**FIRST DATES**



Crear un programa que use sincronización de hilos para resolver el siguiente problema.

Estamos en el restaurante del programa de TV First Dates. Hay 5 mesas para 2 personas cada una. Cada mesa tiene un número (AUTOINCREMENTADO) entre 1 y 5.

Hay 10 personas que van al programa a encontrar pareja.

De cada persona conocemos:

* Número: AUTOINCREMENTADO a partir de 1
* Nombre: NO NULO, NO VACÍO, NO BLANCO, LONGITUD >=2
* Género: NO NULO
  + MASCULINO
  + FEMENINO
* Orientación: NO NULO
  + ASEXUAL
  + BISEXUAL
  + HETEROSEXUAL
  + HOMOSEXUAL

**Lista de personas**

|  |  |  |  |
| --- | --- | --- | --- |
| # | NOMBRE | GÉNERO | ORIENTACIÓN |
| 1 | AMANCIO | MASCULINO | HETEROSEXUAL |
| 2 | TIBURCIO | MASCULINO | HETEROSEXUAL |
| 3 | IGOR | MASCULINO | HOMOSEXUAL |
| 4 | DOROTEA | FEMENINO | HETEROSEXUAL |
| 5 | DULCINEA | FEMENINO | HOMOSEXUAL |
| 6 | LIZBETH | FEMENINO | HETEROSEXUAL |
| 7 | LEOVIGILDO | MASCULINO | BISEXUAL |
| 8 | ODISEA | FEMENINO | BISEXUAL |
| 9 | ADALBERTO | MASCULINO | ASEXUAL |
| 10 | GALATEA | FEMENINO | ASEXUAL |

Dos personas son compatibles según su género y orientación. Estas son las combinaciones compatibles:

1. ASEXUAL
   1. Sólo es compatible con otra persona ASEXUAL
2. BISEXUAL
   1. BISEXUAL
   2. HETEROSEXUAL: con género contrario
   3. HOMOSEXUAL: con el mismo género
3. HETEROSEXUAL
   1. BISEXUAL: con género contrario
   2. HETEROSEXUAL: con género contrario
4. HOMOSEXUAL
   1. BISEXUAL: con el mismo género
   2. HOMOSEXUAL: con el mismo género

**[DESARROLLO]**

Cada vez que una persona entra en el restaurante:

1. Piensa un tiempo aleatorio TIEMPO1 [0-5] segundos
2. Busca aleatoriamente una mesa y se intenta sentar.
   1. Si la mesa está vacía se sienta y dice: “Me he sentado en la mesa %d y espero a mi futuro amor”, mesa. El hilo termina su ejecución.
   2. Si hay sentada ya una persona entonces:
      1. Si son compatibles se sienta y dice: “Me he sentado en la mesa %d con %s y somos compatibles ❤[[1]](#footnote-1)”, número de mesa, persona. El hilo termina su ejecución.
      2. Si no son compatibles no se sienta en la mesa y dice: “Me he encontrado en la mesa %d con %s y no somos compatibles desgraciadamente 💔[[2]](#footnote-2)”. Vuelve al punto 1.
   3. Si no hay sitio (ya hay dos personas sentadas) entonces dice: “¡Perdón! Que vaya bien vuestra cena 😳[[3]](#footnote-3)”. Vuelve al punto 1.

**[TERMINACIÓN]** El programa principal (método main) termina cuando todas las personas están sentadas en alguna mesa.

El método toString() de persona debe tener el siguiente formato:

[número] [nombre] [GENERO] [ORIENTACION]

El método toString() de mesa devuelve:

[número] [persona1] [persona2]

**[MENSAJES]** Todos los mensajes tendrán el siguiente formato:

[fecha-hora con precisión de nanosegundos] [persona] texto

Se debe usar la ayuda aportada para fechas y horas.

