**Universidad de los Andes**

**Caso 1**

**Infraestructura computacional**

**Departamento de Ingeniería**

Alejandro Ahogado Prieto – 201920701 – a.ahogado

Juan Sebastián Hoyos – 201822167 – js.hoyosm

**Diseño del programa y explicación del programa:**

El código del programa consta de dos clases: productor y buzón. Cada clase representa un objeto abstracto del enunciado, están relacionadas entre si y cumplen distintos objetivos. La clase productor cumple al mismo tiempo el rol de consumidor, tomamos esta decisión de implementar los dos roles en una misma clase puesto que creemos que separarlos en dos clases iba a complicar las cosas innecesariamente. El buzón cumple el rol de monitor.

La clase productor cumple con el propósito de crear los threads que van a retirar, modificar y entregar los mensajes a los buzones. Para lograr esto, cada thread conoce el buzón del cual retira y el buzón al que entrega el mensaje por su posición en el arreglo. Es importante que el buzón notifique al thread (notify) si tiene espacio para aceptar el mensaje que el thread desea insertar, de no cumplirse esta condición, el thread debe empezar una pausa (wait). Esta pausa para la entrega puede ser activa o pasiva, esto lo debe indicar el usuario en un archivo .txt. La pausa activa o pasiva también debe ser indicada para el recibimiento de los mensajes. En el archivo .txt también debe estar indicado el tiempo de espera de transformación (en milisegundos).

El rol de los buzones es el de guardar los mensajes cuando el thread se los entregue, y entregarlos al proceso que desea sacarlos. A su vez como se indicó anteriormente, el buzón debe cumplir el rol de monitor para indicar al thread si debe esperar de forma activa o pasiva, y para lograr esto, el buzón debe conocer quién es el thread que le ingresa el mensaje y quien es el thread que saca los mensajes. El archivo .txt debe dar el id de los buzones (una letra) y la capacidad que tiene cada buzón.

El thread 1 tiene una secuencia de métodos: iniciarMensajes(), transformar(), enviarMensajes() y sacarMensajes(). El método iniciarMensajes() se encarga de crear los mensajes que va a crear dependiendo de la cantidad de mensajes que el usuario le pida que cree, en este método el mensaje va a estar compuesto por un numero 1 y un identificador que ira aumentando según el numero de mensaje. En el método transformar(), lo que va a ocurrir es que el thread va a coger el mensaje que tiene y lo va a editar, poniendo su ID y true o false dependiendo de si recibió y entrego en forma activa o pasiva. En enviarMensajes() el thread va a tomar el mensaje modificado y se lo va a pasar al buzón. Y por último, el método sacarMensajes() tiene la función de ir a un buzón especifico y tomar un mensaje para ser transformado y luego enviado. Por ultimo, al terminar de crear la cantidad de mensajes dada por el usuario, se crea un mensaje FIN, el cual no se modifica pero permitirá que se finalicen todos los threads.

Validación realizada:

Una captura de pantalla de una computadora

Descripción generada automáticamente