PSP - U3 Proyectos

Descargar estos apuntes

Índice

1. Clase psp.proyectos.U3P1_JuegoAdivinaNumero

En este ejercicio tendrás que implementar el típico juego para adivinar un número entre 1 y N pensado al azar por la máquina.

La diferencia respecto al juego tradicional es que ahora hay varios jugadores, en concreto jugaremos con 7 jugadores, que quieren participar todos a la vez. La simulación será parecida a si se jugase en clase, es decir, alguien piensa un número y después todos quieren decir el número, pero se tienen que seguir unas reglas:

- Los jugadores estarán identificados por un número, que formará parte de su nombre: Jugador 1, Jugador 2, Jugador 3, ...
- Todos deben tomarse un tiempo entre 1 y 2 segundos para pensar el número que quieren decir.
- Se jugará por turnos, empezando por el Jugador 1 y terminando por el Jugador 7, volviendo entonces a empezar con el Jugador 1
- El juego acaba cuando uno de los jugadores acierta el número, momento a partir del cual el resto de jugadores ya no proponen más números y finalizan la ejecución.

Un ejemplo de ejecución del programa podría ser:

```
Jugador 1 dice: 9
Jugador 2 dice: 2 <-
Jugador 3 dice: 6
Jugador 4 dice: 7
Jugador 5 dice: 2 <-
Jugador 6 dice: 8
Jugador 7 dice: 4
Jugador 1 dice: 3
Jugador 2 dice: 5
Jugador 2 gana!!!
```

Orden de los jugadores

Los jugadores tienen pensado su número, pero solo lo podrán decir cuando sea su turno. El orden de los jugadores es fijo y no cambia.

Es importante tener en cuenta que, si un jugador ha acertado el número, los demás jugadores ya no pueden seguir diciendo números. Por lo tanto, el juego termina cuando un jugador acierta el número.

Ningún jugador se puede quedar colgado cuando acabe la partida. Si es necesario, indica con un mensaje en la consola cómo los jugadores van abandonando la partida.

Si realizamos una simulación sin tiempos de espera, el juego debería seguir funcionando correctamente.

Productor-Consumidor

Es importante que identifiques si hay productor y/o consumidor, y quién realiza cada rol en este ejercicio. No sólo eso, sino qué información tienen que compartir y cómo se va a compartir.

Además, tendremos que controlar el acceso a la información compartida para que no haya problemas de concurrencia.

Esto nos dará pistas de qué clases, métodos y estructuras de datos podemos usar para compartir la información y controlar el flujo de ejecución del programa.

El número de jugadores se recibe como argumento de la clase principal.

Como extra, se pueden implementar mejoras al programa:

- Evitar que se digan números que ya ha dicho otro jugador
- Programar el juego en Swing

2. Clase psp.proyectos.U3P2_Ascensor

En este ejercicio deberás simular el funcionamiento de un ascensor de un edificio de 8 plantas que dará servicio a distintos vecinos.

Por las restricciones COVID, el ascensor sólo va a poder llevar a un único vecino a la vez.

Sobre el ascensor, es importante conocer

- Si el ascensor está parado o en movimiento. Ten en cuenta que cuando alguien llama al ascensor desde un piso, el ascensor acudirá cuando pueda. Si el ascensor está parado, irá inmediatamente. En cambio, si el ascensor está en movimiento, acudirá cuando termine de subir o bajar como tenía previsto.
- El piso en el que se encuentra el ascensor.
- Que el ascensor iniciará la subida o bajada, cuando esté parado, pero no cambiará su destino mientras esté en movimiento. Entre piso y piso el ascensor tardará 300ms.
- Dada la posición del ascensor y el piso desde el que los usuarios hilos le llaman, controlar si el ascensor sube o baja. Ten en cuenta que varios usuarios pueden llamar al ascensor a la vez desde distintos sitios, pero el ascensor solo cambiará su movimiento cuando esté parado. (No vamos a gestionar un algoritmo de proximidad).

