**Лаба 2**

1. **Платформа .NET Framework** — это технология, которая поддерживает создание и выполнение веб-служб и приложений Windows. Платформа .NET Framework состоит из общеязыковой среды выполнения (среды CLR) и библиотеки классов .NET Framework. Основой платформы .NET Framework является среда CLR.
2. **CLR**, общеязыковая среда выполнения.Она обеспечивает интеграцию языков и позволяет объектам благодаря стандартному набору типов и метаданным), созданным на одном языке, быть «равноправными гражданами» кода, написанного на другом.

**CIL**, общий промежуточный язык. это промежуточный код, с помощью которого компилятор может компилировать код .NET в среду CLR. Это язык между языками высокого уровня (например, C #) и инструкциями ЦП.

**FCL,** библиотека классов Framework**.** FCL предоставляет крупномасштабную среду программирования. Это среда, предназначенная для различных приложений. Большинство реализаций FCL относятся к BCL.

**BCL,** библиотека базовых классов. Это общая среда программирования, называемая библиотекой базовых классов, которую могут использовать разработчики всех языков. Это одна из спецификаций CLI (Common Language Infrastructure), которая в основном включает в себя: выполнение сетевых операций, выполнение операций ввода-вывода, управление безопасностью, текстовые операции, операции с базой данных, операции XML, взаимодействие с журналами событий, отслеживание и некоторые другие.

**IL (Intermediate Language)** - код, содержащий набор инструкций, не зависящих от платформы. Иными словами, после компиляции исходного кода он преобразуется не в код для какой-то определенной платформы, а в промежуточный код на языке IL.

1. «**Just** **In** **Time**» - это стратегия компилятора, при которой, во время выполнения приложения, компилируются части кода, так что вместо интерпретирования каждый раз, можно всегда использовать скомпилированную версию. Когда среда CLR выполняет IL код, то происходит следующее: при вызове блока IL-кода JIT-компилятор проверяет и преобразует его в машинные команды которые он сохраняет в динамическом блоке памяти. Затем, он возвращается к структуре данных типа и заменяет адрес вызываемого метода адресом этого блока памяти и при следующем запросе повторная компиляция не будет выполнена, а будет использован уже скомпилированный код.
2. **Common** **Type** **System** (сокр. **CTS**, рус. Общая система типов) — часть .NET Framework, формальная спецификация, определяющая, как какой-либо тип (класс, интерфейс, структура, встроенный тип данных) должен быть определён для его правильного выполнения средой .NET. Кроме того, данный стандарт определяет, как определения типов и специальные значения типов представлены в компьютерной памяти.
3. Неявно считается базовым классом для всех остальных классов и типов, включая и типы значений. Иными словами, все остальные типы являются производными от object.
4. Что находится в MSCorLib dll? Параметры криптографии. Криптогра́фия — наука о методах обеспечения конфиденциальности, целостности данных, аутентификации, шифрования.
5. **Сборка** — двоичный файл ([exe](https://ru.wikipedia.org/wiki/Exe" \o "Exe) или [dll](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8_%D0%BF%D0%BE%D0%B4%D0%BA%D0%BB%D1%8E%D1%87%D0%B0%D0%B5%D0%BC%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0" \o "Динамически подключаемая библиотека)), содержащий номер версии информационного продукта (ИП), метаданные и [инструкции для виртуальной машины .NET](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4). Файл с исходным кодом, написанном на любом языке поддерживаемой .NET платформой (C#, VB.NET...), с помощью соответствующего компилятора (C# компилятора, VB.NET компилятора...) компилируется в сборку.
6. **Блять я не ебу**
7. **Манифест** сборки (**assembly** **manifest**) - это внутренняя часть сборки, которая позволяет идентифицировать сборку, указывает файлы, которые включаются в реализацию сборки, описывает типы и ресурсы, используемые в сборке, указывает зависимости от других сборок, а также набор прав доступа, которые необходимы сборке для корректной работы.
8. **GAC** — Глобальный Кэш Сборок) — в инфраструктуре .NET — подсистема, хранящая сборки CLI (CLI assembly) в централизованном репозитории. Система разработана для борьбы с проблемами разделяемых библиотек, например таких, как DLL hell.
9. Управляемый код (managed code)  – автоматическая сборка мусора. Можно просто создавать объекты.

Неуправляемый код (unmanaged code) – нужно удалять объекты в ручную.

В первом приближении управляемым кодом называется код, выполнение которого управляется средой выполнения. В этом случае соответствующая среда выполнения называется **общеязыковой средой выполнения** или средой CLR, независимо от реализации ([Mono](https://www.mono-project.com/), .NET Framework или .NET Core). Среда CLR отвечает за использование управляемого кода, его компиляцию в машинный код и последующее выполнение.Кроме того, среда выполнения предоставляет несколько важных служб, таких как автоматическое управление памятью, границы безопасности, безопасность типа и т. д.

1. Хз
2. 
3. Хз
4. **Примитивный (встроенный, базовый) тип** — [тип данных](https://ru.wikipedia.org/wiki/%D0%A2%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), предоставляемый [языком программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) как базовая встроенная единица языка. Примеры: bool, byte, sbyte, short, int, uint, float, double, char, string, object.
5. Ссылочные типы:

Тип object

Тип string

Классы (class)

Интерфейсы (interface)

Делегаты (delegate)

В переменных **ссылочных типов** хранятся ссылки на их данные (объекты), а переменные **типа** значений содержат свои данные непосредственно. Две переменные **ссылочного типа** могут ссылаться на один и тот же объект, поэтому операции над одной переменной могут затрагивать объект, на который ссылается другая переменная.

1. Типы значений:

Целочисленные типы (byte, sbyte, short, ushort, int, uint, long, ulong)

Типы с плавающей запятой (float, double)

Тип decimal

Тип bool

Тип char

Перечисления enum

Структуры (struct)

1. Главное **отличие** - это где храниться значение переменной. Для **ссылочных типов** - это куча, для **значимых** - это стек. При этом, **ссылочная** переменная (сама ссылка) храниться на стеке. **Значимыми** могут быть только экземпляры структур, в противоположность им **ссылочные** - это экземпляры классов.
2. **Упаковка и распаковка значимых типов в C#** — доступный для программиста механизм преобразования размерных типов данных языка [C#](https://ru.wikipedia.org/wiki/C_Sharp) из значимых в ссылочные и обратно через задействование свойств фундаментального базового класса *Object*.

**Упаковка**. В языке C# упаковка работает как неявное преобразование экземпляра любого размерного типа в объект базового класса *Object*, которое происходит, когда [компилятор](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) наталкивается на значимый тип в том контексте, где ожидается появление ссылки.

**Распаковка.** это преобразование **Object** типа в **value**(**значимый**) тип. **При распаковке происходит следующее:**  
• Сначала извлекается адрес поля из упакованного объекта  
• значения поля копируется из кучи в экземпляр значимого типа, находящийся в стеке.