6 ソフトウェア工学
Software Engineering

並行プログラミング

CONCURRENT PROGRAMMING

Concurrent Systems

1. 並行システム

プロセスとスレッド
並行システムの実行形態
割り込みとプロセスの切り替え
並行システムの利点

プロセスとスレッド

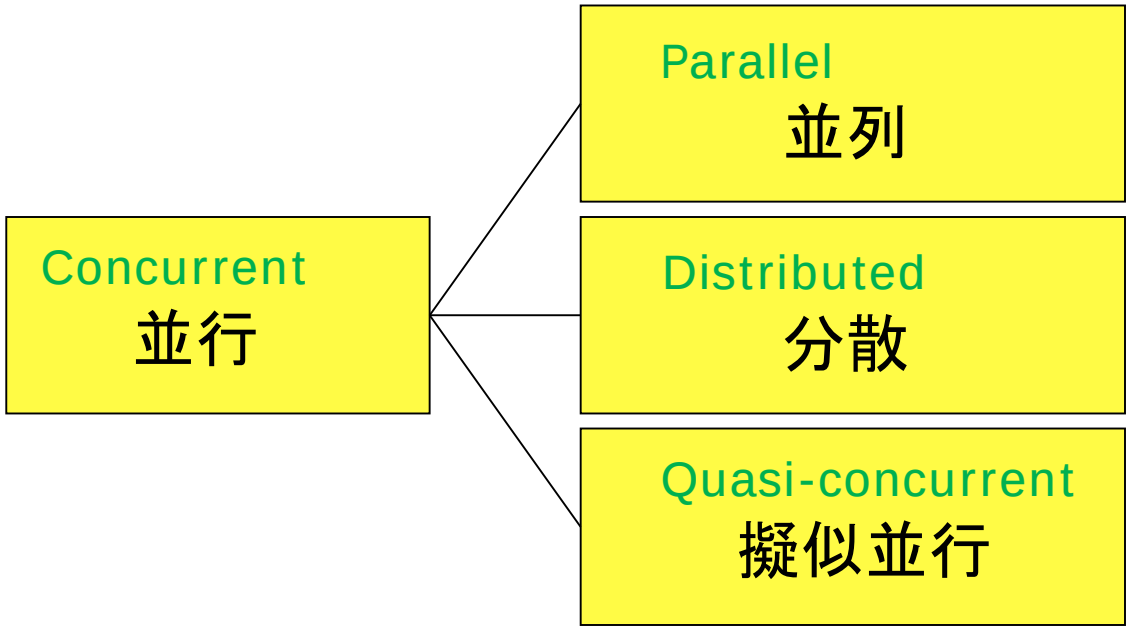
Process
- プロセス： 1つの逐次プログラムの実行単位

Multi-processing
- マルチプロセシング： 複数のプロセスを同時に実行すること

Thread
- スレッド： 1つのプロセス内の処理をさらに細分化した「軽量プロセス」とも呼ばれるプロセスの一種．複数のスレッドはメモリ空間を共有する．

Multi-thread
- マルチスレッド： 複数のスレッドを同時に実行すること．

並行システムの実行形態



物理的に結合して同時実行.

ハードウェア，ファームウェア

地理的に分散して同時実行.

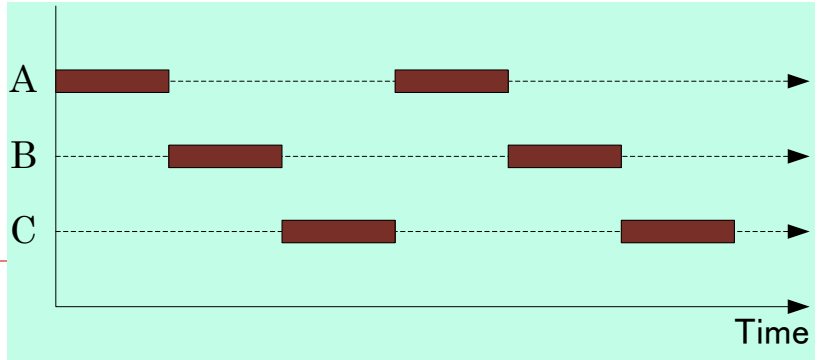
情報ネットワーク

1CPUで見かけ上同時実行.

