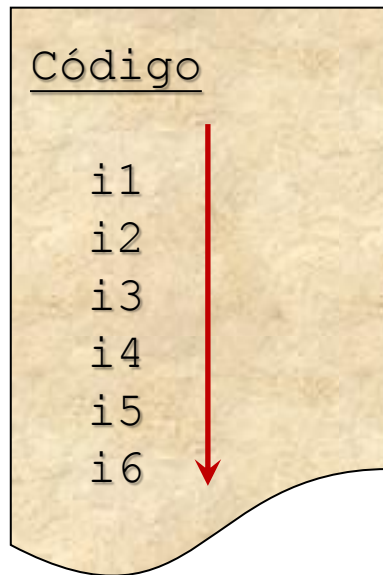


Infraestructura computacional

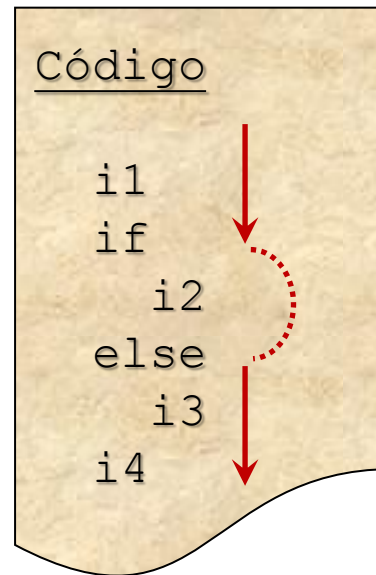
Concurrencia

¿Qué es concurrencia?

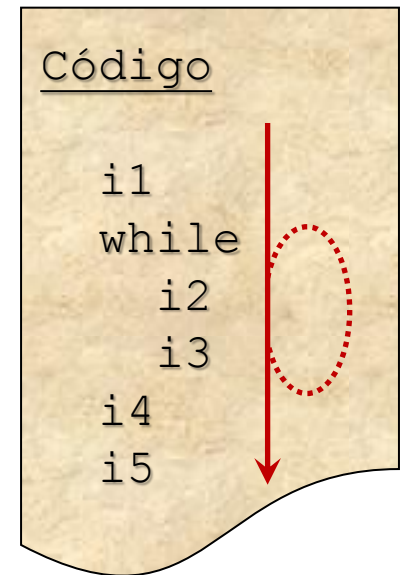
Ejecución secuencial



Ejecución condicional



Ejecución iterativa



¿Qué es concurrencia?

Ejecución concurrente

Código

i1
bifurcar
i2
i3
i4
i5
i6
i7
i8

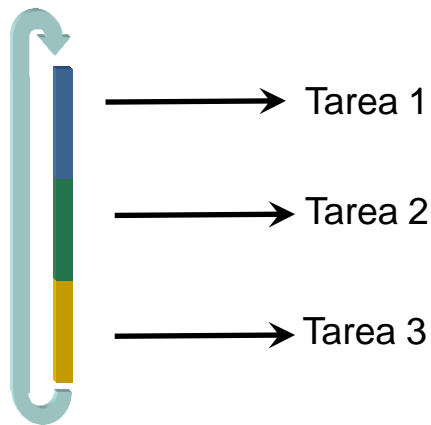


¿Por qué concurrencia?

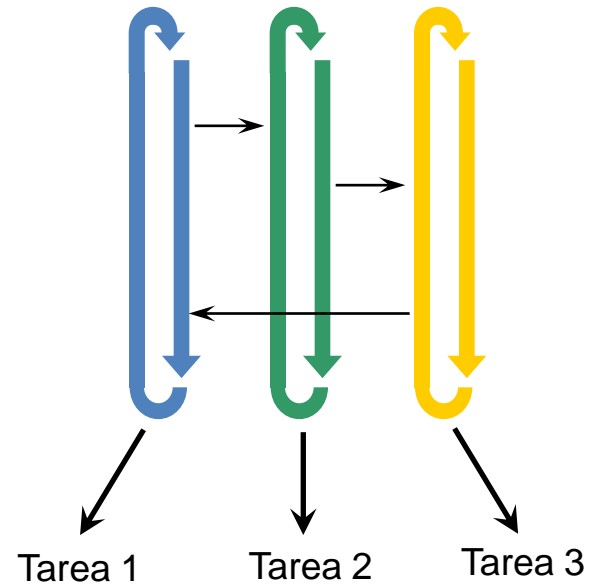
- Facilita modelado de tareas diferentes pero simultáneas
 - Mejor encapsulamiento
 - Mejor separación de problemas
- Mayor reactividad
- Mejor desempeño
- Más adecuado para ambientes modernos (red, GUI, multitarea)

¿Por qué concurrencia?

