PRÁCTICA INCREMENTAL 2

Objetivos:

- Diseño y manipulación de objetos.
- Correcto empleo de herencia, encapsulación y polimorfismo.
- Diseño y manipulación de matrices polimórficas de objetos (no usar estructuras de datos como la clase Vector o ArrayList)
- Detección y modelado de relaciones entre clases.
- Diseño y manipulación de Excepciones e Interfaces.
- Correcto uso de modificadores de visibilidad.
- Correcto uso de métodos sobrecargados.
- Aplicación de todos los conocimientos sobre programación aprendidos hasta el momento.

Enunciado

Desarrolle un programa que simule parte del funcionamiento de un sistema de gestión de compra y venta de productos de una tienda. El funcionamiento normal de este tipo de sistemas se detalla a continuación. Supongamos que llega un empleado delante del ordenador y lo primero que hace es autenticarse (login y password) en el sistema. Una vez autenticado, el empleado puede realizar las siguientes tareas: modificación de los productos en stock, la realización de pedidos según las peticiones de un cliente o la modificación de su contraseña.

Así pues, tenemos un sistema principal que es el encargado de interactuar con el empleado a través de ciertos menús y comunicarle con los subsistemas correspondientes dependiendo de la operación que desee realizar. El sistema principal consta de dos subsistemas: el sistema de gestión de empleados y el sistema de gestión de pedidos.

El sistema de gestión de empleados se encargará de cargar desde sendos archivos de texto plano todos los empleados de la empresa, los productos y las ofertas, cuando comience a funcionar el programa. Si la ubicación de los ficheros no es correcta se le permitirá dos veces cambiarla y si sigue sin ser correcta se terminará.

La función principal de dicho subsistema es la comprobación de que el login y password de cada empleado que entra en el sistema son correctos.

En el caso de que el usuario introduzca el login o el password de forma errónea se lanzará una excepción que avisará al sistema de que se ha producido dicho error. El sistema tendrá que capturar dicho error y gestionarlo de tal manera que sean solicitados los datos de nuevo al usuario. Si se comete el error 3 veces se acabará l ejecución del programa.

Las excepciones en nuestro sistema tienen dos propiedades un código de error (valor entero) y mensaje descriptivo del error. Cuando se haya introducido mal el login la excepción tendrá como código 111 y mensaje de error “Error. Login incorrecto”, mientras que cuando el fallo sea en el password, el código será 222 y el mensaje será “Error. Password incorrecto”.

Introduzca codigo de empleado

1234
Introduzca password de empleado
55
Excepcion # 222: Error. Password incorrecto
Por favor introduzca un password correcto
Introduzca codigo de empleado
1236
Introduzca password de empleado
pozno
Excepcion # 111: Error. Login incorrecto

Cada empleado se describe con las siguientes características: código de acceso (valor entero único), nombre del usuario (cadena de texto), password (cadena de texto), nivel (valor entero) y productividad (dato real). La productividad representa un plus económico que gana el empleado cada vez que un cliente completa un pedido de forma exitosa. Inicialmente cada empleado tiene una productividad de 0€.

El cálculo de la productividad depende del turno que lleven. Cada vez que se completa un pedido, cada empleado recibe una gratificación: el empleado de nivel 1 recibe 1€ más, el de nivel 2 recibe 2€ y el de nivel 3, 3€.

Los empleados pueden trabajar de noche o de día. Los empleados nocturnos tienen un plus de nocturnidad que es un porcentaje del precio de cada pedido realizado. Es decir, si un empleado de nivel 2 con plus de nocturnidad de 4.5% atiende un pedido de 100€, a su productividad se le sumará 2€ + 100€*0.045.

Mientras, los empleados diurnos, al tener un turno mejor de ventas, reciben la misma gratificación que los nocturnos (1€, 2€ o 3€ según el nivel), pero no tienen plus de nocturnidad. Además, tienen asociada un porcentaje de retención por cada gratificación recibida que depende del nivel. A los empleados de nivel 1, al tener una gratificación muy baja, no se les hará retención. Mientras que un empleado de nivel 3 con retención del 4% por ejemplo, que atienda un pedido de x€ se le aplicará a su productividad una retención de 3€ - 3*0.04€, es decir, la gratificación menos un 4% de esa gratificación (los x€ que costó el pedido no son relevantes). A los empleados de nivel 2 solo se les aplicará una retención similar a los del nivel 3 cuando el pedido sea superior a 300€, si fuera inferior no se les aplicará ninguna retención. Todos los valores antes comentados sobre los máximos de las compras (300€) y las gratificaciones (1€, 2€, 3€) son valores que dependen mucho de la situación financiera de la empresa, por lo que, es conveniente tenerlo fácilmente localizables en una interfaz para poder modificarlos cuando sea necesario.

