

Nội dung

Tạo lập tiến trình, tiểu trình, kiểm soát trạng thái của tiến trình, tiểu trình và Lập trình đa luồng MultiThread
Ngôn ngữ: C#

Hướng dẫn và yêu cầu

Phần 1: Tạo Lập tiểu trình mới

- 1) Tạo một dự án có chứa lớp sau:

```
class Program
{
    private int iterations;
    private string message;
    private int delay;
    public Program(int iterations, string message, int delay)
    {
        this.iterations = iterations;
        this.message = message;
        this.delay = delay;
    }
    public void Start()
    {
        // Tạo một thể hiện ủy nhiệm ThreadStart tham chiếu đến DisplayMessage.
        ThreadStart method = new ThreadStart(DisplayMessage);
        // Tạo một đối tượng Thread và truyền thể hiện ủy nhiệm ThreadStart cho phương thức tạo
        Thread thread = new Thread(method);
        Console.WriteLine("{0} : Starting new thread.", DateTime.Now.ToString("HH:mm:ss.ffff"));
        // Khởi chạy tiểu trình mới.
        thread.Start();
    }
    private void DisplayMessage()
    {
        // Hiện thông báo ra cửa sổ Console với iterations lần, nghỉ giữa mỗi thông báo
        // một khoảng thời gian được chỉ định (delay).
        for (int count = 0; count < iterations; count++)
        {
            Console.WriteLine("{0} : {1}", DateTime.Now.ToString("HH:mm:ss.ffff"), message);
            Thread.Sleep(delay);
        }
    }
    static void Main(string[] args)
    {
        // Tạo một đối tượng ThreadExample.
        Program example = new Program(5, "A thread example.", 500);
        // Khởi chạy đối tượng ThreadExample.
        example.Start();
        // Tiếp tục thực hiện công việc khác.
```

```

        for (int count = 0; count < 13; count++)
        {
            Console.WriteLine("{0} : Continue processing...",
                DateTime.Now.ToString("HH:mm:ss.ffff"));
            Thread.Sleep(200);
        }
        // Nhấn Enter để kết thúc.
        Console.WriteLine("Main method complete. Press Enter.");
        Console.ReadLine();
    }
}

```

- 2) Chạy chương trình trên (mục 4) và phân tích kết quả.
- 3) Phân tích ý nghĩa của từng dòng lệnh và ý nghĩa của cả chương trình trên.
- 4) Sử dụng chương trình đã cho trong mục 4 để xây dựng một chương trình cho phép 2 tiểu trình chạy song song, trong đó:
 - Sử dụng biến count dùng chung giữa hai tiểu trình
 - TiểuTrìnhCong: Tăng liên tục biến count lên 2500 lần.
 - TiểuTrìnhTru: Giảm liên tục biến count xuống 2500 lần.
- 5) Nhận xét về chương trình đã viết ở trên.

Phần 2: Lập trình điều khiển trạng thái của tiểu trình

- 6) Xem lại sơ đồ chuyển đổi trạng thái của tiến trình, tiểu trình đã học.
- 7) Trong Java hoặc C#, sử dụng các phương thức sau của lớp Thread để làm thay đổi trạng thái của tiểu trình:
 - Abort: kết thúc một tiểu trình thông qua ngoại lệ System.Threading.ThreadAbortException.
 - Interrupt: tiểu trình gọi Sleep hay đợi tài nguyên. Thông qua ngoại lệ System.Threading.ThreadInterruptedException.
 - Resume: Phục hồi quá trình thực thi của một tiểu trình đã bị tạm hoãn (bị block).
 - Suspend: Tạm hoãn quá trình thực thi của một tiểu trình cho đến khi phương thức Resume được gọi.
 - Start: Khởi chạy tiểu trình mới.
- 8) Tạo dự án mới có chứa lớp sau:

```

class Program
{
    public static void DisplayMessage1()
    {
        // Lặp đi lặp lại việc hiển thị một thông báo ra cửa sổ Console.
        while (true)
        {
            try
            {
                Console.WriteLine("{0} : Second thread running. Enter" + " (S)uspend, (R)esume, (I)nterrupt, or (E)xit.", DateTime.Now.ToString("HH:mm:ss.ffff"));
                // Nghỉ 2 giây.
                Thread.Sleep(2000);
            }
            catch (ThreadInterruptedException)

```

```

    {
        // Tiểu trình đã bị gián đoạn. Việc bắt ngoại lệ ThreadInterruptedException cho phép ví dụ này
        // thực hiện hành động phù hợp và tiếp tục thực thi.
        Console.WriteLine("{0} : Second thread interrupted.",
            DateTime.Now.ToString("HH:mm:ss.ffff"));
    }
    catch (ThreadAbortException abortEx)
    {
        // Đối tượng trong thuộc tính ThreadAbortException.ExceptionState được cung cấp
        // bởi tiểu trình đã gọi Thread.Abort.
        // Trong trường hợp này, nó chứa một chuỗi mô tả lý do của việc hủy bỏ.
        Console.WriteLine("{0} : Second thread aborted ({1})",
            DateTime.Now.ToString("HH:mm:ss.ffff"), abortEx.ExceptionState);
    }
}
static void Main(string[] args)
{
    // Tạo một đối tượng Thread và truyền cho nó một thể hiện ủy nhiệm ThreadStart tham chiếu
    // đến DisplayMessage.
    Thread thread = new Thread(new ThreadStart(DisplayMessage1));
    Console.WriteLine("{0} : Starting second thread.", DateTime.Now.ToString("HH:mm:ss.ffff"));
    // Khởi chạy tiểu trình thứ hai.
    thread.Start();
    // Lặp và xử lý lệnh do người dùng nhập.
    char command = ' ';
    do {
        string input = Console.ReadLine();
        if (input.Length > 0)
            command = input.ToUpper()[0];
        else command = ' ';
        switch (command)
        {
            case 'S': // Tạm hoãn tiểu trình thứ hai.
                Console.WriteLine("{0} : Suspending second thread.",
                    DateTime.Now.ToString("HH:mm:ss.ffff"));
                thread.Suspend();
                break;
            case 'R': // Phục hồi tiểu trình thứ hai.
                try
                {
                    Console.WriteLine("{0} : Resuming second thread.",
                        DateTime.Now.ToString("HH:mm:ss.ffff"));
                    thread.Resume();
                }
                catch (ThreadStateException)
                {
                    Console.WriteLine("{0} : Thread wasn't suspended.",
                        DateTime.Now.ToString("HH:mm:ss.ffff"));
                }
                break;
            case 'T': // Gián đoạn tiểu trình thứ hai.
                Console.WriteLine("{0} : Interrupting second thread.",

```

```

        DateTime.Now.ToString("HH:mm:ss.ffff"));
        thread.Interrupt();
        break;
    case 'E':
        // Hủy bỏ tiểu trình thứ hai và truyền một đối tượng trạng thái cho tiểu trình đang bị hủy,
        // trong trường hợp này là một thông báo.
        Console.WriteLine("{0} : Aborting second thread.",
            DateTime.Now.ToString("HH:mm:ss.ffff"));
        thread.Abort("Terminating example.");
        // Đợi tiểu trình thứ hai kết thúc.
        thread.Join();
        break;
    }
} while (command != 'E');
// Nhấn Enter để kết thúc.
Console.WriteLine("Main method complete. Press Enter.");
Console.ReadLine();
}
}

```

9) Hãy chạy chương trình trong mục 12 và giải thích kết quả.

Phần 3: Sử dụng phương pháp lập trình đa luồng để giải một số bài toán sau:

- Sắp xếp chẵn lẻ
- Nhân 2 ma trận