*Esame di **Sistemi Operativi** del Corso di Laurea in Ingegneria Informatica DM270*
*Prova scritta del **10 settembre 2013** – Durata: 2 ore e 45 minuti – Orario di consegna:*

Cognome e nome:	Matricola:

Prova 1

Si consideri la seguente successione di riferimenti a pagine in memoria centrale:

1, 2, 3, 4, 3, 5, 2, 6, 3, 7, 2, 7, 6, 2, 4

Calcolare quante **assenze di pagine** (page fault) si verificano se si usano 4 blocchi di memoria con i seguenti algoritmi di sostituzione:

1. **LRU**
2. **Ottimale**

Prova 2

Si descriva lo schema di traduzione degli indirizzi in un'architettura di paginazione a due livelli.

Prova 3

Un *boccaccio* contiene fino a **100** caramelle di **N** colori diversi. Alcuni *bambini* ciclicamente prendono una caramella dal boccaccio e la mangiano (impiegando dai 40 agli 80 secondi). Ogni bambino ha un colore preferito e prende dal boccaccio solo le caramelle di quel colore: se non trova alcuna caramella del colore preferito si mette a piangere. Ogni volta che **tre o più** bambini piangono, un *addetto* riempie il boccaccio fino ad esaurire la sua capienza, con un numero casuale di caramelle per ciascun colore, ma avendo cura che nel barattolo ce ne siano almeno tre per ciascun colore.

Si modelli il sistema descritto in Java, dove i **bambini** e l'**addetto** sono dei thread che interagiscono tramite un oggetto **boccaccio**: tale oggetto espone tre metodi, (1) `prendi(int c)`, (2) `piangi()` che permettono ai bambini rispettivamente di prendere una caramella di colore *c* e di piangere nel caso in cui le caramelle del colore preferito siano terminate, e (3) `riempi()`, che permette all'addetto di riempire il boccaccio. Il metodo (1) restituisce un booleano: `true` se il bambino è riuscito a prendere una caramella, `false` altrimenti.

Si implementino due soluzioni che riproducano il funzionamento del problema sopra descritto utilizzando:

- la classe `Semaphore` del package `java.util.concurrent`
- gli strumenti di mutua esclusione e sincronizzazione del package `java.util.concurrent.locks`

Si scriva infine un main d'esempio che, facendo uso di una delle due soluzioni precedenti, inizializzi un **boccaccio** con **N**=3, inizializzi 20 **bambini**, 1 **addetto** e ne avvii l'esecuzione.