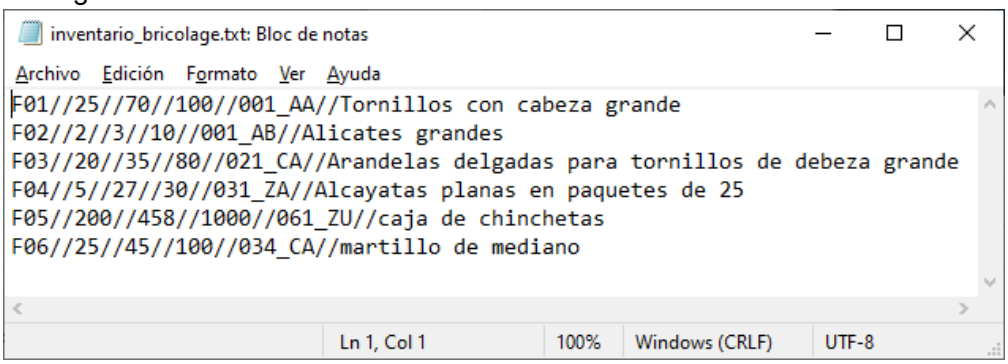
Bricolage

Datos personales

- **Apellidos:** .....
- **Nombre:** .....
- **Email:** .....
- **DNI:** .....

Gestión del inventario de un almacén de bricolage

Necesitamos gestionar el inventario de una tienda de bricolage. Inicialmente, los artículos están registrados en un archivo como el siguiente:



Se trata de un archivo de texto (llamado “inventario_bricolage.txt”). En él, que cada línea contiene la siguiente información referida a las existencias de un producto concreto:

- El código del producto
- La cantidad mínima que debería haber siempre, para atender la demanda previsible
- Las existencias en ese momento
- La cantidad máxima de dicho producto que cabe en los estantes del supermercado
- La ubicación, es decir, el código del estante en la tienda
- Una descripción concisa pero suficiente

Cada dato de los descritos está separado de los datos contiguos por dos barras, así: `//` .

1. Definición de la clase `Articulo`

Define una clase Articulo, cuyos atributos serán:

- La cantidad mínima que debería haber siempre, para atender la demanda previsible
- La cantidad máxima de dicho producto que cabe en los estantes del supermercado
- La ubicación, es decir, el código del estante en la tienda
- Una descripción concisa pero suficiente

Fíjate en que no se incluye el código del producto, ni tampoco la cantidad que hay en cada momento realmente en el almacén.

Las operaciones básicas para empezar serán únicamente las siguientes:

- La de creación (`__init__`), que creará el diccionario a partir del archivo de datos descrito
- Un método (`__str__`) que convierte la descripción de un artículo en un string.

Nota. En este apartado, se valorará también la documentación aportada. Pon atención a documentar adecuadamente la clase definida.

Nota. El funcionamiento será como el que se muestra seguidamente. Esto **no** debe modificarse.

In [1]:

```
# Este apartado debe ser completado por el estudiante
```

In [2]:

```
articulo = Articulo(25, 100, "001_AA", "Tornillos con cabeza grande")
print(articulo)
```

```
<25 100 001_AA 'Tornillos con cabeza grande'>
```

2. Creación de un diccionario con todos los Artículos

Define ahora una función que cargue los datos del archivo en un diccionario, cuya clave es el código de un artículo y cuyo valor asociado, la descripción de dicho artículo recogida en la clase anterior.

Nota. En este apartado, se valorará también la documentación aportada. Po atención a documentar adecuadamente la clase definida.

Nota. El funcionamiento será como el que se muestra seguidamente. Esto **no** debe modificarse.

In [3]:

```
# Este apartado debe ser completado por el estudiante
```

In [4]:

```
inventario, existencias = lectura_datos("inventario_bricolage.txt")
for k, v in inventario.items():
    print(k, existencias[k], v)
```

```
F01 70 <25 100 001_AA 'Tornillos con cabeza grande'>
F02 3 <2 10 001_AB 'Alicates grandes'>
F03 35 <20 80 021_CA 'Arandelas delgadas para tornillos de debeza grande'>
F04 27 <5 30 031_ZA 'Alcayatas planas en paquetes de 25'>
F05 458 <200 1000 061_ZU 'caja de chinchetas'>
F06 45 <25 100 034_CA 'martillo de mediano'>
```

3. Definición de la clase `Almacen`

Define una clase para llevar todo el almacén. Esta clase tendrá *dos* atributos:

- Un diccionario con las descripciones de los `productos`, salvo las existencias en cada momento
- Un segundo diccionario con las `existencias` de cada producto

En ambos diccionarios, la clave será el código del producto, que obviamente ya no se repetirá en la descripción.

