**Lab1 report (Yefimchuk)**

**Тема: Робота з потоками за допомогою засобів мови програмування С#.**

**Мета:** Вивчити основні оператори, класи та їх методи доступні у фреймворку Microsoft.Net для роботи з потоками.

**Текст програми**

using System;

using System.Threading;

class Program

{

static void Main(string[] args)

{

new Thread(Go).Start();

Go();

Console.ReadLine();

}

static void Go()

{

for (int i = 0; i < 5; i++)

Console.Write('P');

}

}

У цій програмі виконується одна функція двома потоками.

Результат: РРРРРРРРРР

class ThreadTest

{

bool done;

static void Main(string[] args)

{

ThreadTest tt = new ThreadTest();

new Thread(tt.Go).Start();

tt.Go();

Console.ReadLine();

}

void Go()

{

if (!done)

{

done = true;

Console.Write("Done");

}

}

}

Тут потоки розділили дані, бо відносяться до одного екземпляру. Функція виконається один раз.

Результат: Done

using System;

using System.Threading;

class ThreadTest

{

static bool done;

static void Main(string[] args)

{

new Thread(Go).Start();

Go();

Console.ReadLine();

}

static void Go()

{

if (!done)

{

done = true;

Console.Write("Done");

}

}

}

Результат буде тим самим, тільки вже розподіл даних буде відбуватися статичними полями

Результат: Done

using System;

using System.Threading;

class ThreadTest

{

static bool done;

static void Main(string[] args)

{

new Thread(Go).Start();

Go();

Console.ReadLine();

}

static void Go()

{

if (!done)

{

Console.Write("Done");

done = true;

}

}

}

Тут результат буде іншим, і функція виконається два рази.

Результат: DoneDone

using System;

using System.Threading;

class ThreadSafe

{

static bool done;

static object locker = new object();

static void Main(string[] args)

{

new Thread(Go).Start();

Go();

Console.ReadLine();

}

static void Go()

{

lock (locker)

{

if (!done)

{

Console.Write("Done");

done = true;

}

}

}

}

За допомогою locker один потік переходить в режим очікування, поки блокування не звільняється

Результат: Done

**Висновок: При виконанні двох потоків одночасно, потоки змагаються за виконання. Один з них можна блокувати за допомогою lock. Тоді виконається лише один потік. Також можна одним потоком очікувати закінчення іншого – оператор Join. Онлайн компілятор не дозволяє працювати з безкінечними циклами, тому не можливо побачити як потоки виконують свою конкретну дію.**