Programación

Control 2ª Evaluación

Problema 1. (3 ptos.)

Realiza un programa Java que simule el algoritmo de planificación de la CPU *Round Robin (quantum=100) con un utilizando* una **cola FIFO**. Los procesos en la cola serán representados por valores numéricos que indican el tiempo de ejecución necesario para cada proceso.

Para probar el programa, en el método `main`, crea 10 procesos con tiempos de ejecución aleatorios entre 100 y 500. Agrégalos a la cola y ejecuta el algoritmo *Round Robin*.

¿Cómo funciona el algoritmo Round Robin?:

El algoritmo de planificación *Round Robin* es uno de los métodos para asignar tiempo de CPU entre los procesos en un sistema operativo. Funciona de la siguiente manera:

- Se define un tiempo fijo llamado "quantum", que es el tiempo máximo que un proceso puede ejecutarse antes de que el siguiente proceso en la cola obtenga su turno.
- Los procesos se colocan en una cola FIFO. El primer proceso en la cola se ejecuta por un tiempo igual al quantum.
- Si un proceso no ha terminado su ejecución al final del quantum, se coloca al final de la cola y el siguiente proceso en la cola obtiene su turno de ejecución. Si el proceso termina dentro del quantum, se retira de la cola y no se vuelve a colocar.
- Este ciclo se repite continuamente hasta que todos los procesos hayan terminado su ejecución.

Problema 2. (7 ptos.) Diseña la clase *TelefonoMovil* teniendo en cuenta las siguientes indicaciones:

- Atributos:
  - marca (*String*): marca del teléfono móvil.
  - modelo (*String*): modelo del teléfono móvil.
  - numeroTf (*String*): número telefónico asociado al teléfono móvil.
  - batería (*int*): nivel de batería del teléfono móvil (0-100).
  - encendido (*boolean*): indica si el teléfono móvil está encendido o apagado.
  - mensajes (*List<Mensaje>*): lista de mensajes enviados y recibidos en el teléfono.
- Constructor:

  Inicializa la marca, modelo y numeroTf con los valores recibidos como parámetro. El nivel de batería se inicializará a 100, la lista de mensajes estará inicialmente vacía y encendido a false.

- Métodos:
  - encender(): método que enciende el teléfono móvil reduciendo la batería en 5 unidades.
  - apagar(): método que apaga el teléfono móvil.
  - hacerLlamada(String numero): método para hacer una llamada a un número telefónico recibido como parámetro. Por cada llamada que se realice la batería se reduce en 10 unidades y se muestra un mensaje indicando que se está llamando a ese número.
  - cargarBateria(int cantidad): método que carga la batería del teléfono móvil. La cantidad de carga debe agregarse al nivel de batería actual, sin exceder 100 unidades.
  - enviarMensaje(Mensaje mensaje): método para enviar un mensaje a un número telefónico. Creará un objeto **Mensaje** y lo añadirá a la lista de mensajes.

    La clase Mensaje dispondrá de los atributos numeroDestinatario (*String*), contenido (*String*), fechaEnvio (*LocalDateTime*), tipo (*String – "enviado" o "recibido"*).
  - verMensajes(): método para ver todos los mensajes enviados y recibidos.

  ⓘ Controla cualquier situación excepcional mediante el lanzamiento de Excepciones.

- Crea una clase **App** con un método main para probar el teléfono móvil.