Programación monolítica



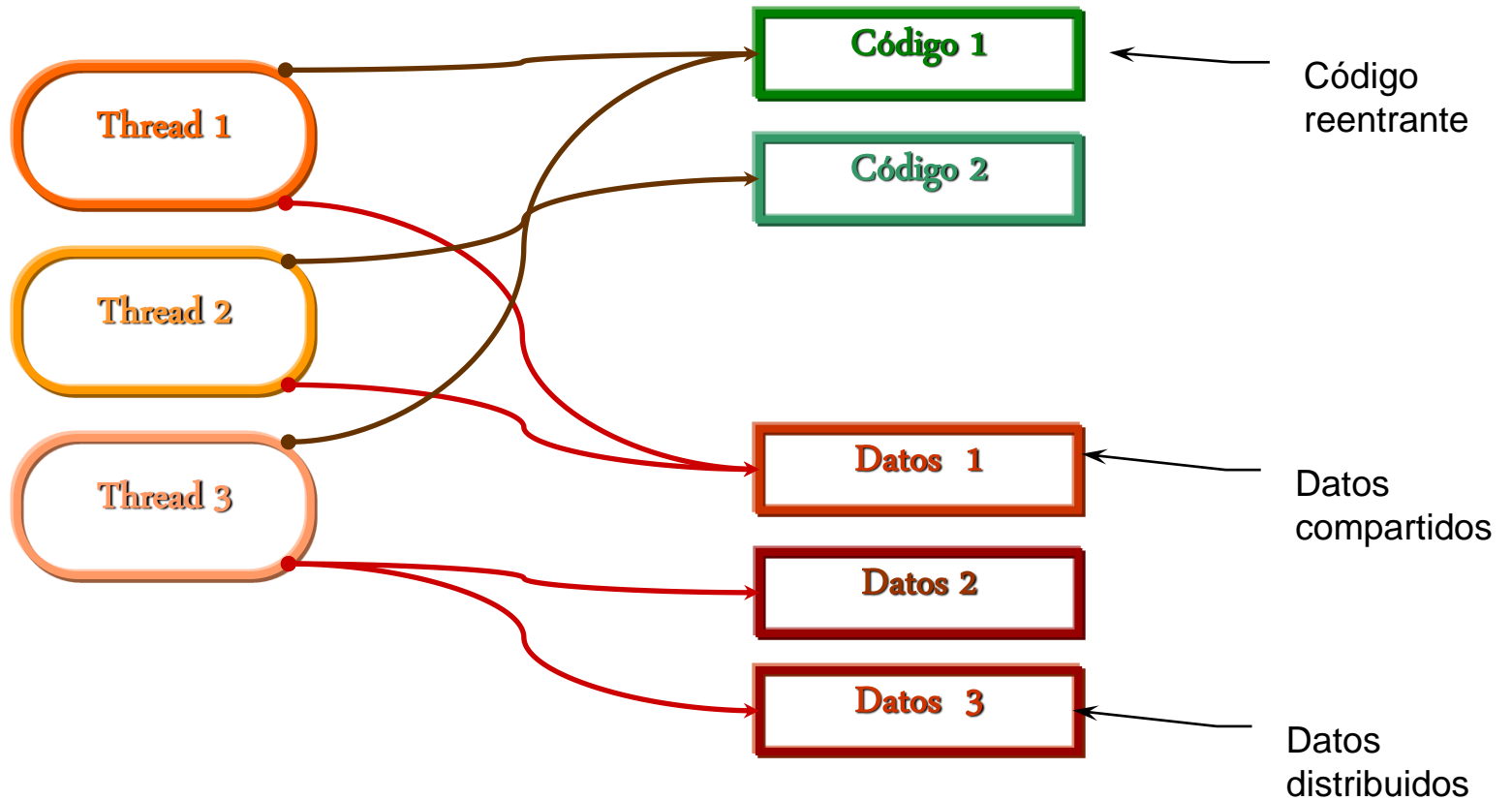
Programación concurrente



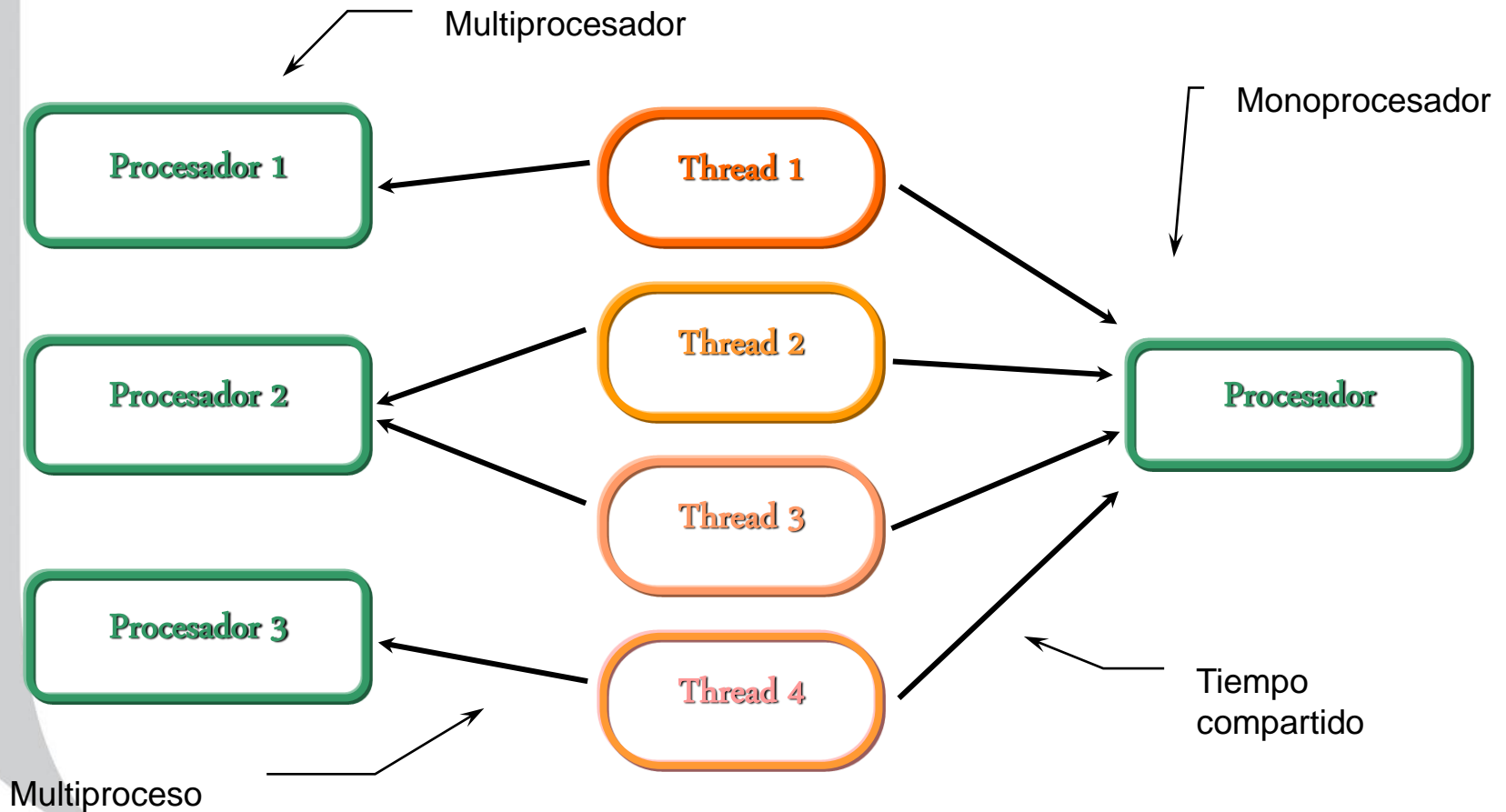
Niveles de concurrencia

- Circuitos
- Instrucciones (ILP)
- Hilos (*threads*) de un proceso
- Procesos locales
- Procesos remotos

¿Cómo es la concurrencia?

