DIMES UNIVERSITÀ DELLA CALABRIA
Dipartimento di Ingegneria Informatica, Modellistica, Elettronica e Sistemistica

Esame di **Sistemi Operativi** del Corso di Laurea in Ingegneria Informatica DM270

Prova scritta del 17 luglio 2014 - Durata: 2 ore e 45 minuti - Orario di consegna: _____

Cognome e nome:	Matricola:

Prova 1

Si supponga di disporre di un disco composto da 2000 cilindri numerati da 0 a 1999. Il disco sta servendo una richiesta relativa al cilindro 500 e la richiesta precedente era relativa al cilindro 700, la coda di richieste inevase in ordine FIFO è composta delle seguenti richieste:

800, 200, 100, 600, 1400, 1200, 300, 1000, 1800

assumendo come punto di partenza la posizione attuale della testina, calcolare la distanza totale (in cilindri) che il braccio del disco percorre per soddisfare tutte le richieste inevase usando i seguenti algoritmi di scheduling:

1. SSTF;
2. SCAN.

Prova 2

Si descriva, anche mediante esempi, che cosa sono le system call in un sistema operativo.

Prova 3

Si vuole realizzare un sistema per gestire una gara di sci. Alla gara partecipano N sciatori, ognuno dei quali è dotato di un numero di maglia che va da 0 a N-1. Ogni sciatore viene fatto partire ogni 20 secondi da un addetto alla gara, in ordine di numero di maglia, ed impiega dai 50.000 ai 70.000 millisecondi per arrivare al traguardo: una volta tagliato il traguardo, il thread che modella lo sciatore stampa su schermo la propria posizione temporanea in classifica (ad esempio "3°"). Dopo che l'ultimo sciatore ha tagliato il traguardo, l'addetto alla gara stamperà su schermo la classifica finale, ovvero i numeri di maglia degli sciatori ordinati da quello che ha impiegato meno tempo a quello che ne ha impiegato di più (ad esempio "43 91 60 9 17...").

Si modelli il sistema descritto in Java, dove gli sciatori e l'addetto sono dei thread che interagiscono tramite un oggetto *gara* che espone solo i seguenti metodi:

- **`void partenza(Sciatore s)`**: sospende lo sciatore *s* fin quando non è il suo turno;
- **`int arrivo(Sciatore s)`**: permette allo sciatore *s* di tagliare il traguardo e, contrariamente a quanto succede nella realtà, di dichiarare il tempo (in millisecondi) che ha impiegato per tagliare il traguardo. Restituisce la posizione temporanea in classifica dello sciatore *s*;
- **`boolean prossimo()`**: permette all'addetto di far partire il prossimo sciatore. Quando gli sciatori sono terminati stampa la classifica finale e restituisce false.

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

- la classe `Semaphore` del package `java.util.concurrent` (per consentire la partenza in ordine di numero di maglia si consiglia di usare un array di N semafori)
- 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 oggetto gara, inizializzi 100 sciatori e l'addetto, e ne avvii l'esecuzione.