Argentina programa

Programación Orientada a Objetos con JAVA

Guía V –Parte 2

Ejercicios Extras

EJERCICIOS DE APRENDIZAJE EXTRA

Estos van a ser ejercicios para reforzar los conocimientos previamente vistos. Estos pueden realizarse cuando hayas terminado la guía y tengas una buena base sobre lo que venimos trabajando. Además, si ya terminaste la guía y te queda tiempo libre en las mesas, podes continuar con estos ejercicios extra, recordando siempre que no es necesario que los termines para continuar con el tema siguiente. Por último, recordá que la prioridad es ayudar a los compañeros de la mesa y que cuando tengas que ayudar, lo más valioso es que puedas explicar el ejercicio con la intención de que tu compañero lo comprenda, y no sólo mostrarlo. ¡Muchas gracias!

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 dé 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 título 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 título 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 número.

• 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, que pida el usuario. • Muestra por pantalla los datos.