EJERCICIOS DE APRENDIZAJE EXTRA

1. Diseñar un programa que lea una serie de valores numéricos enteros desde el teclado

y los guarde en un ArrayList de tipo Integer. La lectura de números termina cuando se

introduzca el valor -99. Este valor no se guarda en el ArrayList. A continuación, el

programa mostrará por pantalla el número de valores que se han leído, su suma y su

media (promedio).

2. Crear una clase llamada CantanteFamoso. Esta clase guardará cantantes famosos y

tendrá como atributos el nombre y discoConMasVentas.

Se debe, además, en el main, crear una lista de tipo CantanteFamoso y agregar 5

objetos de tipo CantanteFamoso a la lista. Luego, se debe mostrar los nombres de

cada cantante y su disco con más ventas por pantalla.

Una vez agregado los 5, en otro bucle, crear un menú que le de la opción al usuario de

agregar un cantante más, mostrar todos los cantantes, eliminar un cantante que el

usuario elija o de salir del programa. Al final se deberá mostrar la lista con todos los

cambios.

3. Implemente un programa para una Librería haciendo uso de un HashSet para evitar

datos repetidos. Para ello, se debe crear una clase llamada Libro que guarde la

información de cada uno de los libros de una Biblioteca. La clase Libro debe guardar

el título del libro, autor, número de ejemplares y número de ejemplares prestados.

También se creará en el main un HashSet de tipo Libro que guardará todos los libros

creados.

En el main tendremos un bucle que crea un objeto Libro pidiéndole al usuario todos

sus datos y los seteandolos en el Libro. Despues, ese Libro se guarda en el conjunto y

se le pregunta al usuario si quiere crear otro Libro o no.

La clase Librería contendrá además los siguientes métodos:

• Constructor por defecto.

• Constructor con parámetros.

• Métodos Setters/getters

• Método prestamo(): El usuario ingresa el titulo del libro que quiere prestar y se lo

busca en el conjunto. Si está en el conjunto, se llama con ese objeto Libro al

método. El método se incrementa de a uno, como un carrito de compras, el

atributo ejemplares prestados, del libro que ingresó el usuario. Esto sucederá

cada vez que se realice un préstamo del libro. No se podrán prestar libros de los

que no queden ejemplares disponibles para prestar. Devuelve true si se ha

podido realizar la operación y false en caso contrario.

• Método devolucion(): El usuario ingresa el titulo del libro que quiere devolver y se

lo busca en el conjunto. Si está en el conjunto, se llama con ese objeto al método.

El método decrementa de a uno, como un carrito de compras, el atributo

ejemplares prestados, del libro seleccionado por el usuario. Esto sucederá cada

vez que se produzca la devolución de un libro. No se podrán devolver libros

donde que no tengan ejemplares prestados. Devuelve true si se ha podido

realizar la operación y false en caso contrario.

• Método toString para mostrar los datos de los libros.

4. Almacena en un HashMap los códigos postales de 10 ciudades a elección de esta

página: https://mapanet.eu/index.htm. Nota: Poner el código postal sin la letra, solo el

numero.

• Pedirle al usuario que ingrese 10 códigos postales y sus ciudades.

• Muestra por pantalla los datos introducidos

• Pide un código postal y muestra la ciudad asociada si existe sino avisa al

usuario.

• Muestra por pantalla los datos

• Agregar una ciudad con su código postal correspondiente más al HashMap.

• Elimina 3 ciudades existentes dentro del HashMap.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ejercicios Extras para Mentores

1) Crear una clase llamada "AgendaTelefónica" que permita almacenar contactos con su nombre y número de teléfono.

La clase debe tener métodos para agregar un contacto nuevo, buscar un contacto por nombre, mostrar todos los contactos y mostrar los contactos ordenados alfabéticamente por nombre.

En el programa principal, crear un mapa (HashMap) de tipo "String" (nombre del contacto) y "Integer" (número de teléfono). Luego, mostrar un menú que le dé al usuario la opción de agregar un contacto, buscar un contacto por nombre, mostrar todos los contactos o mostrar los contactos ordenados alfabéticamente.

2) En este ejercicio vamos a mostrar la tabla de posiciones final de un torneo del deporte que deseen. Para ello deberán crear una clase llamada "EquipoDe…" que represente a un equipo del deporte seleccionado. La clase debe tener los siguientes atributos: nombre del equipo, ciudad, cantidad de partidos jugados, ganados, perdidos y empatados, y puntos totales.

En el programa principal, crear una lista de tipo "EquipoDe…" y agregar 10 objetos de tipo equipo a la lista. Luego, mostrar por pantalla de forma ordenada según cantidad de puntos los nombres de cada equipo, su cantidad de puntos, partidos jugados, ganados, perdidos y empatados.

Una vez finalizado el torneo, determinar el equipo ganador, eliminar al último equipo de la tabla por haber descendido, sumar al nuevo equipo que ascendió de la división inferior, reiniciar los valores de partidos y puntos a 0 para el inicio del siguiente campeonato y mostrar la lista en orden alfabético de los equipos.

3)Desarrolla una función que reciba como parámetro un ArrayList de objetos "Persona" y devuelva un HashSet con los nombres de las personas. A continuación, implementa otra función que tome este HashSet como entrada y retorne un TreeSet ordenado de los nombres. Además, crea las entidades y servicios necesarios.