OS

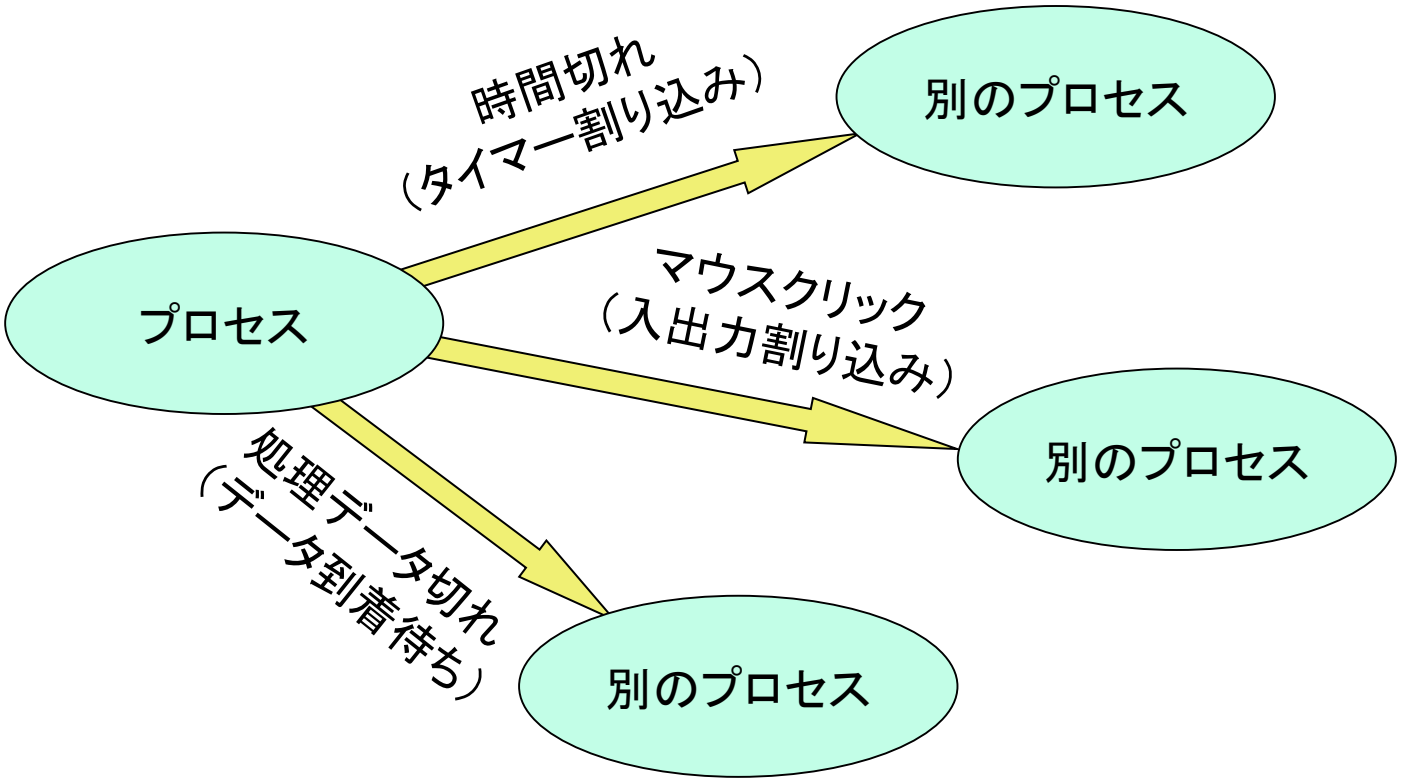
Interleaving

プロセスの切り替え
（インタリーブ）



Interruption

割り込みとプロセスの切り替え



並行システムの利点

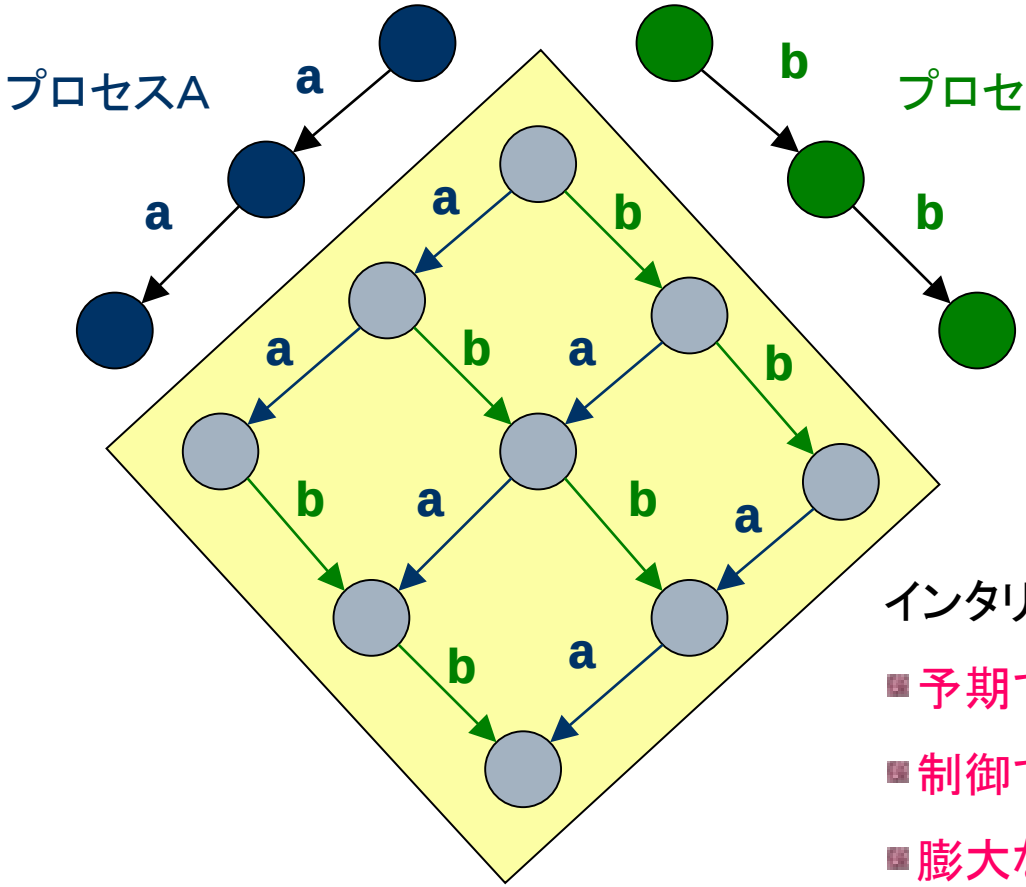
- 資源の有効利用
  - 入出力のような外部操作の待ち時間に，ほかのプログラムを動かすことができる.
- 公平性
  - 複数のユーザやプログラムが資源を平等に使う仕組みを作ることができる.
- 利便性
  - 必要なタスクを全部まとめたプログラムを書くよりも，各タスクごとに書くほうが良い.

