Note sullo svolgimento della prova

- Non è consentito l'uso di vecchi testi d'esame, libri, o appunti.
- Utilizzare NetBeans per editare, compilare ed eseguire i programmi.
- Creare un file **Cognome_matricola.zip** contenente i soli file *.java* da consegnare.
- Al termine della prova, dopo il caricamento degli elaborati, il docente mostrerà una possibile soluzione. Dopo aver visto la soluzione, gli studenti avranno la possibilità di riprendere il proprio elaborato.

Esercizio

I clienti di un negozio sono divisi in due categorie: vip e normali. Il negozio può contenere un massimo di N clienti di cui al più M possono essere normali, con M minore di N. Quando si raggiunge il numero massimo N, non viene permesso nessun nuovo ingresso finché tutti i clienti presenti nel negozio non sono usciti. Se ci sono clienti normali e vip in attesa di entrare, i clienti vip hanno la precedenza su quelli normali.

Realizzare una classe *Negozio* con almeno i seguenti metodi:

- *Negozio(int N, int M)*: costruisce un oggetto *Negozio* dalle caratteristiche specificate; lancia *ParametriErratiException* se i parametri non sono validi;
- *void entrata(boolean vip)*: un cliente chiede di entrare nel negozio specificando se vip o meno; l'operazione può essere bloccante;
- *void uscita(boolean vip)*: un cliente notifica la propria uscita dal negozio.

Realizzare una classe *Cliente* che si comporta come segue: entra nel negozio, rimane nel negozio un tempo casuale compreso tra 1 e 2 secondi, esce dal negozio.

Realizzare una classe di prova che crea 10 clienti, di cui 5 vip e 5 no, e un negozio con M=3 e N=2. Esempio di output:

```
Cliente Thread-0(VIP) entrato
Cliente Thread-9 entrato
Cliente Thread-8(VIP) entrato
Negozio chiuso
Cliente Thread-7 in attesa
Cliente Thread-6(VIP) in attesa
Cliente Thread-5 in attesa
Cliente Thread-4(VIP) in attesa
Cliente Thread-3 in attesa
Cliente Thread-2(VIP) in attesa
Cliente Thread-1 in attesa
Cliente Thread-9 uscito
Cliente Thread-7 in attesa
Cliente Thread-1 in attesa
Cliente Thread-2(VIP) in attesa
Cliente Thread-3 in attesa
Cliente Thread-4(VIP) in attesa
Cliente Thread-5 in attesa
Cliente Thread-6(VIP) in attesa
Cliente Thread-8(VIP) uscito
Cliente Thread-7 in attesa
Cliente Thread-6(VIP) in attesa
Cliente Thread-5 in attesa
Cliente Thread-4(VIP) in attesa
Cliente Thread-3 in attesa
Cliente Thread-2(VIP) in attesa
Cliente Thread-1 in attesa
Cliente Thread-0(VIP) uscito
Negozio riaperto
Cliente Thread-7 in attesa
Cliente Thread-1 in attesa
Cliente Thread-2(VIP) entrato
Cliente Thread-3 in attesa
Cliente Thread-4(VIP) entrato
Cliente Thread-5 in attesa
Cliente Thread-6(VIP) entrato
Negozio chiuso
```

```
Cliente Thread-6(VIP) uscito
Cliente Thread-7 in attesa
Cliente Thread-5 in attesa
Cliente Thread-3 in attesa
Cliente Thread-1 in attesa
Cliente Thread-2(VIP) uscito
Cliente Thread-7 in attesa
Cliente Thread-1 in attesa
Cliente Thread-3 in attesa
Cliente Thread-5 in attesa
Cliente Thread-4(VIP) uscito
Negozio riaperto
Cliente Thread-7 entrato
Cliente Thread-5 entrato
Cliente Thread-3 in attesa
Cliente Thread-1 in attesa
Cliente Thread-5 uscito
Cliente Thread-3 entrato
Cliente Thread-1 in attesa
Cliente Thread-7 uscito
Cliente Thread-1 entrato
Cliente Thread-1 uscito
Cliente Thread-3 uscito
```