1. **Что такое сериализация?**
   Сериализация — это процесс преобразования объектов в поток байтов для хранения объекта в памяти, базе данных или в файле. Основное назначение сериализации — сохранить состояние объекта для того, чтобы иметь возможность воссоздать его при необходимости. Обратный процесс называется десериализация.
2. **Что произойдет если сереализуемый объект будет изменен, а затем произведем десериализацию этого объекта?**
   Для проверки было создано небольшое консольное приложение. Выяснилось, что при добавление новых полей, ничего плохого не происходит, просто эти поля будут иметь начальные значения. Если же мы убираем какое-то поле, то мое первоначальное предположение, что произойдет выброс ошибки, как в Java, не оправдался. Эти поля были просто проигнорированы.
3. **Для чего может понадобиться FileStream и BinaryFormatter?**
   BinaryFormatter может сериализовать/десериализовать объект, но для его работы необходим поток, для этого можно использовать FileStream, так как он предоставляет поток в файле.
4. **Что такое поток ввода/вывода?**
   Поток ввода/вывода относится к передаче данных с носителя информации или на него. В пространстве имен System.IO можно найти типы, которые обеспечивают операции чтения/записи для потоков и файлов.
5. **Почему в программе PersonsToFile когда мы произвели десериализацию, добавили новых объектов и ходим произвести сериализацию файл не перезаписывается?**
   При десериализации позиция потока файла доходит до конца и возможно этот параметр в сериализации играет роль выхода из итерации и раз мы уже достигли конца, то и перезаписать нам нечего, сразу происходит выход из итерации.
6. **Как исправить такое поведение?**
   Мной было найдено как минимум два способа. Первый способ, это закрыть поток файла, а затем обратно его создать. Тогда положение потока (position), будет равняться нулю. Но раз нам необходимо, что бы позиция потока было на нуле, проще эту позицию задать, это второй способ.
7. **Помимо BinaryFormatter как еще можно записать сереализуемый объект в файл?**
   Для этих целей можно использовать BinaryWriter и BinaryReader. BinaryWriter соответственно записывает сереализуемый объект в файл, а BinaryReader считывает объект из графа. Так же для сериализации можно использовать XmlSerializer, SoapFormatter, DataContractJsonSerializer. Они в отличии от бинарной сериализации имеют более читаемый вид.
8. **В чем различие BinaryFormatter от BinaryWriter и BinaryReader?**
   BinaryFormatter в отличии от BinaryWriter/BinaryReader сеарилизует и десериализует объект или весь граф связанных объектов, в то время как BinaryWriter/BinaryReader может только записывать/считывать простые типы данных в поток.
9. **Помимо использования FileStream, как еще можно получить поток файла?**
   Можно воспользоваться классом FileInfo, а точнее его методы такие как Create, Open и OpenRead() так как эти методы возвращают поток файла.

10. **Перечислите какие могут быть события у FileSystemWatcher и для чего они нужны?**
    FileSystemWatcher – ожидает уведомления файловой системы об изменениях и инициирует события при изменении каталога или файла в каталоге. Для работы с файлами/каталогами есть четыре основных события:
    - Changed – происходит при изменении файла или каталога в заданном пути
    - Created – происходит при создании файла или каталога в заданном пути
    - Deleted – происходит при удалении файла или каталога в заданном пути
    - Renamed – происходит при переименовании файла или каталога в заданном пути

    Есть еще события Desposed и Error.
11. **Для чего может понадобиться отслеживание файлов/каталогов?**
    Первое что приходит в голову, это логирование файлов. Т.е. когда, кто и что сделал с файлом (открыл его и закрыл или изменил). Ситуация, что у нас есть сервер с веб-приложением на интерпретируем языке, кто-то получил несанкционированный доступ к серверу, и злоумышленник изменил исходный код или добавил свой вредоносный код и при помощи отслеживания таких действий мы можем их зафиксировать и записать в логирующий файл. Так причину странного поведения веб-приложения легче будет определить.
12. **Что необходимо сделать, чтобы в реальном времени отслеживать добавление и снятие логических дисков?**
    Для этого необходимо воспользоваться знаниями из категорий по отслеживанию файлов и получении информации о логических дисках. Т.е. нам нужен класс который бы отслеживал добавление/снятия логического диска и инициировал событие при таких действиях. К сожалению, я не нашел удобного класса, для этой работы, как например работа с файлами FileSystemWatcher, но есть класс ManagementEventWatcher. Он хорош тем, что мы ему говорим, что мы хотим отслеживать, при помощи WqlEventQuery. Т.е. по сравнению с FileSystemWatcher и у нас появляется больше свободы.