**SE PIDE**

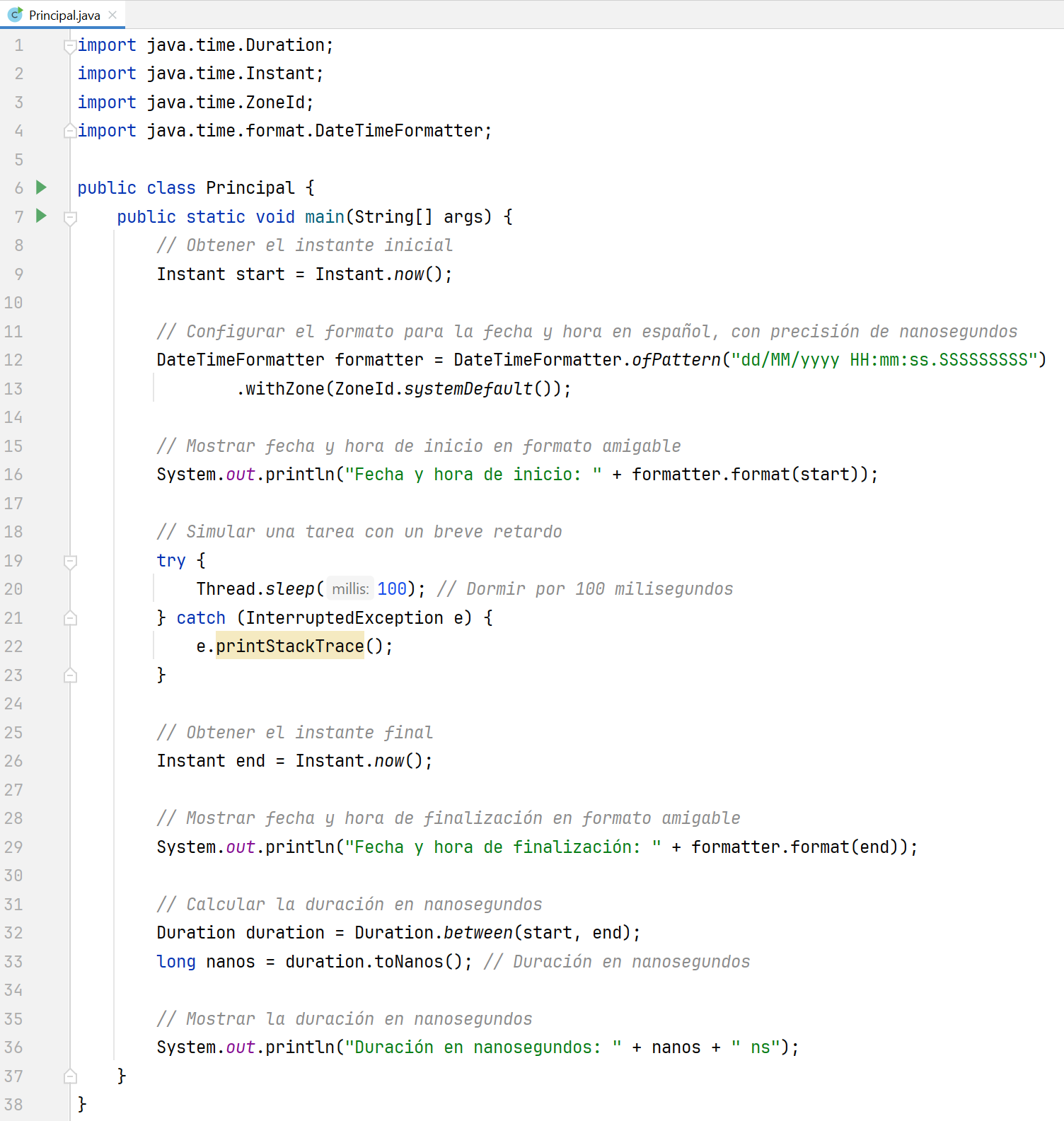
1. [1p] Clase Principal. El hilo principal debe esperar la finalización del resto de hilos. Se debe usar el método **awaitTermination** de ExecutorService. Al terminar debe mostrar todas las mesas y las parejas que se han sentado en cada mesa.
2. [0,25p] Tipos enumerados
   1. Genero
   2. Orientacion
3. [0,75p] Clase Persona
4. [0,25p] Clase Mesa
5. [0,5p] Clase Fabrica. Clase <<utility>> que genera las personas y mesas del problema
6. [0,25p] DESARROLLO. Apartado 1. Piensa
7. [2,5p] DESARROLLO. Apartado 2. [0,5p] Busca aleatoriamente
   1. [0,5p] Mesa vacía
   2. [1p] Una persona
      1. [0,5p] Compatibles
      2. [0,5p] No compatibles
   3. [0,5p] Mesa llena. No hay sitio
8. [0,5p] Implementación de los mecanismos de sincronización. Exclusión mutua. Objetos compartidos.
9. [0,5p] [MENSAJES] En cada paso se mostrará por pantalla mensajes con el formato indicado.
10. [0,25p] Usar ExecutorService con un conjunto (**pool**) de hilos fijo.
11. [0,25p] [TERMINACIÓN] Finalización de los hilos correcta. Al terminar el programa no debe quedar ningún hilo ejecutándose
12. [0,25p] ¿En qué caso se puede dar un interbloqueo? Justifica tu respuesta
13. [0,75p] ¿Qué mecanismo de sincronización has usado? ¿Qué objetos son compartidos y debe ser controlado su acceso para conseguir la exclusión mutua? Justifica tu respuesta
14. [2p] Estructura general del programa (legibilidad, modularidad, buen uso de los conceptos, …). **Se penalizará el uso de clases adicionales y métodos que no sean necesarios para el resolver el problema.**