La operación básica para empezar será un único método:

- La de creación (`__init__`), que creará los diccionarios a partir del archivo de datos descrito
- La operaciónUn `__str__` típica.

Nota. El funcionamiento será como el que se muestra seguidamente. Esto **no** debe modificarse.

In [5]:

```
# Este apartado debe ser completado por el estudiante
```

In [6]:

```
almacen = Almacen("inventario_bricolage.txt")
print(almacen)
```

```
F01 70 -> <25 100 001_AA 'Tornillos con cabeza grande'>
F02 3 -> <2 10 001_AB 'Alicates grandes'>
F03 35 -> <20 80 021_CA 'Arandelas delgadas para tornillos de debeza grand
e'>
F04 27 -> <5 30 031_ZA 'Alcayatas planas en paquetes de 25'>
F05 458 -> <200 1000 061_ZU 'caja de chinchetas'>
F06 45 -> <25 100 034_CA 'martillo de mediano'>
```

4. Ventas

Añade ahora una operación para vender un artículo, en la cantidad que se indique, con la consecuente disminución de esa cantidad en las existencias de dicho artículo. Para esta operación hay que tener en cuenta lo siguiente:

- Si esta cantidad rebasara las existencias, esa venta **no** se efectúa.

Nuevamente, copia la clase definida en el apartado anterior y añade el método de venta de un artículo, teniendo en cuenta el test de funcionamiento que se propone justo a continuación.

In [7]:

```
# Este apartado debe ser completado por el estudiante
```

Nota. El funcionamiento será como el que se muestra seguidamente. Esto **no** debe modificarse.

In [8]:

```
almacen = Almacen("inventario_bricolage.txt")
print(almacen)

print(almacen.datos_producto("F06"))
print(almacen.cantidad_producto("F06"))

almacen.vender("F06", 30)
print(almacen.cantidad_producto("F06"))

almacen.vender("F06", 30)
print(almacen.cantidad_producto("F06"))
```

```
F01 70 -> <25 100 001_AA 'Tornillos con cabeza grande'>
F02 3 -> <2 10 001_AB 'Alicates grandes'>
F03 35 -> <20 80 021_CA 'Arandelas delgadas para tornillos de debeza grand
e'>
F04 27 -> <5 30 031_ZA 'Alcayatas planas en paquetes de 25'>
F05 458 -> <200 1000 061_ZU 'caja de chinchetas'>
F06 45 -> <25 100 034_CA 'martillo de mediano'>

<25 100 034_CA 'martillo de mediano'>
45
15
15
```

5. Ventas, mejorado

Consideramos seguidamente dos mejoras, que nos facilitarán la posterior gestión de nuestro inventario:

- ** Mejora 1**. Nos gustaría anotar también la colección de `articulos_anomalos` que, al pasar por caja, presentan la anomalía de que se intenta comprar una cantidad superior a la que tenemos regitrada.
- ** Mejora 2**. Igualmente, podemos llevar otra relación de `artículos_para_reponer` , con los que precisan reposición, por haber rebasado el umbral mínimo.

Las mejoras descritas requieren modificar nuestra clase en dos aspectos:

- La operación `__init__` , que ha de registrar inicialmente ambas relaciones, y te pido que lo implementes mediante sendos conjuntos
- La operación de venta, que añadirá un artículo al conjunto de `articulos_anomalos` cuando se requiera, y al de `artículos_para_reponer` cuando sea necesario.

Redefine la clase y plantea tú los tests que consideres conveniente.

