

Rellena con las palabras clave correctas, escribe la cadena VACIO si no es necesario que aparezca ninguna palabra clave

```
class Ejemplo {  
    private void F(int x) throws   {  
        if (x < 0) {  
               Exception();  
        }  
    }  
}
```

Rellena con las palabras clave correctas, escribe la cadena VACIO si no es estrictamente necesario añadir otra palabra clave

```
public class Ejemplo {  
    public static void main(String[] args) VACIO ✓ VACIO ✓ {  
        FileOutputStream fos = null;  
  
        try {  
            fos = new FileOutputStream("unfichero.txt");  
  
            fos.write(65);  
            fos.flush();  
  
        } catch (IOException ex ✓ ) {  
            System.out.print("Ha ocurrido un error de E/S: " + ex.getMessage());  
        } finally ✓ {  
            if(fos!=null) {  
                fos.close();  
            }  
        }  
    }  
}
```





Rellena con las palabras clave correctas, no escribas nada si no es necesario

```
class MiExcepcion extends Exception {  
    private int causante;  
    public MiExcepcion(int x) {  
        this.causante = x;  
    }  
    public int getCausante() {  
        return causante;  
    }  
}
```

```
class Ejemplo {  
    private void F(int x) throws MiExcepcion {  
        if (x <= 0) {  
            throw new MiExcepcion ( x );  
        }  
        // haz alguna cosa si no x es positiva  
    }  
}
```

Rellena con las palabras clave correctas, escribe la cadena VACIO si no es estrictamente necesario añadir otra palabra clave

```
class MiExcepcion extends RuntimeException {  
    private int causante;  
    public MiExcepcion(int x) {  
        this.causante = x;  
    }  
    public int getCausante() {  
        return causante;  
    }  
}
```

```
class Ejemplo {  
    private void F(int x) VACIO  {  
        if (x <= 0) {  
            throw  new  MiExcepcion  (x);  
        }  
        // haz alguna cosa si no x es positiva  
    }  
}
```

Rellena con las palabras clave correctas, escribe la cadena VACIO si no es estrictamente necesario añadir otra palabra clave

```
class MiExcepcion extends Exception {
    private int causante;
    public MiExcepcion(int x) {
        this.causante = x;
    }
    public int getCausante() {
        return causante;
    }
}
```

```
class Ejemplo {
    private void F(int x) throws MiExcepcion {
        if (x <= 0) {
            throw new MiExcepcion (x);
        }
        // haz alguna cosa si no x es positiva
    }

    private void G(int y) throws MiExcepcion {
        if (y * y > 100) {
            F(x);
        }
    }
}
```

Rellena con las palabras clave correctas, escribe la cadena VACIO sino es estrictamente necesario añadir otra palabra clave

```
class MiExcepcion extends Exception {  
    private int causante;  
    public MiExcepcion(int x) {  
        this.causante = x;  
    }  
    public int getCausante() {  
        return causante;  
    }  
}
```

```
class Ejemplo {  
    private void F(int x) throws MiExcepcion {  
        if (x <= 0) {  
            throw new MiExcepcion(x);  
        }  
        // haz alguna cosa si no x es positiva  
    }  
  
    private void G(int y) VACIO VACIO {  
        try {  
            if (y * y > 100) {  
                F(y);  
            }  
        } catch (MiExcepcion e) {  
            System.err.println("El valor introducido debe ser positivo y es: " + e.getCausante());  
        }  
    }  
}
```


Rellena con las palabras clave correctas, escribe la cadena VACIO si no es estrictamente necesario añadir otra palabra clave

```
class MiExcepcion extends Exception {  
    private int causante;  
    public MiExcepcion(int x) {  
        this.causante = x;  
    }  
    public int getCausante() {  
        return causante;  
    }  
}
```

```
class Ejemplo {  
    int [] v;  
  
    public Ejemplo(int tamanyo) {  
        v = new int[tamanyo];  
    }  
  
    private void F(int indice, int valor) throws MiExcepcion {  
        try {  
            v[indice] = valor;  
        } catch (ArrayIndexOutOfBoundsException e) { // capturamos que el índice se sale del array  
            throw new MiExcepcion (indice);  
        }  
        // haz alguna cosa si no x es positiva  
    }  
  
    private void G(int indice, int y) throws MiExcepcion {  
        try {  
            if (y * y > 100) {  
                F(indice, y);  
            }  
        } catch (MiExcepcion e) {  
            int indiceMal = e.getCausante ();  
            System.err.println("El valor introducido debe ser positivo y es: " + indiceMal);  
            throw new MiExcepcion(indice);  
        }  
    }  
}
```