２．並列プログラミングの注意点

インタリーブ
相互排他
オブジェクトのロックとスレッドの同期
デッドロック
飢餓，ライブロック

Interleaving

インタリーブ



インタリーブは

- 予期できない
- 制御できない
- 膨大な数の実行経路を生じる

Mutual exclusion

相互排他

共有変数は相互排他が必要

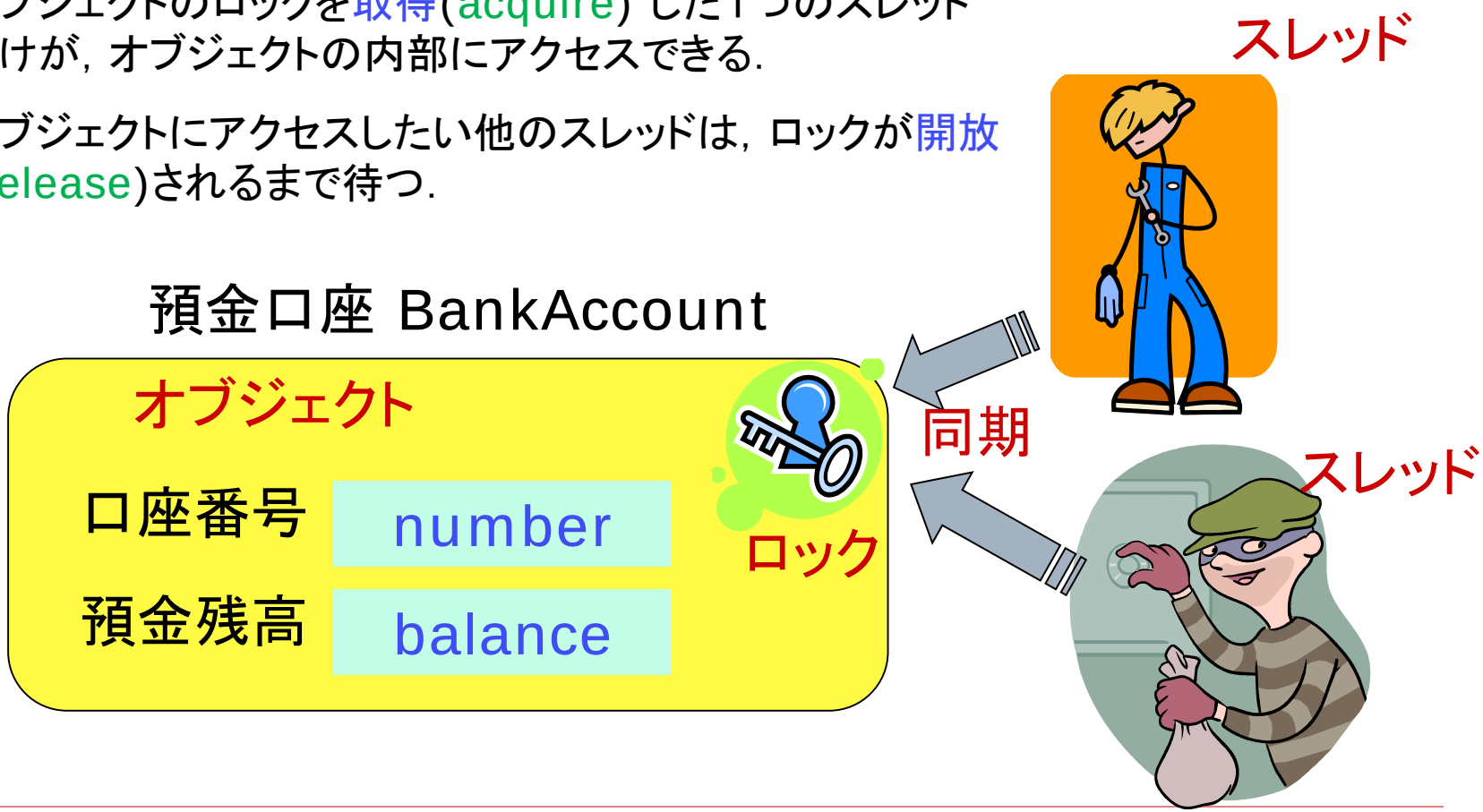
MOV A,m (A=5000)	1	
