Laboratorium 12

1. Ćwiczenie 1

Zaimplementuje poniższy kod. Nie zapomnij dołączyć:

```
using System;
using System.Threading;
```

```
class ThreadExample
{
  static void Main()
  {
    Thread t = new Thread (Write1); //Uruchom inny  wątek
    t.Start();

    // Główny wątek.
    for (int i = 0; i < 1000; i++) Console.Write ("0");
  }

  //Inny wątek
  static void Write1()
  {
    for (int i = 0; i < 1000; i++) Console.Write ("1");
  }
}
```

Przedstaw wynik (wyjście) uruchomienia kodu.

2. Ćwiczenie 2

Wykonaj następujący kod.

```
static void Main()
{
  new Thread (Run).Start();      // Uruchom Run w innym wątku
  Run();                         // Uruchom Run w głownym wątku
}

static void Run()
{
  // Deklaracja i użycie zmiennej lokalnej - 'cycles'
  for (int cycles = 0; cycles < 5; cycles++) Console.Write ('x');
}
```

Ile razy zostanie wypisane x i jak sądzisz dlaczego ?

3. Ćwiczenie 3

Wykonaj następujący kod.

```
class ThreadTest
{
  bool done;

  static void Main()
  {
    ThreadExample tt = new ThreadTest();
    new Thread (tt.Run).Start();
    tt.Run();
  }

  // Zauważ, że Run jest teraz metodą instancji
void Run()
  {
     if (!done) { done = true; Console.WriteLine ("Done"); }
  }
}
```

Ile razy zostanie wypisany string „Done” i dlaczego ?

4. Ćwiczenie 4

Wykonaj następujący kod.

```
class ThreadExample
{
  static bool done;    // Pole statyczne

  static void Main()
  {
    new Thread (Run).Start();
    Run();
  }

  static void Run()
  {
    if (!done) { done = true; Console.WriteLine ("Done"); }
  }
}
```

Co pojawi się na wyjściu ? Czy zawsze jest to samo ?
Jak zmieni się wyjście jeśli zmienimy zawartość w funkcji Run(), następująco:

```
static void Run()
{
  if (!done) { Console.WriteLine ("Done"); done = true; }
}
```

5. Ćwiczenie 5

Środkiem zaradczym jest uzyskanie wyłącznej blokady i zapewnienia dostępu do danej przez jeden wątek podczas czytania i pisania w C # jest zastosowanie **lock()** . W tym celu zdefiniuj w klasie zmienną:

```
static readonly object locker = new object();
```

I odpowiednio zmień funkcję Run():

```
static void Run()
{
        lock (locker)
        {
                //exclusive lock
        }
}
```

6. Ćwiczenie 6

Wykonaj następujący kod. Jest to przykład oczekiwania jednego wątku na zakończenie innego.

```
static void Main()
{
  Thread t = new Thread (Run);
  t.Start();
  t.Join();
  Console.WriteLine ("Thread t has ended!");
}

static void Run()
{
  for (int i = 0; i < 777; i++) Console.Write ("☺");
}
```

Co się stanie jeśli usuniemy Join()?

7. Ćwiczenie 7

Wykorzystując wątki spróbuj napisać program, który na wejściu otrzymuje *pięć* kwadratowych macierzy i wylicza sumę wszystkich ich elementów, przy czym, dla każdej macierzy suma jest wyliczana w osobnym wątku.