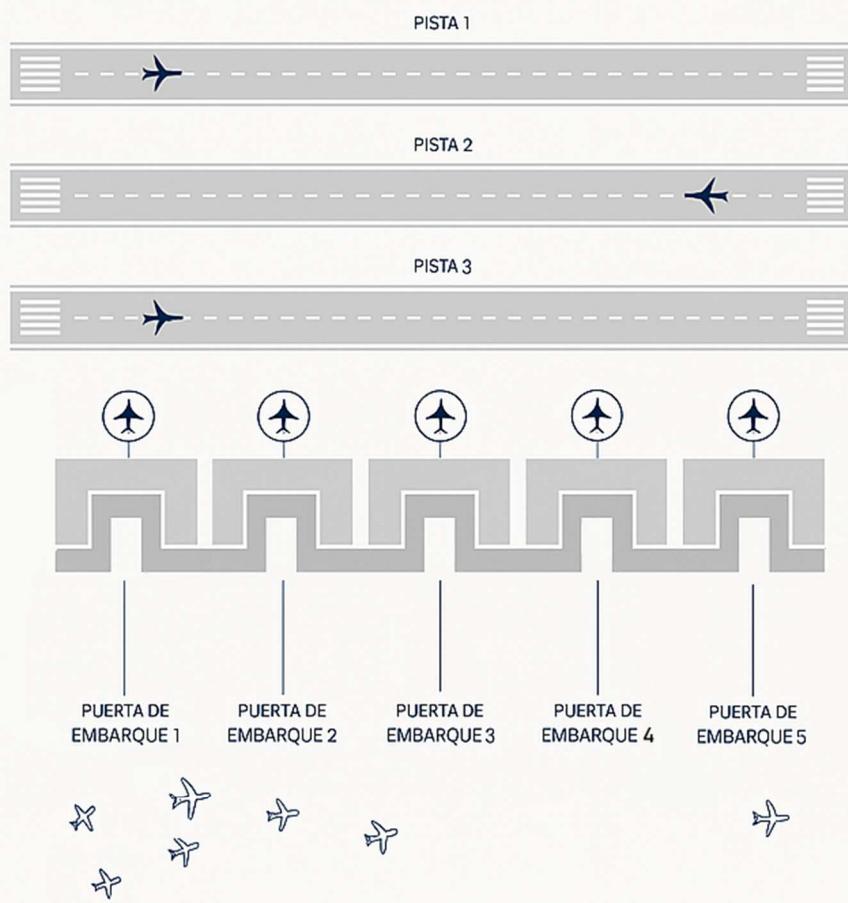




Proyecto de la asignatura *Programación concurrente y distribuida*
simulando el comportamiento de un aeropuerto.

DIAGRAMA DE OPERACIONES AEROPUERTO AERON



RECURSOS CLAVE:
3 PISTAS
5 PUERTAS DE EMBARQUE
20 AVIONES ACTIVOS
1 TORRE DE CONTROL

Tabla de contenido

 Objetivo principal	3
 Contexto	3
 Recursos del aeropuerto.....	3
 Modos de Implementación	3
 Restricciones de Concurrencia	4
 Pistas para el desarrollo	4
 Cosas para tener en cuenta	4
 Finalización del Simulador	5
 Operaciones de un Avión	5
 Objetivos de Aprendizaje	5
 Evaluación	6

Objetivo principal

Simular el funcionamiento concurrente de un aeropuerto con recursos limitados (pistas y puertas de embarque), aplicando los distintos mecanismos de sincronización estudiados en la asignatura:

- Implementación secuencial.
- Implementación con concurrencia (semáforos y monitores)

El objetivo es comprender la necesidad de coordinación entre procesos concurrentes, el uso correcto de recursos compartidos y las diferencias entre los mecanismos de sincronización.

Contexto

El **Aeropuerto Extremeño de Regulación y Operaciones de Navegación (AERON)** opera con **20 aviones** que llegan y parten en un mismo periodo de tiempo.

Cada **avión** se representa como una entidad independiente que realiza sus operaciones dentro del aeropuerto, siguiendo unas reglas y tiempos predefinidos.

El aeropuerto dispone de un número **limitado de recursos**, lo que genera la necesidad de establecer mecanismos de control para evitar conflictos o esperas indefinidas.

Recursos del aeropuerto

Elemento	Descripción
Pistas	3 pistas disponibles para aterrizaje o despegue simultáneo.
Puertas de embarque	5 puertas disponibles para estacionar aviones.
Aviones activos	20 aviones operando durante el mismo periodo.
Torre de control	1 torre de control para controlar el flujo aéreo.
Panel de vuelos	1 panel de vuelo para ver el estado de los aviones

Modos de Implementación

La práctica se compone de dos modos de ejecución, cada modo deberá mantener las mismas reglas de operación de los aviones, pero empleando diferentes técnicas de coordinación.

Modo Secuencial:

Ejecución sin concurrencia real. Los aviones operan uno tras otro.

Modo con concurrencia:

Ejecución concurrente controlando el acceso a recursos mediante mecanismos de sincronización controlada mediante estructuras que gestionan de forma automática la exclusión mutua y la sincronización de los hilos (semáforos y monitores)

Restricciones de Conurrencia

- **Cada pista** solo puede ser usada **por un avión** de forma **simultáneamente** (ya sea para aterrizar o despegar).
- Solo hasta **5 aviones** pueden estar **estacionados** en puertas al mismo tiempo (uno por puerta).
- Si no hay recursos disponibles, los aviones deben **esperar** hasta que se liberen.
- No puede producirse una situación de **bloqueo** o **interbloqueo**.