En resumen, un ejemplo de las propiedades de cada de tipo de empleado, nocturno y diurno, que serán accesibles desde el archivo *empleados.txt* sería:

Empleado Nocturno	Empleado Diurno

nombre: arturo codigo: 1234 password: art1+ nivel: 2 turno: nocturno retencion: 4	nombre: manuel codigo: 1235 password: man2- nivel: 1 turno: diurno plusproductividad: 10.5

Mientras tanto, el subsistema de ventas se encarga de todas las acciones propias de la venta de productos. Este subsistema es el encargado de la gestión de todos los productos disponibles en la tienda, los cuales se leen desde un archivo de texto plano llamado *productos.txt* que se carga cuando comienza a funcionar el programa. Además se encarga también de la carga de todos los tipos de ofertas que realiza el establecimiento.

Cada producto se caracteriza por un código (valor entero), un nombre (cadena de texto), un precio (valor real) y un número de unidades (valor entero) disponibles de cada tipo de producto en la tienda.

Hay dos tipos de productos: perecederos y no perecederos. Los perecederos llevan asociados los días que les quedan para caducar. Con el fin de que se vendan antes de que caduquen su precio normal se verá reducido a un cuarto si queda un día para caducar, a un tercio si quedan dos días y a la mitad si quedan 3. Así un producto que vale 10€, si quedara 1 día, su precio sería 2.5€, si quedaran 2 días, su precio sería 3.33€ y si quedaran 3 días, su precio sería 5€.

Por el contrario, los no perecederos pueden llevar como máximo una oferta asociada, aunque pueden no tener ninguna. En caso de que no lleve asociada ninguna oferta un producto, el archivo *productos.txt* tendrá un identificador de oferta asociado 000, es decir, el campo *oferta* del archivo *productos.txt* valdrá 000.

Todas las ofertas se caracterizan por un identificador (valor entero). Tenemos 3 tipos de ofertas: 2x1, 3x2 y porcentaje de descuento. Cada oferta se encarga de calcular el precio final de un producto en función de su precio y el número de unidades que se hayan comprado de dicho producto.

Así, un producto en oferta 2x1, si se compran 6 unidades sólo se pagarán 3, mientras que si se compran 7 sólo se pagarán 4. Un producto en oferta 3x2, si se compran 3 unidades, sólo se pagan 2, si se compran 4 se pagan 3, si se compran 5 se pagan 4 y si se compran 6 se pagan también 4. Es decir, un producto en oferta 3x2 es sólo realmente rentable si se compran productos en números múltiplos de 3 (3, 6, 9, etc.).

La oferta de porcentaje tiene asociado un porcentaje de descuento sobre cada producto y un número máximo de productos a descontar. Así si la oferta tiene un 7% de descuento para un máximo de 3 unidades, si el usuario compra 3 unidades de un producto a 6€ la unidad, 3*6*(100-7)/100 €. Mientras que si compra 5 unidades de ese mismo producto el precio final sería

3*6*(100-7)/100 + 2*6 € porque sólo 3 productos como máximo pueden tener descuento, los otros dos se pagan sin descuento.

Los tipos de ofertas están localizados en archivo llamado *ofertas.txt.* Un ejemplo de la estructura de archivo con 3 ofertas, una de cada tipo, sería:

```
ofertas:
3
idOferta:
11
tipo:
2x1
idOferta:
12
tipo:
porcentaje
tantoPorciento:
4
maximo:
6
idOferta:
13
tipo:
3x2
```

El subsistema de ventas se encarga de todas las tareas relativas a los productos. Entre esas tareas está listar todos los productos que hay disponibles en el sistema, junto con el número de unidades disponibles en la tienda, actualizar los datos de los productos comprobando que no se repitan datos y consultar los productos existentes.

