Ответы на вопросы л. р. по ООП №1

1. **Что такое .Net Framework и из чего он состоит?** .Net Framework – платформа, которая создана Microsoft для разработки приложений. Состоит из CLR, FCL/BCL, CTS, CLS, IL.
2. **Что такое CLR, FCL/BCL, CLI, IL?**
    1) (CLR) Common Language Runtime). Общеязыковая исполняющая среда. CLR отвечает за выполнение кода, управление памятью, обработку исключений и безопасность. Она позволяет программам, написанным на разных языках программирования, работать вместе
    2) FCL (Framework Class Library). Это обширная библиотека классов, предоставляемая .NET Framework. Она включает в себя множество классов, интерфейсов и типов данных, которые разработчики могут использовать для создания приложений.
    3) BCL (Base Class Library). Это подмножество FCL, которое включает в себя основные классы, такие как коллекции, примитивные типы данных, ввод-вывод и другие базовые функциональности.
    4) CLI (Common Language Infrastructure). Это спецификация, определяющая архитектуру и поведение среды выполнения, которая поддерживает выполнение кода, написанного на разных языках программирования. CLI включает в себя спецификации для метаданных, системы типов и виртуальной машины выполнения.
    5) IL (Intermediate Language). Это промежуточный язык, в который компилируется исходный код, написанный на .NET-совместимых языках программирования (например, C#, VB.NET). IL-код затем компилируется в машинный код во время выполнения с помощью JIT-компилятора (Just-In-Time).
3. **Пояснить работу JIT-компилятора?** JIT-компилятор (Just-In-Time компилятор) — это компонент среды выполнения, который компилирует байт-код в машинный код непосредственно во время выполнения программы.
    1) CLR ищет типы данных и загружает во внутренние структуры.
    2) Для каждого метода CLR заносит адрес внутренней CLR функции JITCompiler.
    3) JITCompiler ищет в метаданных соответствующей сборки IL-код вызываемого метода, проверяет и компилирует IL-код в машинные команды.
    4) Они хранятся в динамически выделенном блоке памяти.
    5) JITCompiler заменяет адрес вызываемого метода адресом блока памяти, содержащего готовые машинные команды.
    6) JITCompiler передает управление коду в этом блоке памяти.

4. **Что такое CTS (Common Type System)?** Это компонент .NET Framework, который определяет, как типы данных объявляются, используются и управляются в приложениях .NET. CTS обеспечивает совместимость и взаимодействие между различными языками программирования, поддерживаемыми .NET.
5. **Какие аспекты поведения определяет тип System.Object?** Тип System.Object определяет базовое поведение для всех типов в .NET. Основные методы включают:
    1) Equals(Object obj): Определяет, равен ли текущий объект другому объекту.
    2) GetHashCode(): Возвращает хэш-код для текущего объекта.
    3) GetType(): Возвращает объект Type, представляющий тип текущего экземпляра.
    4) ToString(): Возвращает строковое представление текущего объекта.
    5) Finalize(): Позволяет объекту попытаться освободить ресурсы перед уничтожением сборщиком мусора.
    6) MemberwiseClone(): Создает поверхностную копию текущего объекта1.
6. **Что находится в mscorlib dll?** (Microsoft Common Object Runtime Library) — это основная библиотека .NET Framework, содержащая базовые классы и типы, такие как System.String, System.Int32, System.Array, и многие другие. Она включает в себя основные функциональные возможности, необходимые для работы приложений .NET.
7. **Что такое «сборка»? Из чего состоит сборка .NET?** Сборка в .NET — это базовая единица развертывания, управления версиями и безопасности. Сборка состоит из:
    1) Манифеста: содержит метаданные о сборке.
    2) Типов и ресурсов: код и данные, которые работают вместе.
    3) Метаданных: информация о типах, версиях и зависимостях.
8. **Какие виды сборок существуют?**
    1) Частные сборки: используются только одним приложением.
    2) Общие сборки: могут использоваться несколькими приложениями и хранятся в глобальном кэше сборок (GAC).
    3) Сателлитные сборки: содержат ресурсы для локализации.
9. **Что такое assembly manifest?** Assembly manifest — это часть сборки, содержащая метаданные, необходимые для описания версии сборки, ее зависимостей и безопасности. Манифест может быть встроен в PE-файл (например, .exe или .dll) или существовать как отдельный файл.
10. **Что такое GAC?** GAC (Global Assembly Cache) — это глобальный кэш сборок, используемый для хранения общих сборок, которые могут быть использованы несколькими приложениями на одном компьютере. Сборки, помещенные в GAC, должны иметь сильное имя.