ADD A,1000 (A=6000)	2	
	3	MOV B,m (B=5000)
	4	SUB B,1000 (B=4000)
MOV m,A (m=6000)	5	
	6	MOV m,B (m=4000)

Lock　　Synchronization

オブジェクトのロックとスレッドの同期

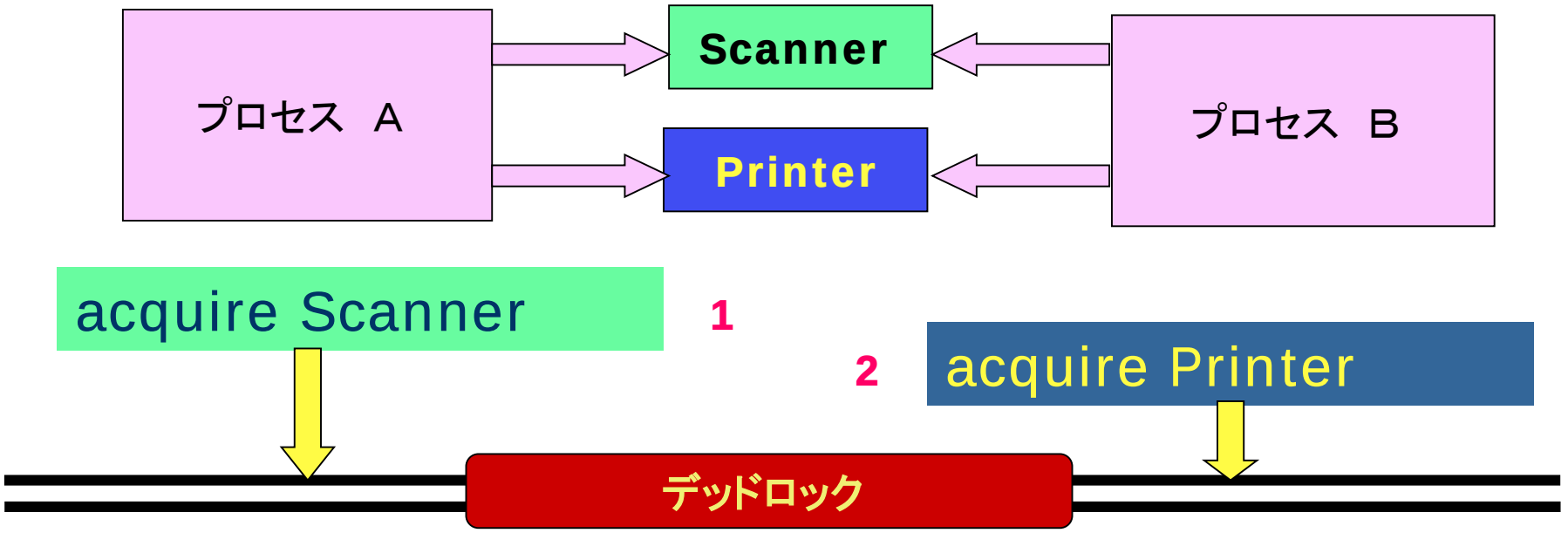
オブジェクトのロックを取得(acquire) した1つのスレッドだけが，オブジェクトの内部にアクセスできる．

