

Il tipo di dato astratto `Barriera` implementa le funzionalità di una barriera autostradale. Una barriera è composta da 5 caselli, numerati a partire da 1. Un casello può essere aperto o chiuso e i veicoli che sopraggiungono si accodano a uno dei caselli aperti. Un veicolo viene univocamente identificato tramite la propria targa, ovvero una sequenza alfanumerica del tipo `YYYYYY`, dove `Y` è una lettera compresa tra 'A' e 'Z'. Implementare le seguenti operazioni che possono essere fatte su `Barriera`:

--- **PRIMA PARTE** --- (qualora siano presenti errori di compilazione, collegamento o esecuzione in questa parte, l'intera prova sarà considerata insufficiente e pertanto non sarà corretta)

✓ **`Barriera b;`**

Costruttore che inizializza una barriera. Inizialmente, dei cinque caselli solo i primi 3 caselli sono aperti e non hanno veicoli in coda, mentre i restanti due sono chiusi.

✓ **`b.nuovoVeicolo(s);`**

Operazione che implementa l'arrivo di un nuovo veicolo con targa `s`. Il veicolo si accoda al casello aperto avente il minor numero di veicoli in coda. Se la targa `s` non ha il formato corretto, la struttura rimane inalterata. Se più caselli hanno lo stesso minor numero di veicoli, viene selezionato il casello con numero d'ordine più basso. **Implementare la funzione in modo che NON debba contare i veicoli in coda ogni volta che la funzione viene invocata.**

✓ **`b.serviVeicolo(i);`**

Operazione che rimuove dalla barriera il primo veicolo in coda sul casello `i`. Se il casello `i` è chiuso o non ha veicoli in coda, la struttura rimane inalterata, e la funzione restituisce `false`. Altrimenti la funzione restituisce `true`.

✓ **`cout << b;`**

Operatore di uscita per il tipo `Barriera`. L'uscita ha il seguente formato:

```
[1] (chiuso)
[2] CCCDDD=>IIIJJJ
[3] (libero)
[4] (chiuso)
[5] (chiuso)
```

L'output corrisponde a una barriera in cui il secondo ed il terzo casello sono aperti, e di questi solo il secondo ha veicoli in coda (due, per la precisione).

--- **SECONDA PARTE** ---

✓ **`b.apriOppureChiudi(mediaIdeale);`**

Operazione che calcola la lunghezza media delle code dei soli caselli aperti. Se la media è minore del numero reale `mediaIdeale`, viene chiuso, se esiste, il casello con numero d'ordine minore **fra quelli aperti e senza auto in coda**. Se invece la media è maggiore della `mediaIdeale`, viene aperto un nuovo casello, qualora ne sia disponibile uno: il casello da aprire, in questo caso, è quello con numero d'ordine più alto fra quelli chiusi. La funzione restituisce 1 o -1 a seconda che sia stato aperto o chiuso un casello. La funzione restituisce 0 se il numero dei caselli rimane inalterato.

✓ **`int(b);`**

Operazione che restituisce il numero totale di veicoli in coda alla barriera. **Implementare la funzione in modo che NON debba contare i veicoli in coda ogni volta che la funzione viene invocata.**

✓ **`~Barriera();`**

Distruttore.

Mediante il Linguaggio C++, realizzare il tipo di dato astratto `Barriera`, definito dalle precedenti specifiche. **Gestire le eventuali situazioni di errore.**

USCITA CHE DEVE PRODURRE IL PROGRAMMA

--- PRIMA PARTE ---

[1] (libero)
[2] (libero)
[3] (libero)
[4] (chiuso)
[5] (chiuso)

[1] AAABBB=>GGGHHH
[2] CCCDDD=>IIIJJJ
[3] EEEFFF
[4] (chiuso)
[5] (chiuso)

[1] (libero)
[2] CCCDDD=>IIIJJJ
[3] (libero)
[4] (chiuso)
[5] (chiuso)

--- SECONDA PARTE ---

apriOppureChiudi: -1 caselli aperti

[1] (chiuso)
[2] CCCDDD=>IIIJJJ
[3] (libero)
[4] (chiuso)
[5] (chiuso)

[1] (chiuso)
[2] CCCDDD=>IIIJJJ=>OOOPPP
[3] KKKLLL=>MMNNN
[4] (chiuso)
[5] (chiuso)

apriOppureChiudi: 1 caselli aperti

[1] (chiuso)
[2] CCCDDD=>IIIJJJ=>OOOPPP
[3] KKKLLL=>MMNNN
[4] (chiuso)
[5] (libero)

Sono presenti 5 veicoli alla barriera

Test del distruttore sull'oggetto b2

[1] QQRRR
[2] (libero)
[3] (libero)
[4] (chiuso)
[5] (chiuso)

(b2 e' stato distrutto)