Pistas para el desarrollo

- Se encuentran escondidos el uso de los tres problemas clásicos (productor-consumidor, lectores-escritores y comida de filósofos) queda a responsabilidad del programador elegir bien el sitio donde se tienen que hacer uso de estos problemas clásicos.

Cosas para tener en cuenta

- Los **pasajeros serán simulados**, no hace falta declarar varios pasajeros para cada avión, tan solo declararemos **un pasajero por avión**.
- El panel de vuelos refleja en todo momento el estado que se encuentran los 20 aviones
- Se debe establecer una **disposición de ventanas** para mostrar lo que ocurre en el aeropuerto en todo momento (uso de la clase JFrame de java, visto en clase como la clase Ventana)
 - o Una ventana para mostrar los eventos de los aviones (cambio de estados de los aviones y peticiones a la torre de control)
 - o Una ventana para mostrar lo que ocurre dentro de la torre de control (cuando procesa las peticiones de los aviones, que pista y puerta asigna, cuando libera una pista o puerta)
 - o Una ventana para hacer el panel de vuelos, mostrando en todo momento el estado actualizado de los aviones.
- La implementación de al menos un problema clásico tiene que estar realizado con semáforos y al menos uno con monitores.
- El uso de semáforos y monitores debe ser el correcto para el total funcionamiento lógico del aeropuerto.
- La especificación de atributos y métodos de cada clase y el funcionamiento de estos quedan a elección del programador.
- Para dar más realismo en la ejecución podéis poner identificadores a los aviones y puertas de embarque para que parezcan más realista
- **Se debe entregar a parte del código, un informe sobre como inicializar el proyecto y como habéis constituido el proyecto.**

🏁 Finalización del Simulador

El simulador finalizará cuando **todos los aviones hayan completado su ciclo completo de operaciones**. Es decir, estén en el aire, aterricen y despeguen de nuevo.

Deberá medirse el **tiempo total de operación** de cada avión, desde el momento en que solicita pista para aterrizar hasta que completa el despegue.

Al finalizar la simulación, se generará un archivo CSV a modo de **tabla resumen** con la siguiente información:

Avión	Tiempo total (ms)	Observaciones
Avión 1	423	3º
Avión 2	408	1º
Avión 3	419	2º

✈️ Operaciones de un Avión

Cada avión debe completar el siguiente ciclo de operaciones para completar con éxito su acometido.

Solicitar una pista de aterrizaje y puerta de embarque a la torre de control.	 Solicita pista y puerta
Aterrizar (100 ms).	 Aterriza
Ocupa la puerta de embarque	 Ocupa puerta
Permanecer estacionado en la puerta de embarque.	 Estacionado en puerta
Los pasajeros suben al avión	 Suben pasajeros
Liberan puerta de embarque	 Liberan puerta
Solicitar nuevamente una pista para despegar.	 Solicita pista
Despegar (100 ms).	 Despega
Finalizar su operación.	 Fin (Se queda en estado en el aire)

🧠 Objetivos de Aprendizaje

- Comprender los problemas inherentes a la programación concurrente.
- Analizar la sincronización y coordinación de procesos o hilos.
- Observar el comportamiento de los sistemas concurrentes bajo restricciones de recursos.
- Hacer uso de los distintos mecanismos de sincronización en programación concurrente para el manejo de sistemas concurrente.
- Comparar el efecto de distintos mecanismos de sincronización sobre el resultado final.



Evaluación

Bloque	Criterio	%	Insuficiente (0)	Aceptable (5)	Bueno (7)	Excelente (10)
Implementación secuencial	Diseño de clases y entidades	5	Código monolítico o mal diseñado	Estructura funcional pero desorganizada	Clases bien diferenciadas, leves redundancias	Modelado limpio, relaciones claras, buena encapsulación
	Flujo lógico secuencial	5	No ejecuta ciclo completo	Errores menores de estado	Secuencia funcional con leves incoherencias	Simulación completa y coherente
Implementación concurrente	Uso correcto de hilos y concurrencia	10	Deficiente o ausente	Concurrencia parcial	Concurrente con leves interferencias	Hilos independientes y ejecución estable
	Aplicación de semáforos (problema clásico)	20	Incorrecta o ausente	Parcialmente funcional	Leve sobre/sincronización	Correcta implementación sin interbloqueos
	Aplicación de monitores (problema clásico)	20	Incorrecto o no usado	Básico o incompleto	Correcto con leves defectos	Monitor bien encapsulado y correcto
	Evita condiciones de carrera y bloqueos	30	Deadlocks frecuentes	Bloqueos puntuales	Esperas leves	Sin deadlocks ni esperas activas
Documentación	Código y comentarios	2.5	Sin comentarios o ilegible	Escasos o mal ubicados	Comentarios suficientes	Estructurado, comentado y legible
	Ventana de eventos de aviones, torre de control y panel de vuelos	2.5	No muestra correctamente	Eventos incompletos	Funcional con leve retardo	Actualización en tiempo real
	Uso de excepciones	2.5	No usa excepciones	Usa menos de 2 excepciones	Usa al menos 3 excepciones	Usa 5 o más excepciones
Resultados y archivo de salida	Medición de tiempo por avión y log de aeropuerto	2.5	Sin CSV o erróneo	CSV incompleto	CSV con tiempos aproximados	CSV preciso y correcto