11. **Чем managed code отличается от unmanaged code?**
    1) Managed code: Код, который выполняется под управлением CLR (Common Language Runtime) и использует все преимущества .NET, такие как сборка мусора, безопасность типов и межъязыковая совместимость.
    2) Unmanaged code: Код, который выполняется напрямую операционной системой, вне контроля CLR. Примеры включают код на C++ без использования .NET.
12. **Как и для чего определен метод Main?** Метод Main является точкой входа для всех приложений .NET. Он определяет, с чего начинается выполнение программы. Метод Main может иметь различные сигнатуры, включая void Main(), int Main(), void Main(string[] args), и int Main(string[] args).
13. **Варианты использования директивы using (using Directive) в C#.** Директива using в C# используется для:
    1) Импортирования пространств имен: позволяет использовать типы из других пространств имен без указания полного имени.
    2) Управления ресурсами: используется в конструкции using для автоматического освобождения ресурсов, таких как файлы или соединения с базой данных9.
14. **Как связаны между собой сборки и пространства имен?** Пространства имен организуют типы в логические группы и предотвращают конфликты имен. Сборки содержат типы и ресурсы, которые могут быть организованы в пространства имен. Пространства имен могут распространяться на несколько сборок, и одна сборка может содержать типы из нескольких пространств имен.
15. **Что такое примитивные типы данных? Перечислите их.** Примитивные типы данных — это базовые типы, встроенные в язык программирования. В C# к примитивным типам относятся: byte, sbyte, short, ushort, int, uint, long, ulong, float, double, decimal, char, bool, string.
16. **Что такое ссылочные типы? Какие типы относятся к ним?** Ссылочные типы в C# хранят ссылку на область памяти, где находятся данные. Примеры ссылочных типов включают:
    1) Классы (class)
    2) Интерфейсы (interface)
    3) Массивы
    4) Делегаты (delegate)
    5) Строки (string).
17. **Какие типы относятся к типам-значениям?** Типы-значения хранят свои данные непосредственно в памяти. Примеры типов-значений включают:
    1) Целочисленные типы (int, short, long, byte)
    2) Типы с плавающей точкой (float, double)
    3) Булевы типы (bool)
    4) Символьные типы (char)

5) Структуры (struct)
6) Перечисления (enum).

18. **В чем отличие между ссылочными и значимыми типами данных?**
    1) Ссылочные типы: хранят ссылку на данные, расположенные в куче. Изменения, внесенные через одну переменную, отражаются на всех переменных, ссылающихся на тот же объект.
    2) Типы-значения: хранят данные непосредственно в памяти стека. Каждая переменная имеет свою копию данных, и изменения в одной переменной не влияют на другие.
19. **Что такое упаковка и распаковка значимых типов?**
    1) Упаковка: процесс преобразования типа-значения в тип object или любой интерфейс, который он реализует. Значение упаковывается в объект и хранится в куче.
    2) Распаковка: обратный процесс, извлечение типа-значения из объекта. Распаковка требует явного приведения типа.
20. **Разница между int и System.Int32, double и System.Double.** В C# int и System.Int32 являются синонимами и представляют собой 32-битное целое число со знаком. Аналогично, double и System.Double представляют собой 64-битное число с плавающей запятой двойной точности. Эти типы могут использоваться взаимозаменяемо.
21. **Использование типа dynamic.** Тип dynamic позволяет пропускать проверку типов на этапе компиляции, что делает его полезным для работы с COM-объектами, динамическими языками и отражением. Однако ошибки будут обнаружены только во время выполнения.
22. **В чем заключается главное отличие между var и dynamic?**
    1) var: Тип переменной определяется на этапе компиляции. Компилятор выводит тип переменной на основе присвоенного значения.
    2) dynamic: Тип переменной определяется на этапе выполнения. Компилятор не проверяет тип переменной на этапе компиляции, что позволяет изменять тип переменной во время выполнения.
23. **Что такое неявно типизированная переменная?** Неявно типизированная переменная объявляется с использованием ключевого слова var. Тип переменной выводится компилятором на основе присвоенного значения.
24. **Для чего используют Nullable тип?** Nullable типы (Nullable<T> или T?) позволяют присваивать переменным значимых типов значение null. Это полезно, когда нужно указать, что переменная может не иметь значения.
25. **Объявление строкового литерала и операции со строкой.** Строковый литерал можно объявить с помощью двойных кавычек: string str = "Hello, World!"; Со строками можно выполнять различные операции, такие как конкатенация, сравнение, получение длины строки и т.д.

