LAB 5

I. Lý thuyết

- Sử dụng Tiểu trình (Thread)
- Cách tạo và thực thi tiểu trình
- Thread pool

II. Bài tập:

1) Thực thi và giải thích ý nghĩa của chương trình sau:

```
class MyThreadClass
{
    private const int RANDOM_SLEEP_MAX = 1000;
    private const int LOOP_COUNT = 10;

    private String greeting;

    public MyThreadClass(String greeting)
    {
        this.greeting = greeting;
    }
    public void runMyThread()
    {
        Random rand= new Random();
        for (int x = 0; x < LOOP_COUNT; x++)
        {
            Console.WriteLine(greeting + "(Thread ID:" + Thread.CurrentThread.GetHashCode() +")");
            try {
                Thread.Sleep(rand.Next(0, RANDOM_SLEEP_MAX));
            } catch (ThreadInterruptedException) { }

        }
    }
}
```

```
class ThreadExample
{
    static void Main(string[] args)
    {
        MyThreadClass mtc1 = new MyThreadClass("Day la tieu trinh thu 1");
        Thread thread1=new Thread(new ThreadStart(mtc1.runMyThread));
        thread1.Start();

        MyThreadClass mtc2 = new MyThreadClass("Day la tieu trinh thu 2");
        Thread thread2 = new Thread(new ThreadStart(mtc2.runMyThread));
        thread2.Start();

        Console.ReadKey();
    }
}
```

2) Viết chương trình TcpEchoServerThread theo mô tả và hướng dẫn sau:

- Đây là chương trình phía server có chức năng nhận dữ liệu từ client và phản hồi (echo) lại cho client (giống như lệnh ping).
- Chương trình hỗ trợ nhiều client kết nối đồng thời
- Mỗi khi có client kết nối sẽ tạo một thread mới cho việc nhận và gửi dữ liệu
- Ghi nhật kí (log): thông tin client, số byte dữ liệu echo.
- Chương trình nhận tham số đầu vào là số port khi thực thi. Ví dụ: ***TcpEchoServerThread 9001*** sẽ chạy chương trình server tại port 9001

Hướng dẫn:

- Bước 1: Tạo interface IProtocol với mục đích có thể phát triển thêm các protocol khác ngoài echo protocol (file: IProtocol.cs)

```
public interface IProtocol
{
    void handleclient();
}
```

- Bước 2: Tạo lớp EchoProtocol (file: EchoProtocol.cs)

```
class EchoProtocol : IProtocol
{
    public const int BUFSIZE = 32; // Byte size of IO buffer
    private Socket clntSock; // Connection socket
    private ILogger logger; // Logging facility

    public EchoProtocol(Socket clntSock, ILogger logger)
    {
        this.clntSock = clntSock;
        this.logger = logger;
    }
    public void handleclient()
    {
        ArrayList entry = new ArrayList();
        entry.Add("Client address and port = " + clntSock.RemoteEndPoint);
        entry.Add("Thread = " + Thread.CurrentThread.GetHashCode());
        try
        {
            int recvMsgSize; // Size of received message
            int totalBytesEchoed = 0; // Bytes received from client
            byte[] rcvBuffer = new byte[BUFSIZE]; // Receive buffer
            try
            {
                while ((recvMsgSize = clntSock.Receive(rcvBuffer, 0, rcvBuffer.Length, SocketFlags.None)) > 0)
                {
                    clntSock.Send(rcvBuffer, 0, recvMsgSize, SocketFlags.None);
                    totalBytesEchoed += recvMsgSize;
                }
            }
            catch (SocketException se)
            {
                entry.Add(se.ErrorCode + ": " + se.Message);
            }
```

```
            entry.Add("Client finished; echoed " + totalBytesEchoed + " bytes.");
        }
        catch (SocketException se)
        {
            entry.Add(se.ErrorCode + ": " + se.Message);
        }
        clntSock.Close();
        logger.writeEntry(entry);
    }
}
```

- Bước 3: Tạo interface ILogger ghi nhật kí kết quả echo (file: ILogger.cs)

```
public interface ILogger
{
    void writeEntry(ArrayList entry);// Write list of lines
    void writeEntry(String entry);// Write single line
}
```

- Bước 4: Thêm các lớp ConsoleLogger (xuất thông tin ra console) và FileLogger (ghi ra file)

ConsoleLogger.cs:

```
class ConsoleLogger : ILogger
{
    private static Mutex mutex = new Mutex();
    public void writeEntry(ArrayList entry)
    {
        mutex.WaitOne();
        IEnumerator line = entry.GetEnumerator();
        while (line.MoveNext())
            Console.WriteLine(line.Current);
        Console.WriteLine();
        mutex.ReleaseMutex();
    }
    public void writeEntry(String entry)
    {
        mutex.WaitOne();
        Console.WriteLine(entry);
        Console.WriteLine();
        mutex.ReleaseMutex();
    }
}
```

FileLogger.cs:

```
class FileLogger:ILogger
{
    private static Mutex mutex = new Mutex();
    private StreamWriter output;

    public FileLogger(String filename)
    {
        //Create log file
        output = new StreamWriter(filename, true);
    }
    public void writeEntry(ArrayList entry)
    {
        mutex.WaitOne();
        IEnumerator line = entry.GetEnumerator();
        while (line.MoveNext())
            output.WriteLine(line.Current);
        output.Flush();
        mutex.ReleaseMutex();
    }
    public void writeEntry(String entry)
    {
        mutex.WaitOne();
        output.WriteLine(entry);
        output.WriteLine();
        output.Flush();
        mutex.ReleaseMutex();
    }
}
```

- **Bước 5:** Chương trình chính

```
class TcpEchoServerThread
{
    static void Main(string[] args)
    {
        if (args.Length != 1)
            throw new ArgumentException("Parameter(s): <Port>");
        int serverPort = Int32.Parse(args[0]);

        TcpListener listener = new TcpListener(IPAddress.Any, serverPort);
        ILogger logger = new ConsoleLogger();
        listener.Start();
        for (; ; )
        {
            try
            {
                Socket client = listener.AcceptSocket();
                EchoProtocol protocol = new EchoProtocol(client, logger);
                Thread thread = new Thread(new ThreadStart(protocol.handleclient));
                thread.Start();
                logger.writeEntry("Created and started Thread = " + thread.GetHashCode());
            }
            catch (System.IO.IOException e)
            {
                logger.writeEntry("Error:" + e.Message);
            }
        }
    }
}
```

3) Viết chương trình TcpEchoClient để gửi dữ liệu lên server, chương trình nhận tham số là địa chỉ, số port của server và nội dung gửi đến server. Ví dụ ***TcpEchoClient 127.0.0.1 9001 "Data send to server"***