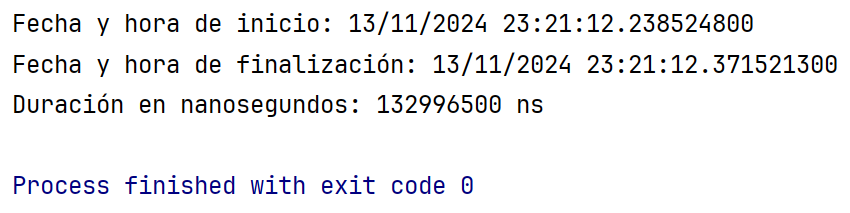
**REQUISITOS DE OBLIGADO CUMPLIMIENTO**

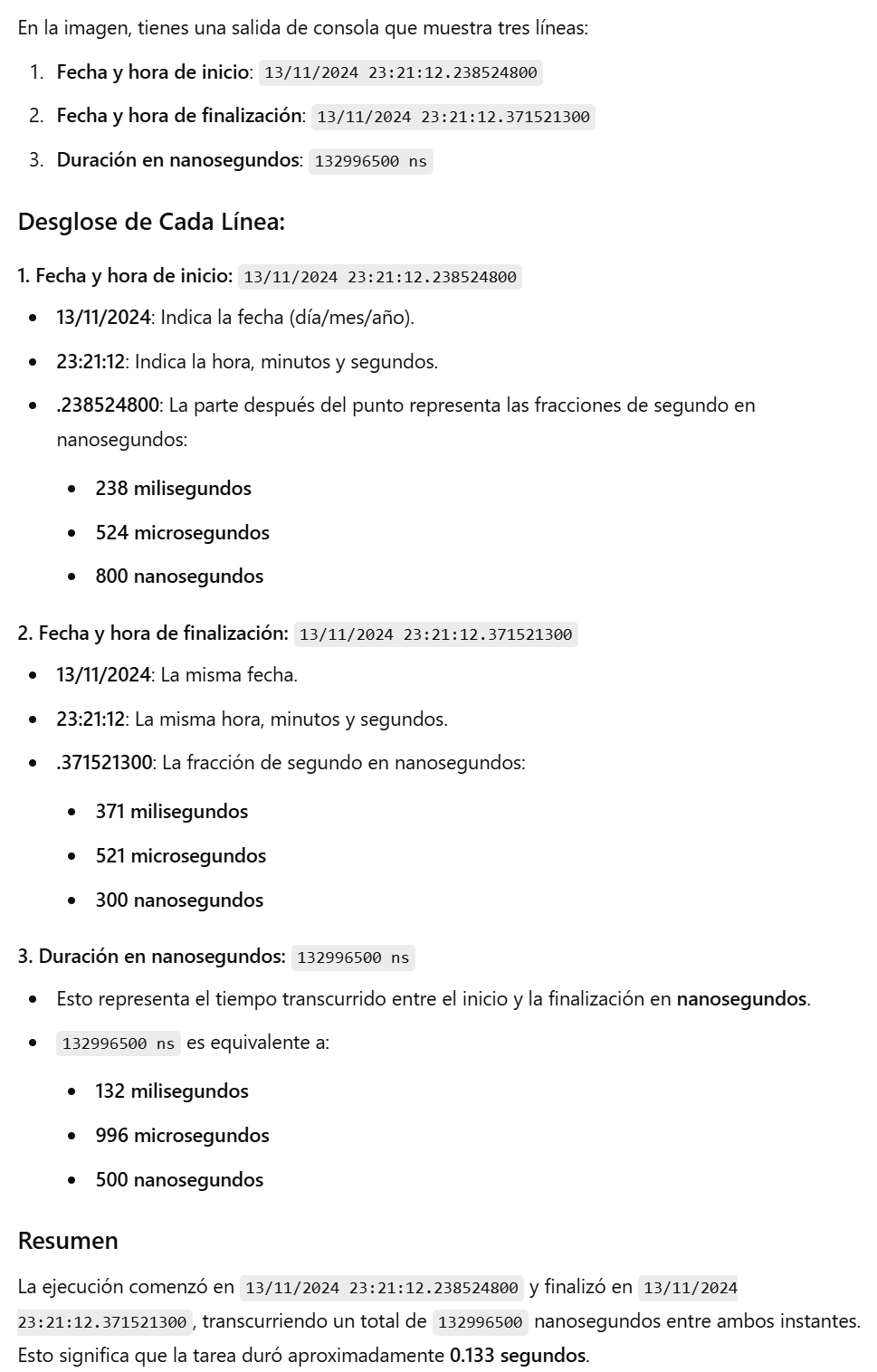
* El examen se realizará obligatoriamente usando el entorno de desarrollo IntelliJ
* Copiar en el examen tiene una calificación de 0 y supondrá la adopción de medidas disciplinarias con el alumnado implicado
* El profesor indicará al alumnado con qué ordenador realizará el examen y el sitio que ocupará en el aula para realizarlo
* Entregar la carpeta COMPLETA del proyecto de todas las versiones solicitadas por el profesor durante el examen
* La entrega se debe realizar en el tiempo y forma indicados por el profesor
* No disponer ni usar conexión a internet ni de red local durante el examen
* No se pueden usar ningún tipo de asistentes de IA ni remotos ni locales
* Para aquellos exámenes sospechosamente parecidos entre ellos habrá un examen oral para demostrar la autoría del mismo
* Realizar copia de la carpeta del proyecto cuando el profesor lo indique
* Respetar los conceptos del paradigma orientado a objetos (legibilidad, ocultación de datos, reusabilidad, robustez, …)



AYUDA PARA USO DE FECHAS Y HORAS







1. Corazón rojo: \u2764 [↑](#footnote-ref-1)
2. Corazón roto: \uD83D\uDC94 [↑](#footnote-ref-2)
3. Cara con mejillas sonrojadas (UTF-16): \uD83D\uDE33 [↑](#footnote-ref-3)