El funcionamiento detallado será el siguiente. En primer lugar, cuando el sistema arranque solicitará al empleado su código de acceso y su password, si ambos son correctos entonces el proceso continuará, en caso contrario, volverá a solicitar los datos hasta que estos sean correctos. Una vez autenticado el empleado se mostrarán las acciones principales del sistema general:

```
1. Hacer pedido
2. Modificar producto
3. Cambiar contraseña empleado
4. Log out
```

Si el usuario selecciona la opción 4 (log out) el sistema desconectará la sesión del empleado y volverá a la primera acción del sistema, volver a solicitar el código y password de un nuevo empleado. Obviamente esta acción simula la marcha de empleado y la espera de que otro nuevo llegue para registrarse en el sistema. Tras solicitar esta acción, el sistema mostrará los datos del empleado desconectado con el fin de poder ver la productividad del empleado hasta ese momento.

Si el usuario elige la opción 3 (cambiar contraseña) el sistema solicitará al usuario una nueva contraseña y modificará la ya existente a través del subsistema de gestión de empleados.

Si el usuario selecciona la opción 2 (modificar un producto) primero se listarán todos los productos registrados en el sistema y se solicitará al empleado que indique cuál desea modificar. Una vez hecho esto, aparecerá el siguiente menú:

```
2.1. Modificar nombre
2.2. Modificar precio
2.3. Modificar código
2.4. Modificar número de unidades
```

El usuario seleccionará qué atributo desea modificar y el sistema le solicitará el nuevo dato y lo actualizará. En el caso del nombre y el código del producto es necesario comprobar que el nuevo nombre o código no coincidan con el nombre o código de otros productos ya existentes, dado que estos identificadores son únicos. En el caso del número de unidades será necesario comprobar que el número introducido no sea menor de cero. En caso de error se mostrará un mensaje al empleado y volverá a solicitarse un valor distinto hasta que el usuario escriba un dato que no esté repetido. Esta función simularía la entrada de nuevas reservas en el almacén de la tienda gestionada, es decir, el número de unidades del producto serán las que ya existían en la tienda más las que indique el usuario por teclado.

Por último, si el empleado decide hacer un pedido (opción 1) lo primero que se le solicitará será indicar el número de tipos de productos que desea comprar. Una vez leído este dato se comprobará que el número de tipos de productos no es mayor que el número de tipos disponibles en el archivo *productos.txt*. En caso de ser mayor se pedirá de nuevo el dato.

A continuación aparecerá el siguiente menú:

```
1.1 Añadir un tipo de producto
1.2 Visualizar precio total
1.3 Imprimir factura
1.4 Terminar pedido
```

Para añadir un nuevo tipo producto (opción 1.1) se listarán todos los productos disponibles en el sistema y se le solicitará al empleado que indique qué producto desea añadir al pedido y cuántas unidades desea añadir de ese tipo de producto. En caso de que ese producto haya sido añadido anteriormente se mostrará un mensaje de error y se volverá otra vez al mismo menú, en caso contrario se añadirá el producto. Si el usuario ya añadió todos los tipos de productos que indicó en la primera acción antes de este menú entonces el sistema le mostrará un mensaje de error advirtiéndole de que no puedo añadir más productos.

En caso de que el número de unidades que desea el cliente sea mayor que el número de unidades que hay en la tienda, se lanzará otra excepción que llevará por código 333 y como mensaje asociado “Error. Número insuficiente de productos disponibles”. En este caso, la operación se abortará y volverá a mostrarse otra vez el menú anterior. En caso de que el número de unidades sea menor o igual que el número de unidades que hay en la tienda, se descontarán del total.

En este punto surge un problema, tenemos productos del mismo tipo pero con características distintas en dos partes del sistema. Con un ejemplo se verá claramente. El sistema de ventas tiene 6 unidades del producto gel en el almacén de la tienda, y el cliente pide 2 unidades del producto gel, así pues, el pedido tiene el producto gel pero con 2 unidades. Obviamente esto en

memoria son dos productos gel con las mismas características menos una, las unidades. En el almacén quedan 4 unidades y en el pedido hay 2. La forma de solucionar este problema es muy sencilla, se debe duplicar el objeto de la tienda, porque ese ya existe, y luego modificar el número de unidades de cada uno de ellos.

