中山大学互联网与开源技术协会

2022 第二次面试技术类题目 C# 部分

注意事项

1. 本试题为原创题目，**请勿传播**。
2. 本试题用于**测试自学成果**，结果仅供第二次面试参考使用。
3. 本试题完成时间最长为 **120** 分钟，不要求必须完全全部任务。
4. 本试题的结果不完全由做出题目的数量决定，也由完成的过程、方式方法、质量及耗时等综合考虑，因此请勿盲目以做出更多题目为目标答题。
5. 我们欢迎你的思考，在做题中如果有任何想法和问题，欢迎保留并在后续与面试官交流的过程中提出。
6. 本试题最终解释权归中山大学互联网与开源技术协会所有。

负数求和

Description

创建一个空控制台项目，尝试编写一个函数，返回数组中所有负数的和。

```
public static int SumOfNegative(int[] arr)
{
    // TODO
}
```

Tasks

1. 完成函数 `SumOfNegative`。
2. 请在 `Program.cs` 中使用顶级语句，定义一个数组，调用定义于另一文件中的 `SumOfNegative` 并输出结果。
3. 使用 LINQ 实现 `SumOfNegative`。
4. 尝试更改函数定义，使得以下代码能够正常运行。

```
var arr = new int[] { 1, 2, 3, 4, 5 };
var sum = arr.SumOfNegative();
Console.WriteLine(sum);
```

格式化输出

Description

对象的格式化输出在 C# 中是一个非常常见的操作，你可以通过覆写 `ToString()` 方法来实现。

`ToString()` 方法继承自 C# 中一切类型的基类 —— `System.Object`，因此在很多情况下，默认调用对象的此方法进行字符串输出。参考如下的类型定义：

```
public class Person
{
    public string Name { get; set; }
    public int Age { get; set; }
}

public class Student : Person
{
    public int StudentNumber { get; set; }
    public double Score { get; set; }
    public string Major { get; set; }
}

public class Teacher : Person
{
    public string Course { get; set; }
    public string Department { get; set; }
}
```

你需要实例化一系列的对象，并覆写 `ToString()` 方法，使得输出的字符串格式如下：

```
========== Teacher ==========
Name     | Age | Course    | Department
John     |  30 | Math      | Computer Science
Mary     |  28 | English   | Foreign Language

========== Student ==========
Name     | Age | StdNumber    | Score | Major
Tom      |  18 | 00-2022-001 |  90.5 | Computer Science
Jerry    |  19 | 00-2022-002 |  80.0 | Computer Science
Sara     |  18 | 01-2022-003 |  95.0 | Computer Science
Jack     |  18 | 02-2022-004 |  85.3 | Computer Science
```

```
========== Person ==========
Name     | Age
John     |  30
Mary     |  28
Tom      |  18
Jerry    |  19
Sara     |  18
Jack     |  18
```

Tasks

1. 按照上述类型定义，实现 `Person`、`Student`、`Teacher` 三个类。
2. 实现 `ToString()` 方法，使得输出的字符串格式如上所示。
3. 请注意，输出的字符串中，每一列的宽度应该与上述示例中的一致。
4. 尝试利用内插字符串完成对象的格式化字符串的构建。

Hints

你可能会使用到 `is` 和 `as` 关键字。部分代码片段如下：

```
public static void TestPerson(IEnumerable<Person> persons)
{
    var stds = from p in persons
        where /* ??? */
        select /* ??? */;
    /* your code ... */
    Console.WriteLine("================== Student ==================");
    /* your code ... */
    stds.ToList().ForEach(std => Console.WriteLine(std));
    /* your code ... */
}
```

斐波那契数列

Description

斐波那群数列是一个非常有名的数列，它的定义如下：

$$F_0 = 0, F_1 = 1, F_n = F_{n-1} + F_{n-2}$$

请参考 C# yield 语句 实现一个迭代器（`IEnumerable<T>`）来生成斐波那契数列，并将其顺序输出。

Tasks

1. 实现一个迭代器，生成斐波那契数列的前 n 项，其中 n 为参数在构造函数中传入。
2. 迭代器可以用于解决什么问题？你能举出一些例子吗？
3. 实现一个无限迭代器，生成斐波那契数列的所有项，并在发生溢出时抛出异常。

计时器

Description

请实现一个定时器对象 `TickTimer`，它每秒钟触发一次 `Tick` 事件，允许多个事件处理程序同时订阅此事件，同时此事件触发时，会传递一个 `sec` 参数，表示自开始自当前的已经经过的秒数。

调用实例如下：

```
var timer = new TickTimer();
timer.Tick += (sec) => Console.WriteLine($"Tick @{sec}");
timer.Start();
Thread.Sleep(10000);
timer.Stop();
```