オブジェクトにアクセスしたい他のスレッドは，ロックが開放(release)されるまで待つ．



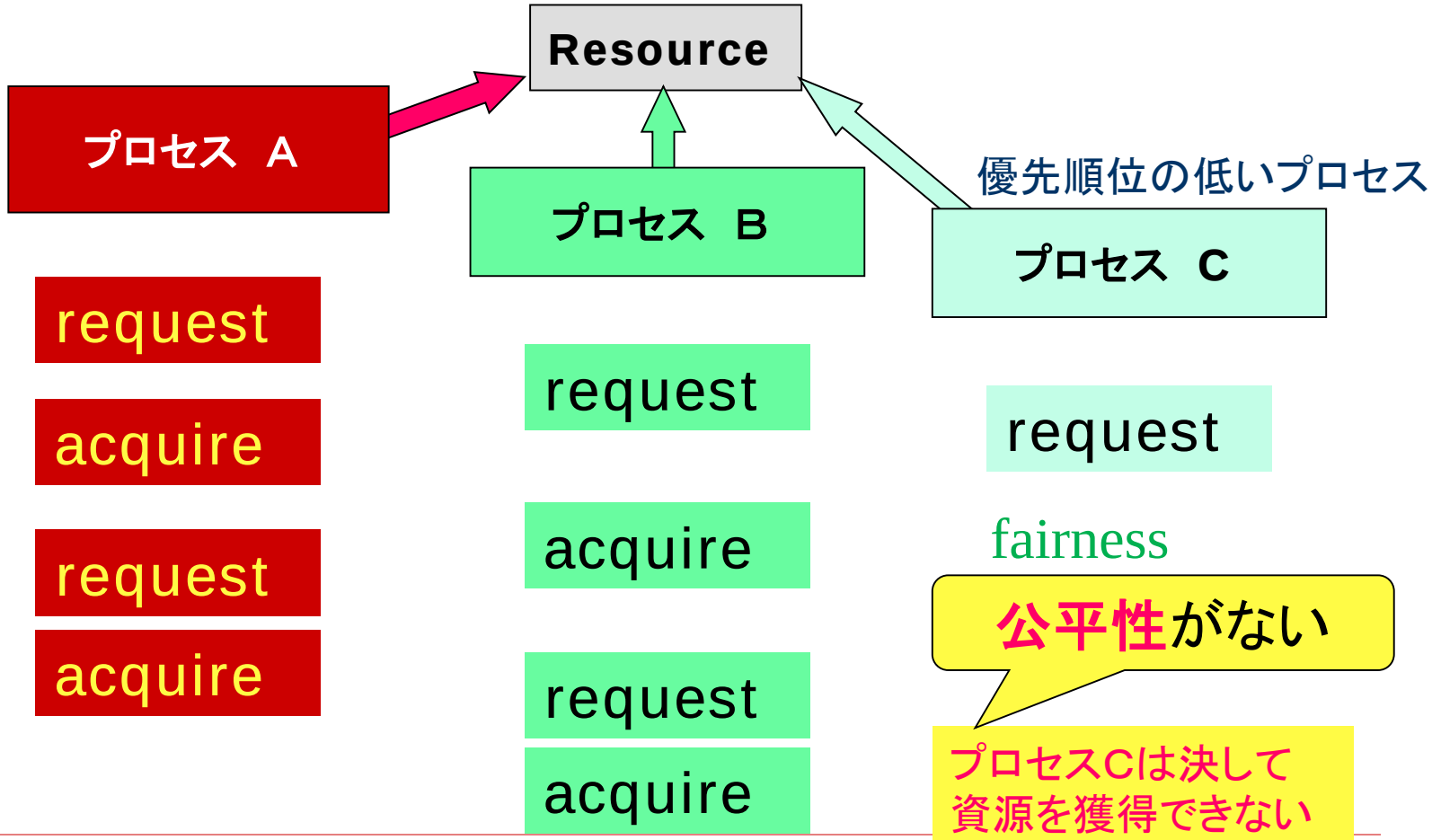
Deadlock

デッドロック



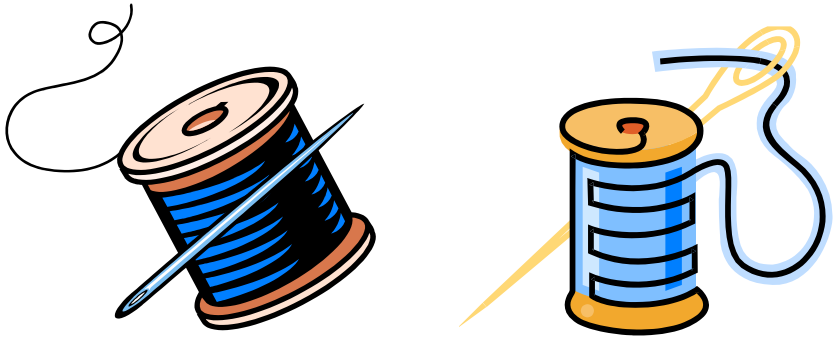
Starvation　Livelock

飢餓，ライブロック



3. Javaマルチスレッドプログラミング

- Threadオブジェクト
- run()メソッド
- スレッドの生成と実行



Threadオブジェクト

Threadオブジェクトを new で生成して，start()する．

```
Thread worker = new Thread();
worker.start();
```

スレッドを生成

スレッドを実行

しかし，これではプログラマが実行させたい動作を書けない．

run()メソッド

実行させたい動作は run()に書く.

```
public void run() {

……

}
```

スレッドの生成と実行(1/3)

Threadクラスを拡張したサブクラスにrun()を記述し，
そのサブクラスのインスタンスを生成

```
public class PingPongThread extends Thread {
    private String word; //what word to print
    private int delay;   //how long to pause

    public PingPong(String whatToSay, int delayTime) {
        word = whatTosay;
        delay = delayTime;
    }
```

コンストラクタ

スレッドの生成と実行(2/3)

Threadクラスを拡張したサブクラスにrun()を記述

```
    public void run() {