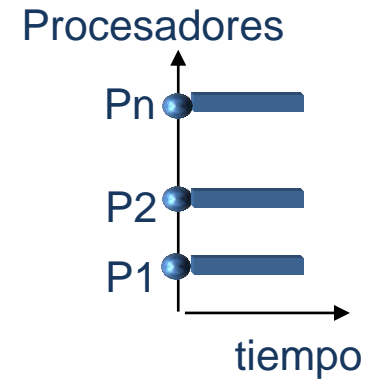
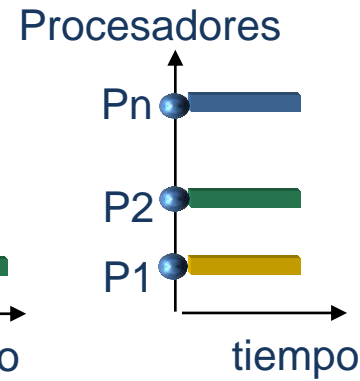
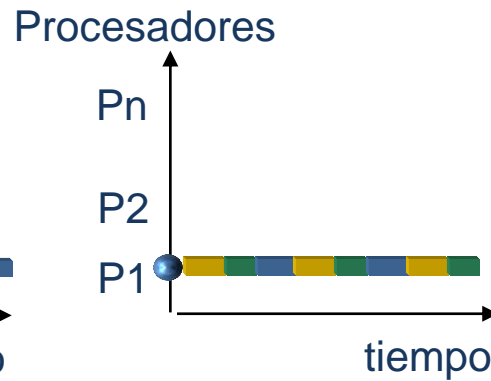
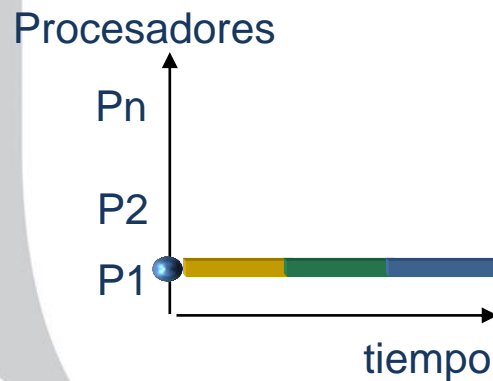


¿Cómo es la concurrencia?

