1.

Верно, что класс Parallel содержит асинхронные реализации следующих циклов (*укажите все верные ответы***):**

1) ForEach.
2) While.
3) Do-While.
4) Repeat.
5) For.

2.

Выберите верные утверждения (*укажите все верные ответы***):**

1) В классе BackgroundWorker, WorkerReportsProgress имеет по умолчанию значение false.
2) В классе BackgroundWorker, при попытке вызвать метод ReportProgress() с параметром, выходящим из предела от 0 до 100, будет выброшен ArgumentOutOfRangeException.
3) В классе BackgroundWorker определены 2 события – DoWork и ProgressChanged.
4) Класс BackgroundWorker содержит метод для прерывания асинхронной операции.
5) Для возможности отмены асинхронной операции или отправления данных о прогрессе операций, требуется задать значение true соответствующим свойствам, в противном случае, будет выброшено исключение.

3.

В результате выполнения фрагмента программы:

```
using System;
using System.ComponentModel;

class Program {
    static void Main() {
        BackgroundWorker bgw = new BackgroundWorker {
            WorkerReportsProgress = true
        };
        bgw.DoWork += (sender, args) => {
            for (int n = 0; n < 5; n++) {
                Console.Write(1);
                bgw.ReportProgress(20 * n);
            }
        };
        bgw.ProgressChanged += (sender, args) => {
            Console.Write(5);
        };
        bgw.RunWorkerCompleted += (sender, args) => {
            Console.Write(2);
        };
        bgw.RunWorkerAsync();
        for (int i = 0; i < int.MaxValue; i++) { }
    }
}
```

на экран будет выведено:

4.

В результате выполнения фрагмента программы:

```
using System;
using System.ComponentModel;

class Program {
    static void Main() {
        BackgroundWorker bgw = new BackgroundWorker();
        for (int i = 0; i < 10; i++) {
            bgw.ReportProgress(i * 5 % 2);
        }
        bgw.ProgressChanged += (sender, args) => {
            Console.Write(args.ProgressPercentage);
        };
        for (int i = 0; i < int.MaxValue; i++) { }
    }
}
```

на экран будет выведено:

Примечание:
Если возникнет ошибка компиляции, введите: ***
Если ошибок и исключений нет, но на экран не выведется ничего, введите: ---
Если возникнет ошибка исполнения или исключение, введите: +++

5.

В результате выполнения фрагмента программы:

```
using System;
using System.Threading;

class Program {
    delegate long MyDel(int first, int second);
    static long Sum(int x, int y) {
        Thread.Sleep(1000);
        return x + y;
    }

    public static void Main() {
        MyDel del = new MyDel(Sum);
        var iar = del.BeginInvoke(5, 5, null, null);
        long result = del.EndInvoke(iar);
        Console.Write(result);
        Console.Write(5);
    }
}
```

на экран будет выведено:

6.

В результате выполнения фрагмента программы:

```
      using System;
using System.Threading;

class Program {
    delegate long MyDel(int first, int second);
    static long Sum(int x, int y) {
        Thread.Sleep(100);
        return x + y;
    }
    public static void Main() {
        MyDel del = new MyDel(Sum);
        var iar = del.BeginInvoke(10, 5, null, null);
        while (!iar.IsCompleted) {
            Console.Write(5);
            Thread.Sleep(70);
        }
        long result = del.EndInvoke(iar);
        Console.Write(result);
    }
}
```

на экран может быть выведено:

1) 5155
2) 5515
3) 5555515
4) 1555
5) 515

7.

Выберите пространства имён, в котором определён класс Timer (*укажите все верные ответы***):**

1) System.Threading.Tasks;
2) System.Windows.Forms;
3) System.Timers;
4) System.Threading;
5) System.Diagnostics;

8.

Асинхронный метод может иметь следующие типы возвращаемого значения (*укажите все верные ответы***):**

1) Task.
2) int.
3) Task<T>.
4) double.
5) void.

9.

Выберите верные утверждения (*укажите все верные ответы***):**

1) Использование ключевого слова async в объявлении метода без ключевого слова await вызовет ошибку компиляции.
2) Каждое из окон Windows Forms работает в отдельном потоке.
3) Открытие нового окна Windows Forms открывает новый поток, закрывая при этом старый в любом случае.
4) Асинхронные методы не поддерживают передачу параметров по ссылке (ref, out).
5) Асинхронные методы могут вызываться из синхронных.

10.

В результате выполнения фрагмента программы:

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

class Program {
    static long Factorial(int factor) {
        long res = 1;
        for (int k = 1; k <= factor; k++) res *= k;
        return res;
    }
    static async Task<long> FactorialAsync(int factor) {
        Console.Write(4);
        var eee = await Task.Run(() => Factorial(factor));
        Console.Write(2);
        return eee;
    }
    public static void Main() {
        Console.Write(1);
        var res = FactorialAsync(5);
        Console.Write(3);
        Console.Write(res.Result.ToString());
    }
}
```

на экран будет выведено:

Примечание:
Если возникнет ошибка компиляции, введите: ***
Если ошибок и исключений нет, но на экран не выведется ничего, введите: ---
Если возникнет ошибка исполнения или исключение, введите: +++

1	15
2	15
3	11155511525
4	+++
5	105
6	25
7	234
8	135
9	245
10	1432120