Se deberá mostrar un mensaje que indique en qué piso se encuentra el ascensor en cada momento, qué dirección lleva y si lleva a algún usuario. Por ejemplo:

> **Ascensor en el piso 2. Subiendo hasta el piso 8 (Juan)** --> Está llevando a un vecino
> **Ascensor en el piso 7. Bajando hasta el piso 4 ( )** --> Va a recoger a algún vecino

Cuando el ascensor llegue al destino también se deberá mostrar un mensaje indicando el nombre del usuario que ha llevado a su piso. Por ejemplo:

> **Ascensor parado en piso 8. Dejando a Pepe** --> Ha llegado al destino con un vecino
> **Ascensor parado en piso 4. Recogiendo a Juan** --> Ha llegado al origen a por un vecino

Los vecinos comparten el ascensor. Estos llamarán al ascensor para que los recoja en un piso y los deje en otro. Por tanto un vecino llamará al ascensor para ir de un piso a otro. Cuando un vecino llame al ascensor se mostrará un mensaje advirtiendo de esta circunstancia. Por ejemplo:

> **Manolita está esperando en el piso 2 para subir al piso 5**
> **Juan está esperando en el piso 3 para bajar al piso 2**

En nuestra simulación, cada vecino llamará al ascensor varias veces (defínelo como constante en el código, inicialmente con valor 3) para ver el comportamiento del ascensor cuando le llaman de forma simultánea.

Entre llamadas el ascensor los vecinos dejarán pasar 1 segundo.

Para realizar las pruebas vamos a tener a 3 vecinos “Manolita”, “Pepe” y “Juan”. Cada uno de los cuales se encontrará inicialmente en un determinado piso (al azar) e irán realizando llamadas para ir a diferentes pisos del edificio (también al azar). Ten en cuenta que el edificio consta de los pisos 0 al 8.

3. Clase psp.proyectos.U3P3_Galaxia

La galaxia BOREAX está formada por un conjunto de varias estrellas, nubes de gas, planetas y polvo cósmico. Dos de estos planetas, PHINEAS y FERB están comunicados a través de un pasadizo interplanetario, construido por los primeros seres que habitaron estos planetas. Las criaturas de estos reinos planetarios se desplazan a través de naves.

El pasadizo es muy estrecho y no permite el cruce de dos naves que circulen en sentidos contrarios. No es posible saber, debido a la insuficiente tecnología, si hay una nave empezando a cruzar el pasadizo desde el otro planeta.

Se han instalado cuatro sensores y dos barreras. Dos sensores indican la llegada de aeronaves en cada dirección, emitiendo una señal por cada nave que pasa, y los otros dos indican las salidas respectivas del pasadizo. Las barreras bloquean las entradas al pasadizo, y cuando reciben una señal se levantan para permitir el paso de una aeronave (y sólo una); acto seguido se vuelven a cerrar.

Diseña un programa concurrente que simule el sistema descrito, incluyendo un proceso que emplee las señales recibidas de los sensores para emitir señales apropiadas a las barreras, de manera que:

- Nunca crucen simultáneamente el pasadizo interplanetario dos aeronaves en sentidos opuestos.
- Ninguna aeronave espere eternamente a pasar.
- Todas las naves pasen lo más pronto posible salvo violación de las norma 1 o 2.
- Las señales de los sensores han de atenderse inmediatamente (o al menos lo más pronto posible), para no “perder” el paso de ninguna nave.

Podemos resolver el problema modelizando el pasadizo interplanetario mediante un monitor, de manera que aseguremos que sus métodos sean ejecutados en exclusión mutua y pueda ser utilizado sin peligro por más de un hilo (aeronave) en ejecución.

Ten en cuenta que las naves de un sentido deben esperar a un objeto y los del otro sentido a otro diferente. No tiene sentido poner a todas las aeronaves, sea cual sea su sentido, esperando en una misma cola para un único objeto.

Podemos construir un objeto monitor (pasadizo) principal al que le pasemos dos auxiliares, uno para cada sentido. De esta forma podemos abstraernos desde la clase principal de esta peculiaridad en la implementación, y será el monitor principal el que se encargará de que todo funcione correctamente.

Por otro lado, ten en cuenta que la ocupación máxima del pasadizo es de 5 naves y el número máximo de naves que pueden pasar seguidas es de 3.

4. Clase psp.proyectos.U3P4_Carrera

Crea una aplicación gráfica con Swing que simule una carrera entre 5 atletas.

El avance de cada corredor se mostrará con una barra de progreso (valor máximo 1000).

Los atletas irán avanzando en orden, es decir, uno detrás de otro, un número aleatorio entre 1 y 10, dejando 100ms entre movimientos.

Cuando un corredor llegue a la meta, todos los demás deben dejar de correr.

En el diseño tendremos además de las barras de progreso, controles para mostrar la información de cada corredor (avance total y último avance) junto con un botón que servirá para comenzar la carrera.

También tendremos algún control que, una vez finalizada la carrera nos muestre cuál de los corredores ha sido el ganador.