In [9]:

```
# Este apartado debe ser completado por el estudiante
```

In [10]:

```
# Tests de funcionamiento:

# Este apartado debe ser completado por el estudiante
```

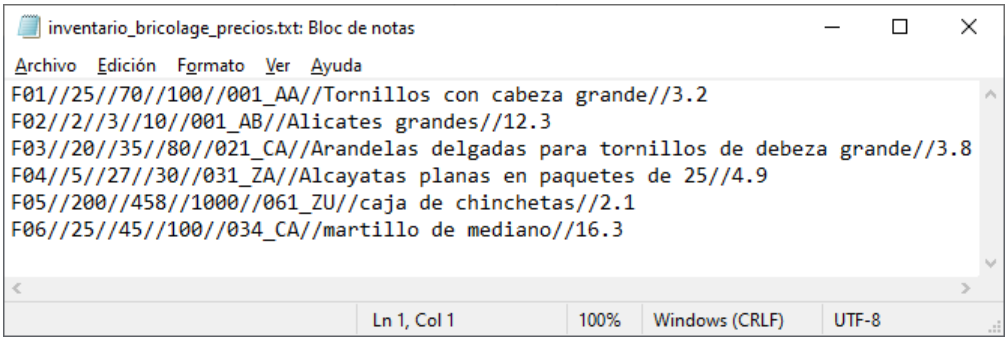
```
Artículos:
F01 70 -> <25 100 001_AA 'Tornillos con cabeza grande'>
F02 3 -> <2 10 001_AB 'Alicates grandes'>
F03 35 -> <20 80 021_CA 'Arandelas delgadas para tornillos de debeza grand
e'>
F04 27 -> <5 30 031_ZA 'Alcayatas planas en paquetes de 25'>
F05 458 -> <200 1000 061_ZU 'caja de chinchetas'>
F06 45 -> <25 100 034_CA 'martillo de mediano'>

Anomalías: set()
Reposiciones: set()
<25 100 034_CA 'martillo de mediano'>
45
15
15
Artículos:
F01 70 -> <25 100 001_AA 'Tornillos con cabeza grande'>
F02 3 -> <2 10 001_AB 'Alicates grandes'>
F03 35 -> <20 80 021_CA 'Arandelas delgadas para tornillos de debeza grand
e'>
F04 27 -> <5 30 031_ZA 'Alcayatas planas en paquetes de 25'>
F05 458 -> <200 1000 061_ZU 'caja de chinchetas'>
F06 15 -> <25 100 034_CA 'martillo de mediano'>

Anomalías: {'F06'}
Reposiciones: {'F06'}
```

6. Balance

Disponemos ahora de un campo adicional en el archivo `inventario_bricolage_precios.txt` , con el precio de cada artículo, un real, en euros:



Efectúa los cambios necesarios para que nuestro inventario incorpore dicho dato nuevo, y añade una operación que calcule el balance del almacén en un instante, sumando para cada artículo el producto de su precio por su cantidad.

In [11]:

```
# Este apartado debe ser completado por el estudiante
```

Nota. El test de funcionamiento siguiente **no** debe modificarse.

In [12]:

```
almacen = Almacen("inventario_bricolage_precios.txt")
print(almacen)

print(almacen.datos_producto("F06"))
print(almacen.cantidad_producto("F06"))

almacen.vender("F06", 10)
print(almacen.cantidad_producto("F06"))

almacen.vender("F06", 10)
print(almacen.cantidad_producto("F06"))

print(almacen)
print("Total balance: ", round(almacen.balance(), 1), "euros")
```

```
Artículos:
F01 70 -> <25 100 001_AA 'Tornillos con cabeza grande' 3.2€>
F02 3 -> <2 10 001_AB 'Alicates grandes' 12.3€>
F03 35 -> <20 80 021_CA 'Arandelas delgadas para tornillos de debeza grande'
3.8€>
F04 27 -> <5 30 031_ZA 'Alcayatas planas en paquetes de 25' 4.9€>
F05 458 -> <200 1000 061_ZU 'caja de chinchetas' 2.1€>
F06 45 -> <25 100 034_CA 'martillo de mediano' 16.3€>

Anomalías: set()
Reposiciones: set()
<25 100 034_CA 'martillo de mediano' 16.3€>
45
35
25
Artículos:
F01 70 -> <25 100 001_AA 'Tornillos con cabeza grande' 3.2€>
F02 3 -> <2 10 001_AB 'Alicates grandes' 12.3€>
F03 35 -> <20 80 021_CA 'Arandelas delgadas para tornillos de debeza grande'
3.8€>
F04 27 -> <5 30 031_ZA 'Alcayatas planas en paquetes de 25' 4.9€>
F05 458 -> <200 1000 061_ZU 'caja de chinchetas' 2.1€>
F06 25 -> <25 100 034_CA 'martillo de mediano' 16.3€>

Anomalías: set()
Reposiciones: set()
Total balance:  1895.5 euros
```