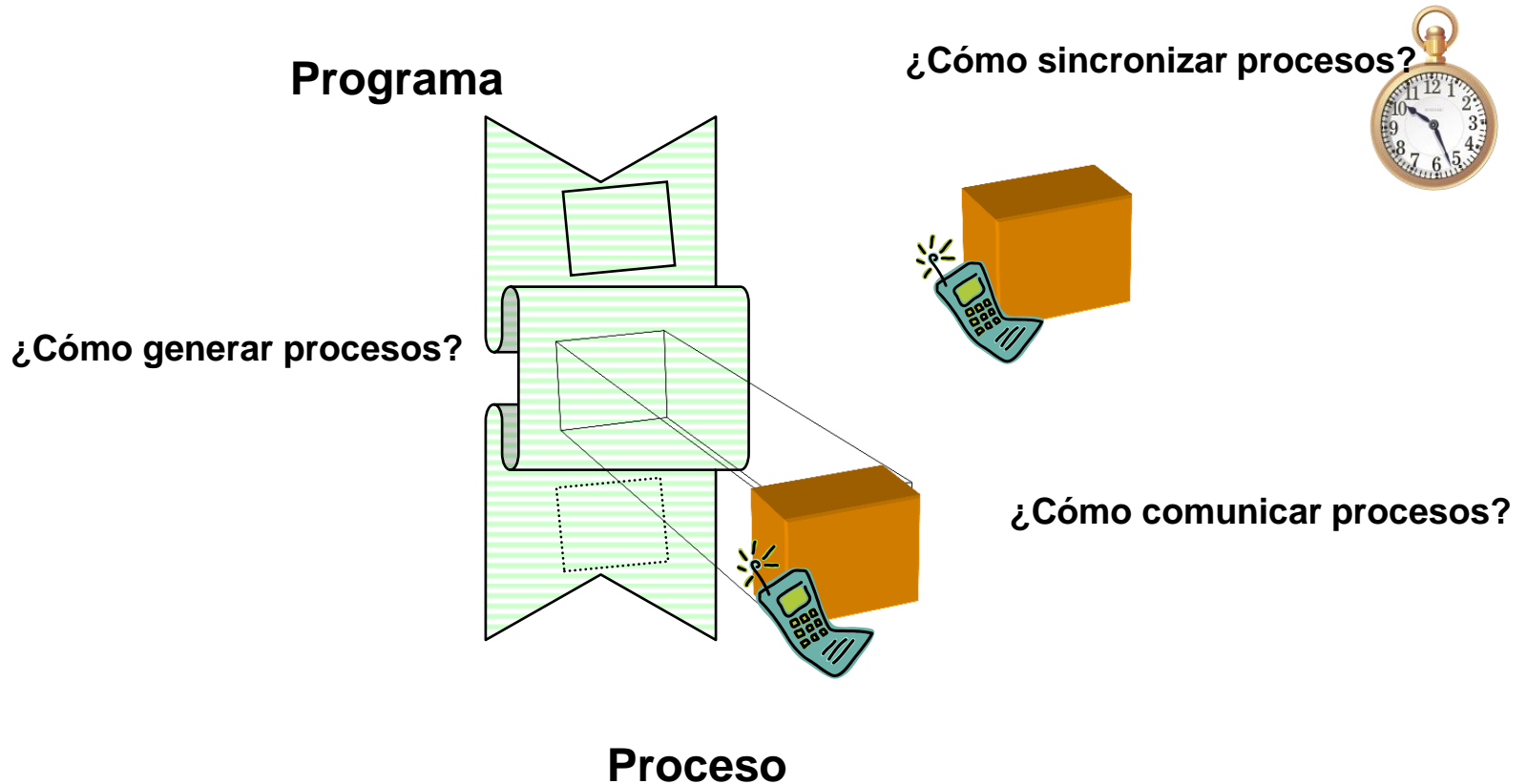


Tipos de concurrencia

- Multiprogramación
- Multitarea (tiempo compartido)
 - S.O. apropiativo (*preemptive*)
- Multiproceso
- Paralelismo



¿Cómo expresar la concurrencia?



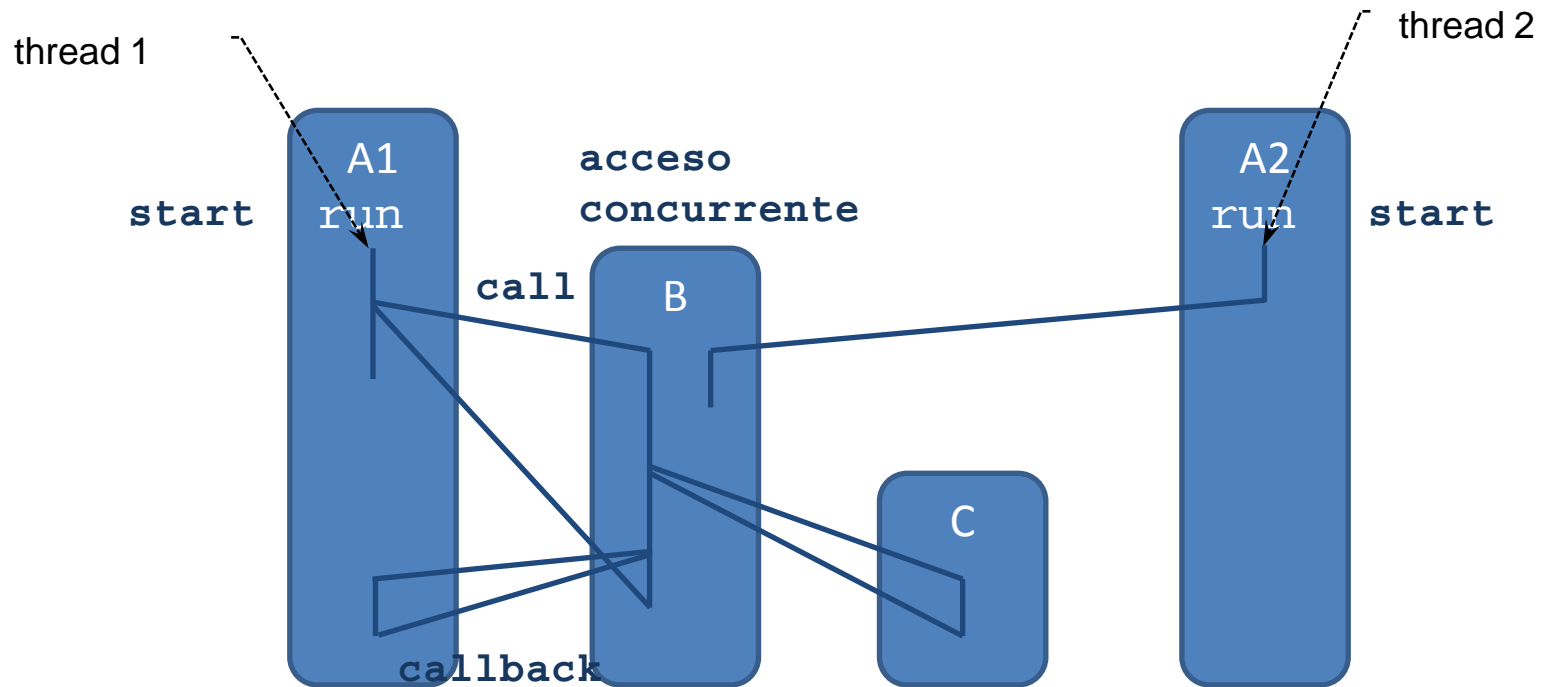
¿Cómo expresar la concurrencia?

