**Tarea07**

**Nombre alumno o alumna:** Podadera González, Andrés Samuel

**Fecha de entrega:** jueves, 2 de marzo de 2023, 16:16

**Retroalimentación global de la actividad:**

Hola, Andrés. Todo correcto. Muy bien. Excelente trabajo. Has sido el primero en entregar esta tarea.
Tan solo te he indicado un par de detalles que quizá se podrían mejorar.
La próxima unidad se abrirá a mediados de mes. Mientras tanto, puedes descansar un poco y ponerte con el trabajo pendiente de otros módulos.
Hasta pronto y enhorabuena de nuevo por tu trabajo.

**Puntuación orientativa de la actividad:** **10,00**

## RESULTADOS DE APRENDIZAJE EVALUADOS EN ESTA ACTIVIDAD

**RA6** ***Escribe programas que manipulen información seleccionando y utilizando tipos avanzados de datos.***

| | | Punt. Máx | Punt. Obt. | Punt. Final (sobre 10) |
|---|---|---|---|---|
| RA6.b | Se han reconocido las librerías de clases relacionadas con tipos de datos avanzados. | 66 | 66,00 | **10,0** |
| RA6.c | Se han utilizado listas para almacenar y procesar información. | 81 | 81,00 | **10,0** |
| RA6.d | Se han utilizado iteradores para recorrer los elementos de las listas. | 79 | 79,00 | **10,0** |
| RA6.e | Se han reconocido las características y ventajas de cada una de la colecciones de datos disponibles. | 64 | 64,00 | **10,0** |
| RA6.f | Se han creado clases y métodos genéricos. | 30 | 30,00 | **10,0** |

## CORRECCIÓN POR EJERCICIO/ACTIVIDAD/ELEMENTO

### Ejercicio 1. Ticket: importe y número de artículos.

| Puntos de control (elementos evaluables) | | Punt. Máx. | Punt. Obt. | CE | CE | CE | CE | CE | CE |
|---|---|---|---|---|---|---|---|---|---|
| *PC1* | *Método `getImporte`. Se recorre adecuadamente la lista de objetos `Compra`.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC2* | *Método `getImporte`. Se calcula correctamente el importe del ticket.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC3* | *Método `getNumArticulos`. Se recorre adecuadamente la lista de objetos `Compra`.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC4* | *Método `getNumArticulos`. Se calcula correctamente el número de artículos del ticket.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC5* | *No se incluyen elementos extraños, innecesarios o sin sentido.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC6* | *No faltan elementos imprescindibles o solicitados en el enunciado.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |

**Retroalimentación:**

Correcto.

### Ejercicio 2. Tienda: total de artículos y facturación.

| Puntos de control (elementos evaluables) | | Punt. Máx. | Punt. Obt. | CE | CE | CE | CE | CE | CE |
|---|---|---|---|---|---|---|---|---|---|
| *PC1* | *Método `getFacturacionTotal`. Se recorre adecuadamente la lista de objetos `TicketCompra`.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC2* | *Método `getFacturacionTotal`. Se calcula correctamente el importe de la facturación total.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC3* | *Método `getNumArticulosVendidos`. Se recorre adecuadamente la lista de objetos `TicketCompra`.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC4* | *Método `getNumArticulosVendidos`. Se calcula correctamente el número de artículos vendidos en la tienda.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC5* | *No se incluyen elementos extraños, innecesarios o sin sentido.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC6* | *No faltan elementos imprescindibles o solicitados en el enunciado.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |

**Retroalimentación:**

Correcto.

### Ejercicio 3. Productos vendidos a partir de una fecha (método `getProductosComprados`).

| Puntos de control (elementos evaluables) | | Punt. Máx. | Punt. Obt. | CE | CE | CE | CE | CE | CE |
|---|---|---|---|---|---|---|---|---|---|
| *PC1* | *Declaración e instanciación de objetos `Set`.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |
| *PC2* | *Recorrido de la lista de objetos `TicketCompra`.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC3* | *Localización de los tickets que cumplen la condición de fecha.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC4* | *Rellenado del conjunto de productos resultado (objeto `Set`).* | 4 | 4,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC5* | *No se incluyen elementos extraños, innecesarios o sin sentido.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |
| *PC6* | *No faltan elementos imprescindibles o solicitados en el enunciado.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |

**Retroalimentación:**

Correcto.

**Ejercicio 4. Importes por fechas (método *getImportesPorFechas*).**

| Puntos de control (elementos evaluables) | | Punt. Máx. | Punt. Obt. | CE | CE | CE | CE | CE | CE |
|---|---|---|---|---|---|---|---|---|---|
| *PC1* | *Declaración e instanciación de objetos `Map`.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |
| *PC2* | *Recorrido de la lista de objetos `TicketCompra`.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC3* | *Localización de los tickets que cumplen la condición de fecha.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC4* | *Rellenado del conjunto de productos resultado (objeto `Map`).* | 4 | 4,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC5* | *No se incluyen elementos extraños, innecesarios o sin sentido.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |
| *PC6* | *No faltan elementos imprescindibles o solicitados en el enunciado.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |

**Retroalimentación:**

El bucle de las líneas 198-200 es innecesario. Ya puedes obtener directamente el importe de un *TicketCompra* sin tener que recorrer cada comprar e ir sumando. Es algo que tú mismo ya has resuelto antes (método *getImporte*).