26. **Способы задания и инициализации строк.** Строки можно задавать и инициализировать несколькими способами:
    1) Объявление и инициализация;
    2) Использование литералов;
    3) Использование конструктора.
27. **Методы типа String.** Тип String предоставляет множество методов, включая:
    1) Substring(): Извлечение подстроки.
    2) IndexOf(): Поиск индекса символа или подстроки.
    3) Replace(): Замена подстроки.
    4) ToUpper() и ToLower(): Преобразование строки в верхний или нижний регистр.
    5) Split(): Разделение строки на массив подстрок.
28. **Различие между пустой и null строкой**
    1) Пустая строка (""): строка, содержащая ноль символов.
    2) null строка: переменная, которая не указывает на объект строки.
29. **Как можно выполнить сравнение строк?** Сравнение строк можно выполнить с помощью метода Equals, оператора == или метода Compare.
30. **В чем отличие типов String и StringBuilder?**
    1) String: Неизменяемый тип. Каждое изменение строки создает новый объект.
    2) StringBuilder: Изменяемый тип. Позволяет изменять содержимое без создания новых объектов, что делает его более эффективным для частых изменений строк.
31. **Явные преобразования с помощью команд Convert.** Команды Convert используются для явного преобразования типов: int num = Convert.ToInt32("123");
32. **Консольный ввод/вывод.** Для выполнения консольного ввода/вывода используются методы Console.ReadLine() и Console.WriteLine().
33. **Что такое ступенчатый массив? Как его задать?** Ступенчатый массив — это массив массивов, где каждый вложенный массив может иметь разную длину.
34. **Какие типы можно использовать в foreach? Приведите пример.** В foreach можно использовать любые типы, которые реализуют интерфейс IEnumerable или IEnumerable<T>.
35. **Что такое кортеж?** Кортеж (tuple) — это упорядоченная последовательность элементов фиксированной длины, которая может содержать объекты различных типов. В отличие от массивов, кортежи неизменяемы, то есть после создания их содержимое нельзя изменить.
36. **Для чего используется кортеж?** Кортежи используются для группировки связанных данных в один объект. Это удобно, когда нужно вернуть несколько значений из функции или передать несколько параметров в функцию. Кортежи также могут использоваться для

создания сложных ключей в словарях или для хранения данных, которые не должны изменяться.

37. **Что такое локальная функция? Какова область ее видимости?** Локальная функция — это функция, определенная внутри другого метода. Она доступна только в пределах этого метода.
38. **Разница между checked и unchecked блоками**
    1) checked: проверяет переполнение арифметических операций и выбрасывает исключение.
    2) unchecked: игнорирует переполнение.
39. **Контекст по умолчанию и его переопределение.** По умолчанию используется unchecked контекст для арифметических операций. Можно переопределить с помощью ключевых слов checked и unchecked.
40. **Ключевое слово fixed и его использование.** Ключевое слово fixed используется для закрепления переменной в памяти, чтобы сборщик мусора не перемещал ее. Это необходимо при работе с указателями в небезопасном контексте.