- Incorporar construcciones específicas a un lenguaje
 - Compilador nuevo
 - Requiere entrenamiento
 - Puede no ser “natural”
- Directivas al compilador
 - Programas paralelos a partir de programas secuenciales
 - Restrictivo
- Usar un API
 - Solo se necesita disponer de la librería respectiva
 - Programación poco, o nada, estructurada

Concurrencia en Java

- Basada en threads
- Threads de lenguaje
- Creación de threads:
 - Los threads son representados por objetos
 - ... por ende existe una clase **Thread**
 - Los threads del programador son clases derivadas de **Thread**
 - La ejecución empieza en el método **run**
 - El **new** no activa al thread, solo crea al objeto
 - Para activar un thread (para que empiece a ejecutar su **run**) se invoca el método **start**
 - También se puede implementar la interfaz **Runnable**
 - Un thread no es un objeto: es un flujo de control sobre el código

Concurrencia en Java



Concurrencia en Java

Declaración

```
public class T extends Thread {  
    ...  
    public void run( ) {  
        acciones del thread  
    }  
}
```

Creación y activación

```
T t = new T( ); //Creación  
...  
t.start( ); //Activación
```

Concurrencia en Java

Generar n threads con identificación

```
public class T extends Thread {  
    private int id;  
  
    public T ( int n ) {  
        id = n;  
    }  
    ...  
    public void run( ) {  
        puede usar id  
    }  
    ...  
    for ( i = 0; i < nThreads; i++) {  
        new T( i ).start( );  
    }  
}
```

Concurrencia en Java

Buscar un valor en una matriz

```
public class T extends Thread {  
    private static int valor;  
    private static int n;  
    private static int [][] M;  
  
    private int id;  
  
    public T ( int i ) {  
        id = i;  
    }  
  
    private static void inicializar( ) {  
        inicializa valor, n, M  
    }  
}
```


Concurrencia en Java

Buscar un valor en una matriz

```
public void run () {  
    int nElementos = M[id].length;  
  
    for ( int j = 0; j < nElementos; j++) {  
        if ( M[id][j] == valor ) {  
            System.out.println( id );  
        }  
    }  
}  
  
public static void main(String[] args) {  
    inicializar( );  
    for ( int i = 0; i < n; i++)  
        new T( i ).start();  
}
```

Concurrencia en Java

Interfaz runnable

```
public class R implements Runnable{  
    ...  
    public void run( ) {  
        acciones del thread  
    }  
}
```

Creación y activación

```
R r = new R( ); //Creación del objeto  
Thread t = new Thread( r ); //Creación del thread  
...  
t.start( ); //Activación
```