

**SISTEMI OPERATIVI IDT/IEL/IIN**  
**INFORMATICA INDUSTRIALE E SISTEMI OPERATIVI IDI**  
**SISTEMI DI ELABORAZIONE p.o.**  
**prova scritta del 07 settembre 2007**

Nome \_\_\_\_\_  
Cognome \_\_\_\_\_  
Matricola \_\_\_\_\_  
Corso di laurea \_\_\_\_\_

Un database è letto da N thread di tipo A e M thread di tipo B. I thread A hanno priorità=10, maggiore dei thread B (priorità=5).

Thread con uguale priorità possono accedere contemporaneamente al database. I thread a più bassa priorità possono accedere al database solo se nessun thread a priorità più elevata sta leggendo.

Ogni thread dopo aver completato K accessi al database cambia la propria priorità (i thread A passano da priorità 10 a priorità 5, ed i thread B da priorità 5 a priorità 10).

Si scriva il programma Java che gestisce la sincronizzazione tra i thread A e B per la lettura dal database.

## Soluzione

```
public class Database {
    private int readAccount;

    public Database() {
        readAccount = 0;
    }

    public synchronized void startRead( int priority ) {
        // se il thread è di tipo B e ci sono thread di tipo A attende
        while ( priority == 5 && readAccount >0 )
            try {
                wait();
            }
        if ( priority == 10 )
            readAccount ++;
    }

    public synchronized void endRead( int priority ) {
        if ( priority == 10 ) {
            readAccount --;
            if ( readAccount == 0 )
                notifyAll();
        }
    }

    public void read() {
        // lettura dal database
    }
}

public class Lettore extends Thread {
    private int max;        // massimo numero di volte in cui è ammesso l'accesso prima di
                           // cambiare priorità
    private int count;     // numero di volte che il thread ha acceduto il database
    private int priority;
    private Database db; // oggetto database condiviso tra tutti i thread

    public Lettore( Database db, int k, int p ) {
        count = 0;
        this.db = db;
        max = k;
        priority = p;
    }
}
```

```

public void run() {
    while ( true ) {
        db.startRead( priority );
        db.read();
        db.endRead( priority );
        count ++;
        if ( count == max ) {
            if ( priority == 10 )
                priority = 5;
            else
                priority = 10;
        }
    }
}

```

```

public class Principale {
    public void main( String argv[] ) {
        int N = 10;
        int M = 10;
        int K = 10;
        Database db = new Database();

        // thread A con priorità = 10
        for ( int i=0; i<N; i ++ ) {
            Lettore A = new Lettore( db, K, 10 );
            A.start();
        }

        // thread B con priorità = 5
        for ( int i=0; i<M; i ++ ) {
            Lettore B = new Lettore( db, K, 5 );
            B.start();
        }
    }
}

```