Esercitazione #3 -- Corso di Sistemi Operativi

Sincronizzazione in Java

*(*modificatore *synchronized* e i metodi *wait/notify/notifyAll)*

Luca Gherardi e Patrizia Scandurra – a.a. 2012-13

1. **CokeMachine**. Usando il modificatore *synchronized* e i metodi *wait/notify/notifyAll* per la sincronizzazione, si realizzi in Java una soluzione al problema di prelevare lattine di coca-cola da una macchinetta e di rifornirla nel caso in cui rimanga vuota. Definire le classi per i thread utenti e il thread rifornitore. Definire, inoltre, la classe CokeMachine contenente le lattine di coca-cola (oggetti condivisi!) ed i metodi:
   - remove(…), eseguito dal generico utente per prelevare una lattina dalla macchinetta;
   - deposit(…), eseguito dal fornitore del servizio per caricare la macchinetta bel caso in cui rimane vuota.

   Si assuma che inizialmente la macchinetta è piena, e che un utente (*a scelta*: il primo a trovare la macchinetta vuota o l'utente che preleva l'ultima lattina) può segnalare al fornitore che la macchinetta è vuota. Lo scheletro della classe CokeMachine è riportato di seguito come piccolo aiuto.

```
class CokeMachine{
static final int N = 50; //Capacità della macchinetta
int count ; //Numero effettivo di lattine presenti nella macchinetta
…. <COMPLETARE>….
}
```

2. **Magazzino**. Dei *fornitori* consegnano periodicamente prodotti di un certo tipo ad un *magazzino*. Tali prodotti sono venduti a dei *clienti* su prenotazione. I fornitori ed i clienti (i thread!) sono in concorrenza per l'accesso al magazzino (risorsa condivisa!). Si supponga che il magazzino contenga *N tipi* di prodotti. Usando il modificatore *synchronized* e i metodi *wait/notify/notifyAll* per la sincronizzazione, si definisca in Java una classe che implementi l'interfaccia Magazzino riportata nel riquadro sotto, fornendo gli attributi necessari per rappresentare le quantità per ogni tipo di prodotto ed una definizione dei metodi.

```
public interface Magazzino {
    public final static int N = 5; //numero di tipi di materiale
    /** Metodo invocato dai thread cliente per effettuare una prenotazione di alcuni  prodotti; ad es.
     * ordine=[2,0,0,0,1] significa 2 quantità del prodotto di tipo 0 e 1 quantità del prodotto di tipo 4.
     * L'operazione è bloccante: se nel magazzino non è disponibile la quantità richiesta per un certo tipo di
     * prodotto il cliente rimane in attesa che venga rifornito per completare la prenotazione.
     */
    public abstract void compra(int ordine[]);

     /**
      * Metodo invocato dai thread fornitore per rifornire il magazzino di un prodotto di un certo tipo  e di
      * una certa quantità.
      */
     public abstract void produci(int tipo, int quantita);
}
```

3. **Lettori-scrittori senza starvation**. La soluzione dei lettori e scrittori presentata durante l'introduzione di questa esercitazione può portare a una situazione di starvation. Quale tipo di thread potrebbe essere indefinitamente ritardato? Modificare la soluzione in modo da risolvere il problema della starvation.