**Ejercicio 5. Clasificación de compras por vendedor y por año (*getComprasPorVendedorYear*).**

| Puntos de control (elementos evaluables) | | Punt. Máx. | Punt. Obt. | CE | CE | CE | CE | CE | CE |
|---|---|---|---|---|---|---|---|---|---|
| *PC1* | *Declaración e instanciación de objetos `Map` dobles.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |
| *PC2* | *Recorrido de la lista de objetos `TicketCompra`.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC3* | *Localización de los tickets que cumplen la condición de fecha.* | 1 | 1,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC4* | *Rellenado del conjunto de productos resultado (objeto `Map` doble).* | 4 | 4,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC5* | *No se incluyen elementos extraños, innecesarios o sin sentido.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |
| *PC6* | *No faltan elementos imprescindibles o solicitados en el enunciado.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |

**Retroalimentación:**

Igual que en el caso anterior, podrías haberte ahorrado el bucle de las líneas 224-226, pues la clase *Ticket* ya dispone de método *getImporte()*.
A la hora de hacer el new, si indicas en la declaración lo que hay dentro de la estructura, puedes usar directamente el operador “diamente” (<>) en el *new*:
*Map<String, Map<Integer, Double>> comprasPorVendedorEnAnyo = new HashMap<>();*

| **Ejercicio 6. Borrado de tickets (métodos *removeCompras*).** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Puntos de control (elementos evaluables)** | | **Punt. Máx.** | **Punt. Obt.** | **CE** | **CE** | **CE** | **CE** | **CE** | **CE** |
| *PC1* | *Método `removeCompras(String vendedor)`. Recorrido de la lista de objetos `TicketCompra` mediante iteradores.* | 4 | 4,0 | RA6.c | RA6.d | | | | |
| *PC2* | *Método `removeCompras(String vendedor)`. Se eliminan los objetos `TicketCompra` apropiados y se contabiliza correctamente.* | 1 | 1,0 | RA6.c | RA6.d | | | | |
| *PC3* | *Método `removeCompras(localDate fecha)`. Recorrido de la lista de objetos `TicketCompra` mediante iteradores.* | 4 | 4,0 | RA6.c | RA6.d | | | | |
| *PC4* | *Método `removeCompras(localDate fecha)`. Se eliminan los objetos `TicketCompra` apropiados y se contabiliza correctamente.* | 1 | 1,0 | RA6.c | RA6.d | | | | |
| *PC5* | *Método `removeCompras(String vendedor, localDate fecha)`. Recorrido de la lista de objetos `TicketCompra` mediante iteradores.* | 4 | 4,0 | RA6.c | RA6.d | | | | |
| *PC6* | *Método `removeCompras(String vendedor, localDate fecha)`. Se eliminan los objetos `TicketCompra` apropiados y se contabiliza correctamente.* | 1 | 1,0 | RA6.c | RA6.d | | | | |
| *PC7* | *No se incluyen elementos extraños, innecesarios o sin sentido.* | 3 | 3,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |
| *PC8* | *No faltan elementos imprescindibles o solicitados en el enunciado.* | 3 | 3,0 | RA6.b | RA6.c | RA6.d | RA6.e | RA6.f | |

**Retroalimentación:**

Correcto.

| **Ejercicio 7. Ordenación de tickets (métodos sort + comparadores)** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Puntos de control (elementos evaluables)** | | **Punt. Máx.** | **Punt. Obt.** | **CE** | **CE** | **CE** | **CE** | **CE** | **CE** |
| *PC1* | *Métodos `sortListaTicketsPorInstante`, `sortListaTicketsPorNumArticulos`, `sortListaTicketsPorVendedor`, `sortListaTicketsPorImporte`.* | 2 | 2,0 | RA6.b | RA6.c | | | | |
| *PC2* | *Clase `ComparadorTicketsPorInstante`.* | 2 | 2,0 | RA6.f | | | | | |
| *PC3* | *Clase `ComparadorTicketsPorNumArticulos`.* | 2 | 2,0 | RA6.f | | | | | |
| *PC4* | *Clase `ComparadorTicketsPorVendedor`.* | 2 | 2,0 | RA6.f | | | | | |
| *PC5* | *Clase `ComparadorTicketsPorImporte`.* | 2 | 2,0 | RA6.f | | | | | |
| *PC6* | *No se incluyen elementos extraños, innecesarios o sin sentido.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |
| *PC7* | *No faltan elementos imprescindibles o solicitados en el enunciado.* | 2 | 2,0 | RA6.b | RA6.c | RA6.d | RA6.e | | |

**Retroalimentación:**

Correcto.