CONCURRENTDICTIONARY

Represents a thread-safe collection of key/value pairs that can be accessed by multiple threads concurrently.

```
private  ConcurrentDictionary<int, string> dictionary = new();
```

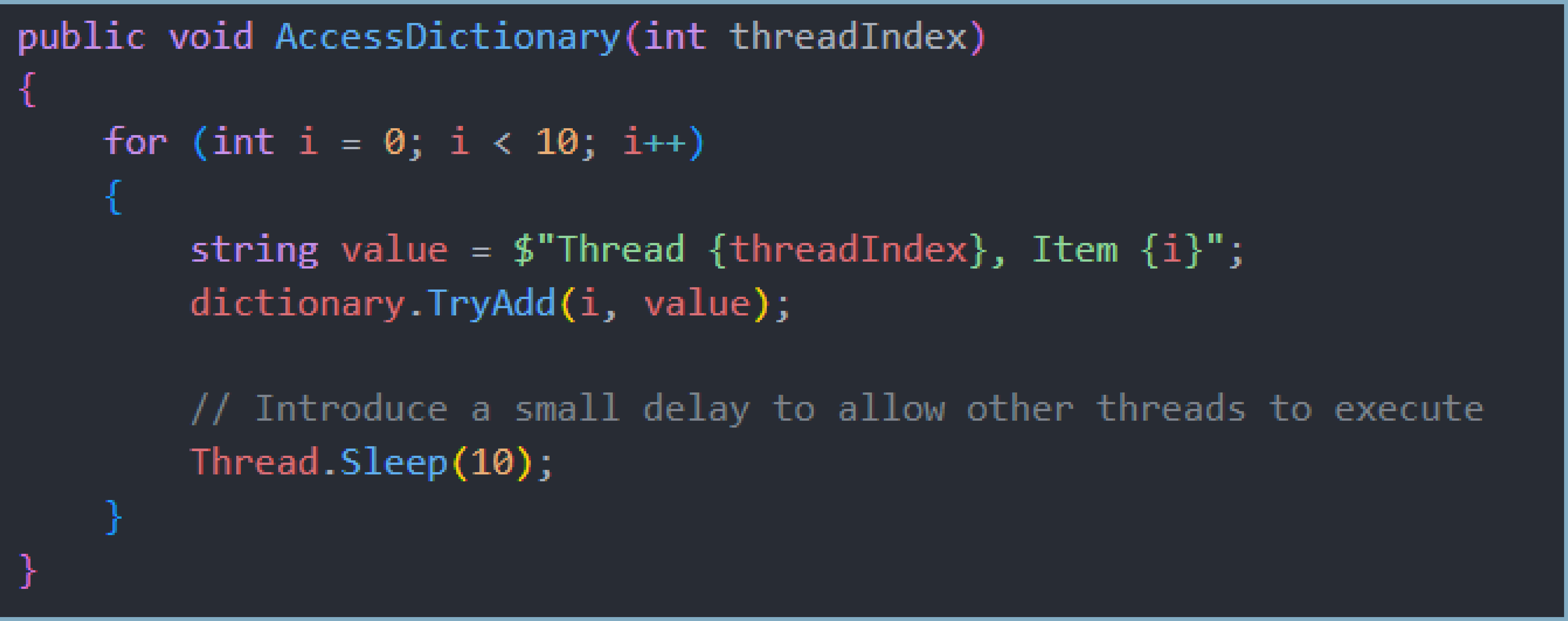
```
public void run()
{
    // Start multiple threads to access the ConcurrentDictionary concurrently
    Task[] tasks = new Task[5];
    for (int i = 0; i < tasks.Length; i++)
    {
        int index = i;
        tasks[i] = Task.Run(() => AccessDictionary(index));
    }

    // Wait for all threads to complete
    Task.WaitAll(tasks);

    // Print the contents of the ConcurrentDictionary
    foreach (var kvp in dictionary)
    {
        Console.WriteLine($"{kvp.Key}: {kvp.Value}");
    }
}
```

@romain-od

www.code-review.tech

ACCESSDICTIONNARY

Within this method, each thread adds ten items to the ConcurrentDictionary, using the current thread index and item index as the key and value, respectively. We introduce a small delay using Thread.Sleep() to allow other threads to execute concurrently.