Tasks

1. 实现 `Timer` 类，使得上述调用可以正常工作。
2. 请注意，`Tick` 事件的处理程序应该在一个单独的线程中执行。

Hints

1. 你可能需要用到 `Thread` 类和 `Task` 类。
2. 你可能会用到 `event` 和 `delegate` 关键字。

异步上班族

Description

在当下计算机大多拥有不止一个 CPU 核心，因此多线程编程已经成为了必不可少的技能。

在 C# 中，我们使用 `Thread` 类来创建和管理线程，而对于更加常见的异步编程，我们可以使用 `Task` 类和 `async` / `await` 关键字。`Task` 是比 `Thread` 更加高级的抽象，我们所创建的 `Task` 对象会被调度到一个线程池中的线程上执行，而不是直接创建一个新的线程，这在一定程度上可以提高程序的性能和降低管理线程的开销。

在本题中，你需要实现一个 `AsyncWorker` 类，模拟它从起床到上班的过程，它需要完成以下任务：

1. 起床 `WakeUp`，需要花费 5 tick。
2. 吃早饭 `EatBreakfast`，需要花费 10 tick，你可以在吃早饭的过程中操作手机。
3. 点咖啡 `OrderCoffee`，需要花费 2 tick，咖啡需要准备 20 tick，但是咖啡师傅会在你点完咖啡后立即开始准备咖啡，因此你可以在咖啡准备好之前继续做其他事情。
4. 预定会议 `BookMeeting`，需要花费 5 tick，但是会议室需要提前至少 30 tick 预定，因此你需要在上班前 30 tick 预定会议室，但是你不能在订咖啡的时候预定会议室。
5. 坐地铁 `TakeSubway`，需要花费 15 tick。
6. 浏览邮件 `BrowseEmail`，需要花费 10 tick，你可以在搭乘地铁的过程中浏览邮件。
7. 取咖啡 `GetCoffee`，需要花费 2 tick，但是你需要等待咖啡师傅准备好咖啡。
8. 上班 `GoToWork`，需要花费 3 tick，需要完成所有任务后才能上班。

你需要实现 `AsyncWorker` 类，模拟上述过程。

Tasks

1. 你需要使用 `Task` 类和 `async` / `await` 关键字来实现异步编程。
2. 实现 `AsyncWorker` 类，使得这位上班族可以顺利开始工作，并记录总体花费的时间。
3. 请注意，你需要模拟全部的并行过程，包括咖啡师煮咖啡的过程。同时，你需要输出每个任务的开始和结束时间。
4. 你可以使用一个静态变量来定义 tick 的时间，例如 `static int tick = 500`，表示每个 tick 需要 500 毫秒。

Hints

1. 你可以使用如下的方式模拟一个事情：

```
public async static Task DoSomethingAsync() {
    Console.WriteLine($"[{DateTime.Now:HH:mm:ss.fff}] Start...");
    await Task.Delay(1000);
    Console.WriteLine($"[{DateTime.Now:HH:mm:ss.fff}] Finish...");
}
```

2. 你可以使用 `Task.WhenAll` 和 `Task.WhenAny` 来等待多个任务完成，控制流程。
3. 你可以将咖啡师傅的准备咖啡的过程也模拟为一个异步函数，并在 `OrderCoffee` 方法返回其调用结果。
4. 时刻注意 `await` 关键字的使用，以及对 `Task` 使用 `await` 的时机。

简单博客系统

Description

在本题中，你需要使用 ASP.NET Core + EFCore + SQLite + React 实现一个简单的博客系统。

可以参考：使用 ASP.NET Core、最小 API 和 .NET 6 创建 Web 应用和服务

Tasks

以下任务你可以根据自身情况选择完成，前端为可选任务。

1. 实现博文的增删改查功能，不考虑用户系统，博文 ID 使用 6 位随机字符。
2. 实现博文的评论功能，可以根据博文 ID 获取评论。
3. 实现博文的标签功能，可以为博文添加多个标签，尝试只用 JSON 列表存储标签。
4. 为你的博客系统实现一个前端页面，实现简单的博客和评论只读查看功能。你并不需要实现编辑、增加和删除功能，可以通过 API 进行实现。
5. 尝试使用 `marked` 库来实现 Markdown 的解析，使你的博客支持 Markdown 语法。

思考问题：

1. 数据库、后端 API、前端页面分别是什么？它们是如何配合工作的？
2. 你对现代 Web 开发有什么新的理解？数据是如何在数据库和用户之间传输的？
3. 异步编程，及 `async` / `await` 关键字是如何帮助现代 Web API 实现高并发的？
4. 有并发就可能存在冲突，你有什么解决相关问题的思路？
5. 基于框架的功能实现相对于从零开始的项目有什么优势？在进行这些开发的过程中你专注于实现什么？