Duplicar un objeto en Java NO es sencillo, aunque en este caso si lo es. En el ANEXO 1 de esta documentación se puede ver un simple ejemplo.

La opción 1.2 muestra por pantalla el precio del pedido en ese momento calculado con sus correspondientes descuentos.

La opción 1.3 visualiza la factura total, indicando los datos de cada producto adquirido, si lleva alguna oferta asociada, el precio final de la factura y la persona que le atendió. Un sistema real debería imprimir la factura mediante una impresora y guardarla en algún archivo o base de datos, pero en este caso se limitará a mostrarla por pantalla para simplificar el problema.

La opción 1.4 terminará este menú aunque no se haya terminado de realizar el pedido y se volverá otra vez el menú principal. En caso de que el pedido no se haya completado la productividad del empleado no será modificada.

Nota 1: El sistema solo deja de funcionar cuando se cometen tres veces errores al introducir el código de empleado o su password.

Nota 2: Al terminar de ejecutar el programa es necesario actualizar los ficheros productos.txt, ofertas y empleados.txt si se cambian algunos de los datos como el password de los empleados o las unidades de cada producto.

Condiciones de la entrega:

La fecha de entrega de la prácticas será la semana del 2 al 8 de mayo, cada grupo de prácticas en su horario de prácticas habitual. Se subirá el archivo antes del 2 de mayo a las 10:00.

Se recuerda que los alumnos deberán defender en clase de prácticas el trabajo realizado, siendo necesario que asistan ambos miembros de cada pareja de prácticas, en caso de que uno de ellos no se presente, ese alumno no será evaluado y por tanto suspenso en la convocatoria ordinaria. Igualmente es necesario recordar que es necesario que la documentación entregada contenga análisis, diseño, manual de usuario y código fuente. En la fase de diseño deberá entregarse tanto el diagrama de clases como el pseudocódigo correspondiente a aquellos métodos que se consideren los suficientemente relevantes y complejos como para tener que detallarlos mediante pseudocódigo.

Toda esta información se subirá a través de Moodle en un espacio habilitado para tal efecto, antes de la hora de la defensa para que quede constancia de la entrega. Posteriormente a esta fecha no se permitirá subir ningún otro documento. Toda la documentación deberá ir comprimida en un único archivo con extensión .rar o .zip. Preferiblemente la documentación debería ir en formato PDF y el código fuente en su correspondiente archivo .java. Los archivo binarios .class no son necesarios ya que deberán generarse de forma fácil siguiendo las instrucciones redactadas en el manual de usuario.

El funcionamiento de la práctica no garantiza el aprobado de la misma, se valorarán otros aspectos como la robustez y eficiencia de la misma, el correcto diseño y perfecta utilización de las técnicas y metodologías comentadas en clase de teoría, la documentación entregada y el cumplimiento de los requisitos definidos en el enunciado. Además, la práctica deberá incluir todas las modificaciones que fueran indicadas por el profesor cuando fue corregido el hito 1.

Anexo 1

Interfaz Clonable

Un interface declara un conjunto de funciones, pero sin implementarlas. El interface Cloneable es muy simple ya que no define ninguna función.

```
public interface Cloneable {
\\ Está vacía
}
```

Una clase que implemente este interface le indica al método clone de la clase Object que puede hacer una copia miembro a miembro de las instancias de dicha clase. Si una clase no implementa esta interface, e intenta hacer una duplicación del objeto a través de la llamada al método clone de la clase base Object, da como resultado una excepción del tipo CloneNotSupportedException.

Un ejemplo sería:

```
public class Punto implements Cloneable{
   private int x;
   private int y;

//constructores ...

   public Object clone(){
      Object obj=null;
      try{
         obj=super.clone();
      }catch(CloneNotSupportedException ex){
         System.out.println(" no se puede duplicar");
      }
      return obj;
   }
// otros métodos……
}


public class EjemploDeUso {

   public static void main(){
      Punto punto1 = new Punto(3,4);
      Punto punto2 = punto1.clone();
            // y ya tenemos dos objetos punto exactamente iguales pero cada uno tiene su
            //espacio de memoria propio
   }
}
```