PLAN DE REFUERZO
ACTIVIDADES 2º TRIMESTRE

SEMANA 18-24 de MAYO

EJERCICIO 1

Elaborar una aplicación que reciba el nombre de un fichero de texto y una cadena.

Al ejecutar el programa deberá mostrar:

- **"El fichero no existe"** si el fichero que le hemos pasado no exist**e**
- **La cadena "....." no existe** si no se encuentra la cadena en el fichero que se lee.
- **La cadena "....." aparece N veces** siendo N el número de veces que aparece la cadena en el fichero en cuestión.

EJERCICIO 2

En ocasiones, es necesario particionar un fichero de texto de gran tamaño en otros ficheros más pequeños. Se pide crear una aplicación a la que se le proporciona un fichero de texto, y un tamaño. Este puede ser en Bytes, kiloBytes o MegaBytes. La aplicación debe fragmentar el fichero original en tantos ficheros del tamaño especificado como necesite. Los nombres de estos ficheros serán idénticos con el prefijo "parte999_" donde 999 es el nombre del volumen (la parte en concreto).

Para indicar el tamaño se escribirá una cantidad seguida de b (bytes), k (kbytes) o m (MBytes)

EJERCICIO 3

Escribir una programa en el que partiendo de los ficheros generados en el ejercicio anterior se reestablezca el fichero original.

EJERCICIO 4

Utilizando arrays (no collections) Implementar un programa que permita dar de alta y baja aviones y pasajeros (clases que debéis haber realizado en las tareas del trimestre anterior).

Al iniciarse el programa se cargarán desde el disco los aviones y pasajeros almacenados en los ficheros binarios aviones.dat y pasajeros.dat y se mostrará el menú. Al salir del programa se volverán a guardar de nuevo todo en disco.

El menú deberá tener la siguiente forma:

1. Alta Avión
2. Baja Avión
3. Alta Pasajero
4. Baja Pasajero
5. Mostrar aviones por pantalla
6. Mostrar pasajeros por pantalla

EJERCICIO 5

Hacer lo mismo que en el ejercicio anterior pero con ficheros de texto en los que la información de los aviones y los pasajeros se muestre correctamente de manera formateado y donde se pueda distinguir perfectamente los diferentes aviones y los diferentes pasajeros.

SEMANA 25-31 de MAYO

EJERCICIO 6

Escribe un programa que genere aleatoriamente 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 7

Realiza un buscador de sinónimos.

Utiliza el diccionario español-inglés que se proporciona a continuación. El programa preguntará una palabra y dará una lista de sinónimos (palabras que tienen el mismo significado).

Por ejemplo, si se introduce la palabra “caliente”, el programa dará como resultado: ardiente, candente, abrasador. ¿Cómo sabe el programa cuáles son los sinónimos de “caliente”? Muy fácil, en el diccionario debe existir la entrada (“caliente”,“hot”), por tanto solo tendrá que buscar las palabras en español que también signifiquen “hot”; esta información estará en las entradas (“ardiente”, “hot”) y(“abrasador”, “hot”). Cuando una palabra existe en el diccionario pero no tiene sinónimos, debe mostrar el mensaje “No conozco sinónimos de esa palabra”.

Español	caliente	rojo	ardiente	verde	agujetas	abrasador	hierro	grande
Inglés	hot	red	hot	green	stiff	hot	iron	big

Si una palabra no está en el diccionario se mostrará el mensaje “No conozco esa palabra”. El usuario sale del programa escribiendo la palabra “salir”.

Ejemplo:

Introduzca una palabra y le daré los sinónimos: caliente
Sinónimos de caliente: ardiente, abrasador
Introduzca una palabra y le daré los sinónimos: rojo
No conozco sinónimos de esa palabra
Introduzca una palabra y le daré los sinónimos: blanco
No conozco
Introduzca
No conozco
Introduzca
esa palabra
una palabra y le daré los sinónimos: grande
sinónimos de esa palabra
una palabra y le daré los sinónimos: salir

EJERCICIO 8

Amplía el programa anterior de tal forma que sea capaz de aprender palabras y sinónimos.

Cuando una palabra no tiene sinónimos, es decir, cuando aparece la palabra en español con su traducción y esa traducción no la tiene ninguna otra palabra española, se le preguntará al usuario si quiere añadir uno (un sinónimo) y, en caso afirmativo, se pedirá la palabra y se añadirá al diccionario.