      try {
        for (;;) {
          System.out.print(word + " ");
          Thread.sleep(delay); //wait until next time
        }
      } catch (InterruptedException e) {
          return;                    //end this thread
    }
}
```

run()メソッド

スレッドの生成と実行(3/3)

```
  public static void main(String[] args) {
    PingPongThread t1 = new PingPongThread(”ping”, 33);
    PingPongThread t2 = new PingPongThread (”PONG”,100);
    t1.start();
    t2.start();
  }
}
```

実行結果

```
ping PONG ping ping PONG ping ping ping PONG ping …
```

演習問題 6

スキャンとプリントの機能をもつ複合機3台(M0,M1,M2)およびこれらを利用する3つのプログラム(P0,P1,P2)が図のようにリング上に結合されている並行システムを考える.

複合機はロックを用いて相互排他が実現されているので，1つの複合機に同時に複数のプログラムがアクセスすることはできない.

各プログラムはいずれも隣接した1つの複合機から画像をスキャンし，もう1つの隣接した複合機にそれをプリントするものである.

ただし，各プログラムが複合機にアクセスする順序が異なり，P0,P1,P2はそれぞれ，M0,M1,M2から画像をスキャンし，M1,M2,M0にプリントする.

このシステムの問題点を説明し，その解決策について考察しなさい.