Relación ejercicios
Colecciones II

Ejercicio 1 (colecciones I)

Crea un **ArrayList** con los nombres de 6 compañeros de clase. A continuación, muestra esos nombres por pantalla. Utiliza para ello un bucle for que recorra todo el **ArrayList** sin usar ningún índice.

Ejercicio 2 (colecciones I)

Realiza un programa que introduzca valores aleatorios (entre 0 y 100) en un **ArrayList** y que luego calcule la suma, la media, el máximo y el mínimo de esos números. El tamaño de la lista también será aleatorio y podrá oscilar entre 10 y 20 elementos ambos inclusive.

Ejercicio 3 (colecciones I)

Escribe un programa que ordene 10 números enteros introducidos por teclado y almacenados en un objeto de la clase **ArrayList**.

Ejercicio 4 (colecciones I)

Realiza un programa equivalente al anterior, pero en esta ocasión, el programa debe ordenar palabras en lugar de números.

Ejercicio 5

Realiza de nuevo el ejercicio de la colección de Discos pero utilizando esta vez una lista para almacenar la información sobre los discos en lugar de un array convencional. Comprobarás que el código se simplifica notablemente ¿Cuánto ocupa el programa original hecho con un array? ¿Cuánto ocupa este nuevo programa hecho con una lista?

Ejercicio 6

Implementa el control de acceso al área restringida de un programa. Se debe pedir un nombre de usuario y una contraseña. Si el usuario introduce los datos correctamente, el programa dirá “Ha accedido al área restringida”. El usuario tendrá un máximo de 3 oportunidades. Si se agotan las oportunidades el programa dirá “Lo siento, no tiene acceso al área restringida”. Los nombres de usuario con sus correspondientes contraseñas deben estar almacenados en una estructura de la clase **HashMap**

Ejercicio 7

La máquina Eurocoin genera una moneda de curso legal cada vez que se pulsa un botón siguiendo la siguiente pauta: o bien coincide el valor con la moneda anteriormente generada - 1 céntimo, 2 céntimos, 5 céntimos, 10 céntimos, 25 céntimos, 50 céntimos, 1 euro o 2 euros - o bien coincide la posición – cara o cruz. Simula, mediante un programa, la generación de 6 monedas aleatorias siguiendo la pauta correcta. Cada moneda generada debe ser una instancia de la clase Moneda y la secuencia se debe ir almacenando en una lista.

Ejemplo:

2 céntimos – cara

2 céntimos – cruz

50 céntimos – cruz

1 euro – cruz

1 euro – cara

10 céntimos – cara.

Ejercicio 8

Realiza un programa que escoja al azar 10 cartas de la baraja española (10 objetos de la clase Carta). Emplea un objeto de la clase **ArrayList** para almacenarlas y asegúrate de que no se repite ninguna.

Ejercicio 9

Modifica el programa anterior de tal forma que las cartas se muestren ordenadas. Primero se ordenarán por palo: bastos, copas, espadas, oros. Cuando coincida el palo, se ordenará por número: as, 2, 3, 4, 5, 6, 7, sota, caballo, rey.

Ejercicio 10

Crea un mini-diccionario español-inglés que contenga, al menos, 20 palabras (con su correspondiente traducción). Utiliza un objeto de la clase **HashMap** para almacenar las parejas de palabras. El programa pedirá una palabra en español y dará la correspondiente traducción en inglés.

Ejercicio 11

Crea un mini-diccionario español-inglés que contenga, al menos, 20 palabras con su correspondiente traducción). Utiliza un objeto de la clase **HashMap** para almacenar las parejas de palabras. El programa pedirá una palabra en español y dará la correspondiente traducción en inglés.

Ejercicio 12

Escribe un programa que genere una secuencia de 5 cartas de la baraja española y que sume los puntos según el juego de la brisca. El valor de las cartas se debe guardar en una estructura **HashMap** que debe contener parejas (figura, valor), por ejemplo (“caballo”, 3). La secuencia de cartas debe ser una estructura de la clase **ArrayList** que contiene objetos de la clase Carta. El valor de las cartas es el siguiente: as → 11, tres → 10, sota → 2, caballo → 3, rey → 4; el resto de cartas no vale nada.

Ejemplo:

as de oros

cinco de bastos

caballo de espadas

sota de copas.

tres de oros

Tienes 26 puntos

Ejercicio 13

Modifica el programa **Gestisimal** * realizado anteriormente añadiendo las siguientes mejoras:

• Utiliza una lista en lugar de un array para el almacenamiento de los datos.

• Comprueba la existencia del código en el alta, la baja y la modificación de artículos para evitar errores.

• Cambia la opción “Salida de stock” por “Venta”. Esta nueva opción permitirá hacer una venta de varios artículos y emitir la factura correspondiente. Se debe preguntar por los códigos y las cantidades de cada artículo que se quiere comprar. Aplica un 21% de IVA.

Crea el programa **GESTISIMAL** (GESTIón SIMplificada de Almacén) para llevar el control de los artículos de un almacén. De cada artículo se debe saber el código, la descripción, el precio de compra, el precio de venta y el stock (número de unidades). El menú del programa debe tener, al menos, las siguientes opciones:

1. Listado
2. Alta
3. Baja
4. Modificación
5. Entrada de mercancía
6. Salida de mercancía
7. Salir

La entrada y salida de mercancía supone respectivamente el incremento y decremento de stock de un determinado artículo. Hay que controlar que no se pueda sacar más mercancía de la que hay en el almacén