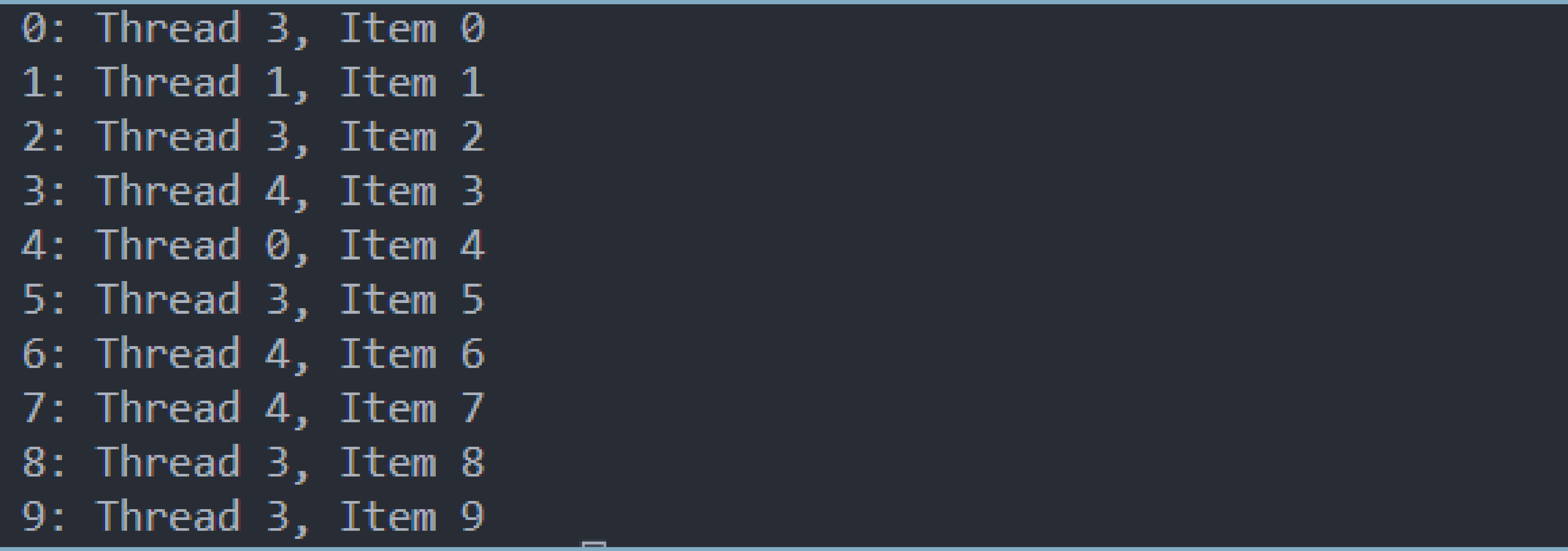
```
public void AccessDictionary(int threadIndex)
{
    for (int i = 0; i < 10; i++)
    {
        string value = $"Thread {threadIndex}, Item {i}";
        dictionary.TryAdd(i, value);

        // Introduce a small delay to allow other threads to execute
        Thread.Sleep(10);
    }
}
```

@romain-od

www.code-review.tech

OUTPUT

Since ConcurrentDictionary ensures thread safety, we can safely access and modify it concurrently without additional synchronization.



```
0: Thread 3, Item 0
1: Thread 1, Item 1
2: Thread 3, Item 2
3: Thread 4, Item 3
4: Thread 0, Item 4
5: Thread 3, Item 5
6: Thread 4, Item 6
7: Thread 4, Item 7
8: Thread 3, Item 8
9: Thread 3, Item 9
```

@romain-od

www.code-review.tech

THANKS FOR READY

SHARE YOUR THOUGHTS

FOLLOW ME FOR MORE SHARING

♡ 🔖 🔔 💬 ➤

@romain-od

www.code-review.tech