

Simulazione di un Parcheggio

Obiettivo

Realizzare un'applicazione console in **C#** utilizzando **Visual Studio 2012** che simuli il funzionamento di un parcheggio a capienza limitata, gestendo l'accesso concorrente di più auto attraverso meccanismi di sincronizzazione tra thread.

Contesto

Si vuole modellare un parcheggio con un numero massimo di posti disponibili. Numerose auto cercano contemporaneamente di entrare nel parcheggio, ma solo un numero limitato può sostarvi nello stesso momento. Le auto che non trovano posto devono attendere che si liberi uno spazio. Una volta entrate, sostano per un tempo variabile, dopodiché escono liberando il posto per un'altra auto.

Requisiti Funzionali

L'applicazione deve:

1. Creare un parcheggio con una **capienza massima configurabile** (ad esempio 10 posti).
2. Simulare l'arrivo di **almeno 100 auto** che tentano contemporaneamente di entrare nel parcheggio.
3. Garantire che **non venga mai superato il numero massimo** di posti disponibili.
4. Gestire correttamente la **coda di attesa** delle auto quando il parcheggio è pieno.
5. Simulare un **tempo di sosta variabile** per ogni auto (ad esempio tra 200 e 5000 millisecondi).
6. Loggare a console, con colori differenti, i seguenti eventi:
 - **Verde**: ingresso di un'auto, con timestamp, targa e posti occupati
 - **Rosso**: uscita di un'auto, con timestamp, targa e posti occupati
 - **Giallo**: eventuali errori durante le operazioni

Requisiti Tecnici

L'applicazione deve essere strutturata come segue:

Classe Auto

Rappresenta un veicolo. Deve contenere almeno la proprietà `Targa` (stringa). La targa deve essere generata casualmente seguendo il formato italiano: **2 lettere + 3 numeri + 2 lettere** (es. AB123CD).

Classe Parcheggio

Gestisce la logica del parcheggio. Deve includere:

- Un **SemaphoreSlim** per limitare il numero di auto contemporaneamente presenti
- Una **collezione** (es. `List<Auto>`) per tracciare le auto attualmente parcheggiate
- Un **oggetto di lock** per sincronizzare l'accesso alla collezione e alle stampe a console
- Una proprietà `CapienzaMassima` in sola lettura dall'esterno
- Il metodo `EntraAuto(Auto auto)` che gestisce l'ingresso, l'attesa, la sosta e richiama l'uscita

- Il metodo `EsciAuto(Auto auto)` che gestisce l'uscita e rilascia il posto sul semaforo
- Metodi privati di logging per ingresso, uscita ed errori

Classe `Program`

Contiene il `Main` e deve:

- Istanziare un parcheggio con capienza definita
- Creare un array di **almeno 100 Task** che simulano l'ingresso delle auto
- Avviare tutti i task e attendere il loro completamento con `Task.WaitAll`
- Implementare il metodo `GeneraTarga()` per la generazione casuale delle targhe
- Implementare il metodo `SimulaIngressoAuto(Parcheggio parcheggio)` che crea una nuova auto e ne tenta l'ingresso

Vincoli di Sincronizzazione

- L'accesso ai posti del parcheggio deve essere regolato tramite **`SemaphoreSlim`** inizializzato con valore iniziale e massimo pari alla capienza
- L'accesso alla lista delle auto parcheggiate e alle operazioni di logging deve essere protetto tramite **`lock`** su un oggetto dedicato
- Il rilascio del semaforo (`Release`) deve avvenire **solo dopo** la rimozione effettiva dell'auto dalla lista
- Utilizzare il blocco `try/catch/finally` per garantire la corretta gestione delle eccezioni e l'uscita dell'auto dal parcheggio

Output Atteso

A console deve essere visualizzata una sequenza di eventi simile alla seguente:

```
[04/05/2026 10:15:32] Auto AB123CD ENTRATA. Posti occupati: 1/10
[04/05/2026 10:15:32] Auto EF456GH ENTRATA. Posti occupati: 2/10
...
[04/05/2026 10:15:34] Auto AB123CD USCITA. Posti occupati: 9/10
...
Simulazione del parcheggio completata.
```

In ogni momento il numero di posti occupati non deve mai superare la capienza massima.