4) Eres el responsable de una tienda de libros y necesitas gestionar el inventario de tu tienda. Debes desarrollar un programa en Java que te permita realizar las siguientes operaciones:

A. Crear una clase Libro con los siguientes atributos: titulo (String), autor (String) y precio (double).

B. Implementar la clase InventarioLibreria que contenga las siguientes funcionalidades:

· Un ArrayList para almacenar los libros en el inventario.

· Un HashSet para almacenar los autores únicos de los libros en el inventario.

· Un TreeMap para almacenar los libros ordenados por título.

· Un método agregarLibro que reciba como parámetros el título, autor y precio de un libro y lo agregue al inventario.

· Un método obtenerAutoresUnicos que retorne el conjunto de autores únicos presentes en el inventario.

· Un método obtenerLibrosOrdenadosPorTitulo que retorne una lista de libros ordenados alfabéticamente por título.

· Un método calcularPrecioPromedio que calcule y retorne el precio promedio de todos los libros en el inventario.

· Un método imprimirInventario que imprima por pantalla todos los libros en el inventario, mostrando título, autor y precio.

C. En el método principal (main), crea una instancia de InventarioLibreria y realiza las siguientes operaciones:

· Agrega al menos 5 libros al inventario utilizando el método agregarLibro.

· Muestra por pantalla los autores únicos presentes en el inventario utilizando el método obtenerAutoresUnicos.

· Muestra por pantalla los libros ordenados por título utilizando el método obtenerLibrosOrdenadosPorTitulo.

· Muestra por pantalla el precio promedio de los libros en el inventario utilizando el método calcularPrecioPromedio.

· Utiliza el método imprimirInventario para mostrar todos los libros en el inventario.

5) Crear la clase “Tarea” con los siguientes atributos: título, descripción, estado (boolean) y fecha de creación. Agregar constructor vacío, con parámetros, getters, setters y demás necesario.

Dentro del servicio vamos a crear un objeto de tipo Tarea pidiéndole los datos al usuario.Por defecto, al crear una tarea se tiene que asignar el valor de true al atributo estado, esto indicará que la tarea está en proceso.

Posteriormente, guardamos el objeto dentro de un conjunto de Tareas, para evitar duplicados.

Luego debemos:

-Mostrar todas las tareas

- Buscar una tarea por su nombre. El usuario debe ingresar el nombre de la tarea a buscar y se debe mostrar el objeto completo.

- Modificar el estado de la tarea a false, de esta forma se considera terminada. El usuario debe ingresar el nombre de la tarea a modificar.

- Remover una tarea. El usuario debe ingresar el nombre de la tarea a remover.

6) Crear la clase "Pedido" con los siguientes atributos: número de orden, descripción y fecha de creación. Agregar constructor vacío, con parámetros, getters, setters y demás necesario.

Dentro del servicio vamos a crear un método para la creación del objeto Pedido, solicitando los datos al usuario. Dichos objetos creados, los vamos a guardar dentro de una lista de Pedidos.

Luego debemos:

- Buscar un pedido según el número de orden y mostrarlo.

- Buscar un pedido según una fecha ingresada por el usuario y mostrarlo. En el caso de que no existan pedidos con esa fecha, debe informar al usuario.

- Mostrar todos los pedidos

- Ordenar los pedidos por fecha de creación y mostrarlos

- Remover los 3 primeros pedidos de la lista

Utilizar un switch para que el usuario elija la opción que desea:

1)Agregar Pedido

2)Buscar un Pedido por número de orden

3)Buscar un Pedido por fecha de creación

4)Mostrar todos los pedidos

5)Ordenar y mostrar los pedidos según fecha de creación

6)Remover un Pedido de la lista

7) Sistema de gestión de empleados

Crea un sistema de gestión de empleados en Java. Este sistema debe ser capaz de agregar, eliminar, buscar y actualizar empleados. Cada empleado debe tener un nombre, una edad, un número de identificación único y una fecha de contratación.

Crea las siguientes clases de entidad:

Empleado: que debe contener los atributos mencionados anteriormente.

Crea las siguientes clases de servicios:

ServicioEmpleado: que debe contener los métodos para agregar, eliminar, buscar y actualizar empleados.

Utiliza la colección ArrayList para almacenar los empleados.

8) Crea un juego de cartas en Java. Este juego debe ser capaz de repartir cartas a varios jugadores y determinar el ganador de la partida. Cada carta debe tener un valor y un palo.

Crea la siguiente clase de entidad:

Carta: que debe contener los atributos mencionados anteriormente.

Crea las siguientes clases de servicios:

ServicioJuego: que debe contener los métodos para repartir cartas a los jugadores y determinar el ganador de la partida.

Utiliza la colección ArrayList para almacenar las cartas y los jugadores. Utiliza la clase HashMap para asignar un valor numérico a cada carta.

El ServicioJuego debe tener los siguientes métodos:

crearMazo(): Crea un mazo de cartas, con todas las combinaciones de valores y palos.

barajar(mazo): Baraja el mazo de cartas.

repartirCartas(mazo, numJugadores, numCartas): Reparte un número determinado de cartas a un número determinado de jugadores.

determinarGanador(jugadores): Determina el ganador de la partida, según las cartas que tiene cada jugador.

Puedes mejorar el juego agregando más funcionalidades, como la posibilidad de apostar fichas o la opción de jugar varias rondas.