Se puede dar la circunstancia de que el usuario introduzca una palabra en español que no está en el diccionario; en tal caso, se mostrará el consiguiente mensaje y se dará la posibilidad al usuario de añadir la entrada correspondiente en el diccionario pidiendo, claro está, la palabra en inglés.

EJERCICIO 9

Desarrolla un programa para una estación meteorológica.

Tendremos una clase “EstacionMeteorologica” que dispondrá de:

- una lista de objetos de tipo Medicion
- un atributo de tipo Coordenadas que determinará la localización de la estación.

El constructor de la EstacionMeteorologica recibirá un nombre de un fichero binario que contendrá las mediciones, y un objeto de tipo Coordenadas. Deberás por tanto insertar los elementos del fichero en el atributo de tipo lista, y el objeto de tipo Coordenadas determinará su localización. Para probarlo al principio, create un archivo vacío "datos.dat" o con el nombre que quieras.

- un método addMedicion que reciba una medición y la introduzca en la lista.
- un método "ordenaTemperaturasAsc" que ordene la lista por orden ascendente de temperaturas, teniendo en cuenta que consideramos este orden el orden natural de objetos de tipo Medicion.
- un método "ordenaHumedadesDesc" que ordene la lista por orden descendente de humedades.
- un método "presionMaxima" que devuelva el objeto Medicion de mayor presion.
- un método "buscaMedicion" que reciba un objeto de tipo Medicion y devuelva true o false dependiendo de si lo encuentra o no.
- un método guardarFichero que reciba el nombre de un fichero y guarde los datos de la lista en ese fichero binario.

La clase Medicion tendrá estos atributos:

- temperatura, humedad, presión. Consideramos que los 3 son valores enteros.
- un constructor que reciba los valores de todos sus atributos.

La clase Coordenadas tendrá los atributos:

- latitud, longitud. También enteros.
- un constructor que reciba los valores de todos sus atributos.

EJERCICIO 10

Se pide desarrollar una aplicación de gestión de un instituto. Se tendrán estas clases con estas funcionalidades:

Una clase Instituto, que tendrá como atributos:

- Un atributo que no podrá verse desde clases externas llamado "departamentos" que contendrá el conjunto de los departamentos del instituto.
- Una cadena "nombre" que será el nombre del instituto
- Un código de instituto, que será un número entero primitivo

Su constructor recibirá:

- El nombre de un fichero binario que contiene los profesores de ese instituto
- El nombre del instituto
- El codigo del instituto

Se deberá por tanto dar valor a los atributos nombre y código, así como recorrer el fichero de profesores.

En el recorrido del fichero, por cada profesor, comprobaremos si su departamento ya está incluído en el instituto. Si es así, añadiremos el profesor al departamento correspondiente. Si no está incluído ese departamento, lo crearemos, añadiendo este profesor, y lo incluiremos en la lista de departamentos del instituto.

Para desarrollar este constructor debes ayudarte de dos métodos que deberás implementar en esta misma clase, y cuyos prototipos son:

- boolean contieneDepto(Departamento d): devolverá true si hay un departamento igual que otro (debes utilizar el método equals que compara objetos Departamento, que se indica en el apartado de la clase Departamento).
- Departamento getDepartamento(Departamento d)
- boolean anadeDepto(Departamento d)
- Otro método que deberá tener esta clase es el siguiente:

  void guardarProfesoresEnFichero(String nomfichero)

Que deberá guardar cada uno de los profesores del instituto en el fichero cuyo nombre se pase por parámetro.

Una clase Departamento, que tendrá como atributos:

- Una lista de objetos de tipo Profesor.
- Un identificador de departamento, que es un número entero primitivo.
- Su constructor recibirá un entero que se corresponderá con el id del departamento.
- Consideramos dos departamentos iguales siempre que tengan el mismo identificador. Implementar el método equals que lo determine.

Una clase Profesor, que tendrá como atributos:

- El nombre del profesor
- El dni
- La profesión
- Un entero que se corresponde con el id de departamento al que pertenece el profesor

En la clase principal debes crear varios profesores asociados a un id_departamento, crear ese departamento, añadirle esos profesores a su lista, crear un Instituto con los datos que quieras y añadirle el departamento creado, y llamar al método que guarda los profesores de ese instituto.

Además, desde la clase principal, podrás llamar a system.out.println pasándole como parámetro el instituto, y se mostrarán por pantalla todos sus datos, así como los departamentos con todos sus profesores. Implementa los métodos que necesites para conseguir esto.