

Sistemas Concurrentes y distribuidos



Universidad
Zaragoza

Practica 2

Pablo Villa
Alvaro Perez

Índice

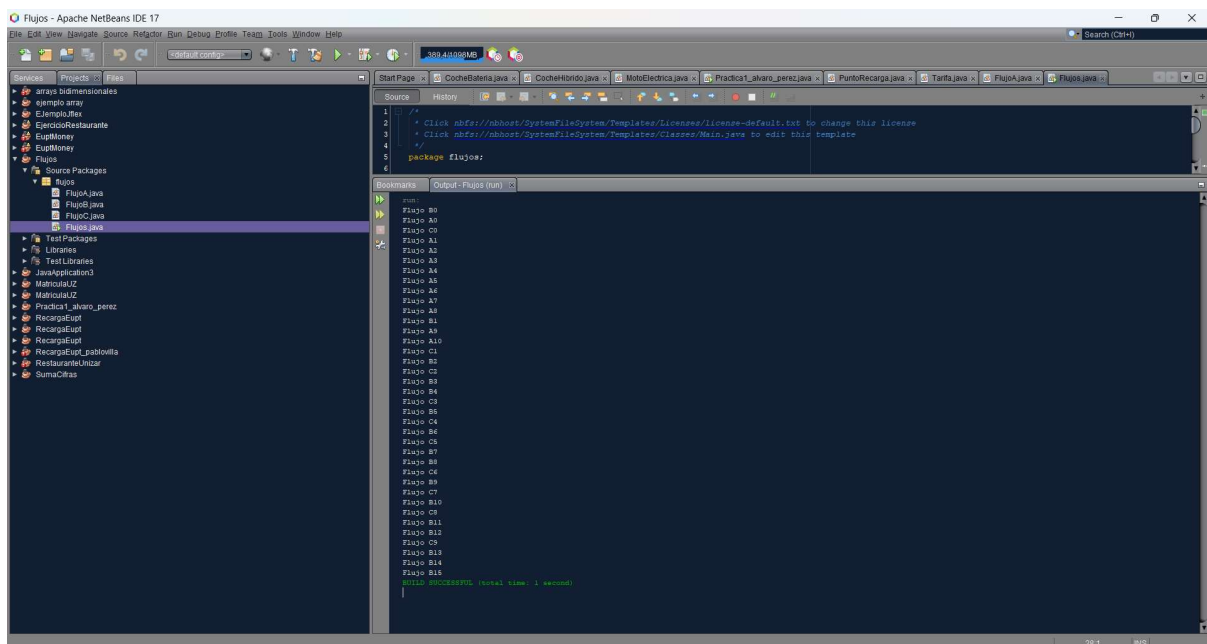
Portada	-----	1
Índice	-----	2
Ejercicio 1	-----	3
Ejercicio 2	-----	4

Ejercicio 1

1) En el lenguaje Java. Los flujos de ejecución se corresponderá con hilos de ejecución (clase que hereda de Thread).

Para este ejercicio en java crearemos 3 clases que serán clase hijo de La clase thread en ellas crearemos un tiempo de espera con Math.random. Utilizaremos el método public void run con un try y un catch el try para imprimir en pantalla el mensaje con un sleep que será el tiempo de espera . Además el catch que interrumpirá y hará que termine.

En el main Crearemos un objeto de cada clase y utilizaremos el .start que ejecutara el hilo



2) En el lenguaje Ada. Los flujos de ejecución se corresponden con tareas (task) de Ada. Debe recordarse que, al crear tareas en Ada, por defecto éstas arrancan de manera implícita al arrancar el programa principal. Por otro lado, para crear números aleatorios, se puede utilizar el paquete `Ada.Numerics.Discrete_Random`.




Posteriormente, llamando a `Azar.Random(generaAzar)` se generará un número aleatorio en el rango antes especificado. Por otro lado, para hacer que la tarea espere se llamará a `delay Duration(tiempo)` donde tiempo será un número, en este caso el obtenido por el generador de números aleatorios.

Pasos a seguir para la ejecución del programa:

1. Declaramos un subtipo llamado *segundos*, el cual es tipo Integer con un rango que va desde el 1 hasta 7, esto determinara el rango en el que queramos que esté comprendido nuestro tiempo de retardo.
2. Utilizaremos el *Ada.Numerics.Discrete_Random* para generar números aleatorios en el rango definido por *segundos*.
3. Declararemos tres tareas las cuales se ejecutarán de forma concurrente simulando un flujo de ejecución diferente.
4. En cada una de las tareas realizaremos un bucle el cual realizara tantas iteraciones como le hayamos asignado, de esta forma imprimira cada vez que se ejecute el bucle el siguiente mensaje(FlujoX, I), siendo I el número de iteraciones por el que va.

En resumen lo que conseguimos con este programa es imprimir por pantalla mensajes con retrasos aleatorios, cada uno con un nombre diferente(Flujo A, Flujo B, Flujo C) y un número variable de iteraciones. Estas tareas se ejecutarán en paralelo, es decir, que empezaran todas las iteraciones al mismo tiempo pero dependiendo del número de iteraciones que se tengan que ejecutar para cada Tarea una acabará antes que otra.

```
File Edit Navigate Find Code VCS Build SPARK Analyze Debug View Window Help
Locations Run: ejercicio1.exe
C:\Users\Alvaro\OneDrive\Escritorio\PRACTICAS ADA\PRACTICA2_870097_874773\obj\ejercicio1.exe
Flujo A 1
Flujo B 1
Flujo C 1
Flujo B 2
Flujo C 2
Flujo A 2
Flujo B 3
Flujo C 3
Flujo B 4
Flujo B 5
Flujo A 3
Flujo C 4
Flujo B 6
Flujo A 4
Flujo C 5
Flujo A 5
Flujo A 6
Flujo B 7
Flujo C 6
Flujo B 8
Flujo A 7
Flujo C 7
Flujo A 8
Flujo B 9
Flujo C 8
Flujo A 9
Flujo A 10
Flujo B 10
Flujo C 9
Flujo B 11
Flujo B 12
Flujo B 13
Flujo B 14
Flujo B 15
[2023-10-03 17:21:07] process terminated successfully, elapsed time: 00.97s
```



```
Flujo A0
Flujo C0
Flujo B0
Flujo B1
Flujo A1
Flujo B2
Flujo C1
Flujo A2
Flujo B3
Flujo B4
Flujo A3
Flujo C2
Flujo B5
Flujo A4
Flujo B6
Flujo A5
Flujo C3
Flujo B7
Flujo B8
Flujo A6
Flujo B9
Flujo C4
Flujo A7
Flujo B10
Flujo A8
Flujo B11
Flujo C5
Flujo B12
Flujo A9
Flujo B13
Flujo A10
Flujo C6
Flujo B14
Flujo B15
Flujo C7
Flujo C8
Flujo C9
BUILD SUCCESSFUL (total time: 8 seconds)
```