[] args) {
6
7         String s=Input.string("Introduce una Frase :");
8         System.out.println("LA fras introducida es :"+s);
9         System.out.print("introduce un numero :");
10        int numero=Input.integer();
11    }
12 }
13

Consola × Problemas Intérprete de depuración
<terminado> Principal (1) [Aplicación Java] C:\Program Files\eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\
Introduce una Frase :hola morcilla
LA fras introducida es :hola morcilla
introduce un numero :hola morcilla
Exception in thread "main" java.lang.NumberFormatException: For input string: "hola morcilla"
    at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:67)
    at java.base/java.lang.Integer.parseInt(Integer.java:668)
    at java.base/java.lang.Integer.parseInt(Integer.java:786)
    at Input.integer(Input.java:28)
    at Input.integer(Input.java:21)
    at principal.main(principal.java:10)
```

No funciona el ejemplo , Clase Input??? De donde la saco?? Apuntes de SQL