TICKETS SOPORTE TÉCNICO

Vamos a modelar los tickets creados por una empresa para dar soporte técnico.

PRIMERA PARTE (5 puntos)

Crea un paquete "models" y dentro de él, las siguientes clases:

- Creamos una clase **Persona** con los siguientes atributos: id, nombre, apellidos, email, móvil. Implementa, constructor, getters, setters, toString.
- De esa clase van a heredar dos clases: **Usuario** que añade un campo fechaAlta, y **Tecnico** que añade un campo enumerado de especiadad (INFORMATICA, ALBAÑILERIA, FONTANERÍA, CARPINTERIA, PINTURA), valoración (1-5). Para cada una implementa constructor, getters, setters, toString.
- Creamos una clase **TicketSoporte** que tendrá los siguientes atributos: id (autoincremental), fechaCreacion, fechaFinalizacion (nullable), estado (ABIERTO, ENPROCESO, RESUELTO), prioridad (1-3), ***usuario*** solicitante, ***técnico*** asignado, comentarios. Implementa constructor con id y sin id, getters, setters, toString y equals por id. Implementa *Comparable* por fechaCreacion.

Crea un paquete "services" y, dentro de él, la siguiente clase:

- Clase **ServicioSoporte**. Esta clase tendrá como atributos una **lista** de TicketSoporte, y dos **sets,** uno de usuarios y otro de técnicos. Crea métodos para añadir y eliminar a cada uno de ellos (tickets, usuarios, tecnicos). Al añadir una reserva, el hotel que lleva como parámetro debe estar en la lista de hoteles. Métodos:
    - **addUsuario**(Usuario u), **deleteUsuario**(int id)
    - **addTecnico**(Tecnico u), **deleteTecnico**(int id)
    - **addTicketSoporte**(fechaCreacion, fechaFinalizacion, prioridad, comentarios, ***usuario***, ***tecnico***). Para sacar el id del ticket deberías sacar el máximo de los ids que ya hay y sumarle uno.
    - **deleteTicketSoporte**(int id): elimina un Ticket de la lista correspondiente.
    - Tecnico **findTecnicoById**(int id): técnico con el id indicado.
    - Usuario **findUsuarioById**(int id): usuario con el id indicado.
    - List<Tecnico> **getTecnicosByEspecialidad**(Especialidad esp): lista de técnicos de una especialidad.
    - Map<Especialidad, List<Tecnico>> **getTecnicosGroupByEspecialidad**(Especialidad esp): mapa con especialidad y para cada una la lista de técnicos de esa especialidad ordenados por valoración.
    - TicketSoporte **findTicketById**(int id): ticket de soporte con el id indicado.
    - List<TicketSoporte> **getTicketsAbiertos**(): lista con todos los tickets abiertos ordenados por fecha de creacion descendente.
    - List<TicketSoporte> **getTicketsCerrados**(): lista con todos los tickets resueltos ordenados por fecha de finalización descendente.
    - List<TicketSoporte> **getTicketsEnProcesoTecnicoInformatico**(): devuelve todos los tickets ENPROCESO que tienen asignado un técnico de la especialidad de informática.
    - Integer **getTotalTicketsResueltos**(Integer prioridad): devuelve el total de tickets resueltos de una determinada prioridad.
    - Set<TicketSoporte> **findTicketsByEstadoAndPrioridad**(Estado estado, Integer prioridad): conjunto con los tickets con el estado y la prioridad indicados, ordenados por fecha de creación ascendente.
    - Map<Especialidad, List<Tecnico>> **findTecnicosInTickets**(): muestra todos los técnicos que han trabajado en algún ticket, agrupados por especialidad.

- Set<Tecnico> **findTecnicosRapidos**(): muestra una lista de los técnicos que han *solucionado* tickets en menos de 5 días.
- Integer **getTotalTicketsRetardados**(): devuelve el total de tickets *cerrados* que han tardado más de una semana en cerrarse desde que se abrieron.
- Double **getMediaResolucionTickets**(Integer prioridad): devuelve la media de días en que se *resuelven* los tickets de una determinada prioridad.
- Map<Tecnico, Double> **getMediaResolucionTicketsGroupByTecnico**(): devuelve un mapa donde para cada técnico que resuelve tickets muestre la media de días en que *resuelve* los tickets.
- Boolean **areAllTicketsFinishedLessThanTenDays**(): dice si todos los tickets han sido o no resueltos en menos de 10 días, solo para los tickets resueltos.
- Optional<TicketSoporte> **getFirstTicketSolvedOneDay**(): devuelve un ticket que haya sido resuelto el mismo día.

SEGUNDA PARTE (4 puntos)

Utiliza los ficheros **usuarios.csv, tecnicos.csv** y **tickets.csv** proporcionados para cargar datos de prueba.

Crea un paquete "**io**" y mete dentro la siguiente clase:

- Una clase **TicketsSoporte** con los siguientes métodos estáticos:
  - **static cargarCSV**(): que cree un objeto ServicioSoporte. Luego lea el fichero usuarios.csv y añada los usuarios del fichero leídos al set de usuarios del objeto ServicioSoporte. Que haga lo mismo para los técnicos. Por último, lea el fichero *tickets*.csv y los añada a la lista de tickets de ServicioSoporte. Fíjate que ya llevan id los Tickets en el fichero y que su técnico y usuario llevan un id, tendrás que buscarlos en sus sets para pasarle objeto Usuario y Tecnico al objeto TicketSoporte. Por último, fíjate que todos los tickets llevan fecha de cierre, aunque estén abiertos, si están abiertos o en proceso no pongas fecha de cierre.
  - **static grabarCSV**(): debe leer los datos en memoria del objeto TicketSoporte y machacar los datos de los ficheros usuarios.csv, tecnicos.csv y tickets.csv.

Crea un paquete "**application**" y mete dentro la siguiente clase:

- Una clase **App** que lleve un main donde comience cargando en memoria todos los datos de los ficheros y luego muestre un **MENÚ** que muestre las siguientes opciones.
  - 1. Listar todos los tickets abiertos
  - 2. Listar técnicos agrupados por especialidad
  - 3. Mostrar el total de tickets resueltos
  - 4. Mostrar la media de resolución de tickets pidiendo la prioridad de los tickets.
  - 6. Insertar ticket soporte: que pida todos los datos y lo inserte en BD.
  - 7. Eliminar ticket soporte: que pida el id y lo elimine de la BD.
  - 8. Salir: que llame a grabarCSV().

ENTREGA (1 punto)

La entrega debe ser un fichero zip que incluya el **proyecto Maven Java** con los ficheros java, jar y javadoc. Indica la versión de Java para probarlo. También puedes subir un enlace a un *repositorio* **Github** donde esté todo el proyecto y los archivos.
En este apartado se valorará también la inclusión de comentarios aclaradores y la limpieza y tabulación del código.