Hoja 15

1. Elige 10 series de televisión que hayan comenzado a emitirse en años diferentes. Crea una HashMap donde almacenes el título de cada serie (su clave será el año de emisión). Recorre el HashMap usando un iterador y un bucle for-each e imprime por pantalla el título y el año de emisión de cada serie en ambos casos.

2. Diseñar una clase llamada Canción con los atributos privados: título y autor; con un constructor y métodos que permitan acceder y modificar los atributos.

En otra clase Reproducción crea una lista de elementos de tipo canción, y como métodos:

Un método llamado reproduceCancion que muestra el título y autor de la canción de la posición indicada.

Un método llamado avance que muestra la siguiente canción.

Un método llamado retroceso que muestra la anterior canción.

Un método llamado inicio que muestra la primera canción.

Un método llamado fin que muestra la última canción.

Un método llamado aleatorio que genera los datos de una canción aleatoriamente.

Añadir otros métodos que parezcan útiles (reproduceCDenOrden, reproduceCDenAleatorio, …)

En una clase principal crea dos CD (dos reproducciones) con tus 10 canciones favoritas. Cada CD debe ser un HashMap y la clave de cada canción será en nº de la canción.

Ofrece al usuario reproducir uno de los CD, y un menú con todas las opciones para dicha reproducción, (incluir ON y OFF).

3. Escribe un programa que pida 5 números por teclado, los almacene en un array y que luego muestre el máximo valor, el mínimo y las posiciones que ocupan en el array.

4. Escribe un programa que genere 10 números enteros aleatorios entre 0 y 999 y los almacene en un array. El programa debe crear un nuevo array con los números primos que haya entre esos 10 números. Luego debe mostrar por pantalla los dos arrays.

5. Muestra por pantalla un array de 4 filas y 3 columnas con 12 números aleatorios entre 0 y 99. Muestra también la suma total de los números de cada columna.

6. Proyecto Taller:

Se ha de diseñar una aplicación que permita crear un taller y a través de él gestionar los coches que en él se irán almacenando. Se distribuirá el código en tres clases:

Clase Coche:

Atributos: color y marca

Métodos:

Se espera encontrar en la clase los métodos Constructores necesarios, los métodos set y get, y aquellos que se consideren oportunos para el trabajo con objetos de tipo Coche. No hace falta validar los atributos.

Se sugiere sobreescibir el método toString, para que muestre los atributos del objeto.

Clase Taller:

Atributo privado:

HashMap que permita almacenar dinámicamente parejas formadas por una matrícula junto a un objeto de tipo Coche.

Métodos:

anadeElemento: Incluye la petición de datos (color, marca y matrícula), la creación del objeto de tipo Coche, valida la matrícula, comprueba que no hay ningún contacto ya en la colección con la misma matrícula y añade la pareja matricula-Coche al mapa.

eliminaElemento: pide matrícula del coche a borrar y lo elimina, informando de la no existencia del mismo dado el caso.

visualizaMatriculas: Recorre el mapa mostrando el conjunto de matrículas almacenadas.(Basado en keySet())

visualizaCoches: Recorre el mapa mostrando color, marca de cada coche almacenado. (Basado en values())

visualizaTaller: Recorre el mapa mostrando color, marca y matricula de cada coche almacenado. (Basado en entrySet().toString)

En esta clase se podrán incluir aquellos métodos que se consideren oportunos y estén relacionados con la gestión de elementos del mapa y del mapa en sí.

Clase Principal

En la clase Principal se creará un objeto Taller y se interactuará con un usuario al que se le presentará el siguiente menú:

1. Añadir coche

2. Eliminar coche

3. Salir

Al salir se llamará a los métodos visualizaMatriculas, visualizaCoches, visualizaTaller.

El menú se resolverá utilizando los métodos creados. El menú será repetitivo en todos los casos incluidos los casos de opción no válida o dato de formato incorrecto, hasta que el usuario seleccione la opción 3.

7. Utiliza la clase "Coche" creada en ejercicios anteriores. Cópiala en un nuevo paquete de trabajo. Modifica la clase cambiando el atributo velocidad por la cadena "color".
Crea una clase "Almacén" que contenga un ArrayList que almacene objetos de tipo Coche.
La clase Almacén deberá tener además métodos que permitan: añadir un nuevo coche al almacén (método sobrecargado que admite como argumento un objeto, o bien una posición y un objeto que insertar en esa posición), eliminar un coche del almacén (método sobrecargado que admite como argumento un objeto, o bien una posición del objeto que eliminar) y visualizar todos los coches que hay en el almacén en ese momento con sus datos.
La clase Almacén deberá poseer un constructor copia que permita realizar una copia de seguridad de todo el almacén existente.
Crea un método "buscar"

- Crea un método "buscar" donde pases como parámetro la matrícula de un coche y te devuelva la información asociada al mismo (marca y color).
- Asegúrate además de que si la matrícula del coche no se encuentra en el Almacén te devuelva un mensaje donde así te lo indique.
- Crea un método "buscar" donde pases como argumento un objeto y localice si ese objeto está o no está en la colección.

Diseña una clase Principal (main) que pruebe todas estas funciones de almacén:

- Una vez creado el almacén deberás rellenarlo con al menos cuatro coches.
- Comprueba el tamaño del almacén.
- Elimina dos de los coches. Visualiza la colección de nuevo.
- Añádelos en la posición final.
- Pregunta por teclado a un usuario qué vehículo desea localizar (pidiendo su matrícula) y da respuesta a tal búsqueda con la información de ese objeto.
- Utiliza el constructor copia para comprobar que realizas correctamente la copia de seguridad del Almacén.
- Pregunta por teclado a un usuario qué vehículo desea localizar (necesitas todos sus datos) y da respuesta a tal búsqueda, diciendo si el objeto existe o no en el almacén.