1. Что такое .Net Framework и из чего он состоит?

Microsoft.NET (.NET Framework) – программная платформа. Содержит следующие основные компоненты:

***CLR (Common Language Runtime)*** – общеязыковая среда исполнения, виртуальная машина на которой исполняются все приложения, работающие в среде .NET. Реализация CLI VES компанией Microsoft. Компилятор ***JIT(Just in Time)***.

***MSIL(Microsoft IL)*** – реализация CLI CIL компанией Microsoft.

***FCL (Framework Class Library)*** – реализация CLI BCL компанией Microsoft. Можно рассматривать, как API CRL.

2. Поясните, что такое CLR-среда.

***CLR (Common Language Runtime)*** – общеязыковая среда исполнения, виртуальная машина на которой исполняются все приложения, работающие в среде .NET.

3. Что такое FCL?

FCL (.NET Framework Class Library) – объектно-ориентированная библиотека классов, интерфейсов и системы типов (типов-значений).

4. Какая наименьшая исполнимая единица в .NET?

**Сборка**(assembly) - один или несколько файлов, содержащий логический набор функциональности (код и другие данные, связанные с кодом). Бывают статические сборки, хранящиеся на диске, и динамические, которые создаются во время выполнения программы. Сборка - это базовый блок приложения, все ресурсы, относящиеся к ней, доступны или только внутри этого блока, или экспортируются наружу. При выполнении сборка задает область видимости имен и следит за ее соблюдением.

5. Что такое IL?

**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.

6. Пояснить работу JIT-компилятора?

Вместо компиляции всего приложения за один проход (что может привести к задержкам при запуске), JIT-компилятор просто компилирует каждую порцию кода при ее вызове (т.е. оперативно). Если промежуточный код однажды скомпилирован, то результирующий машинный исполняемый код сохраняется до момента завершения работы приложения, поэтому его перекомпиляция при повторных обращениях к нему не требуется.

7. Что такое CTS (Common Type System)?

CTS определяет, как типы объявляются, используются и управляются в среде исполнения общего языка, а также является важной частью поддержки межязыковой интеграции во время выполнения.

8. Какие аспекты поведения определяет тип System.Object?

9. Что находится в MSCorLib dll?

10. Что такое частные и общие сборки?

Частные находятся в каталоге программы, а общие в GAC

11. Что такое assembly manifest?

**Манифест сборки** (assembly manifest) - это внутренняя часть сборки, которая позволяет ей быть самоописанной. Assembly manifest 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. Эта информация используется во время выполнения для разрешения ссылок, проверку корректности версий, проверку целостности загруженных сборок.

12. Что такое GAC?

**GAC**— это глобальный кэш сборок. В нем хранятся совместно используемые сборки.

13. Чем managed code отличается от unmanaged code

Управляемый код (managed code) - это код, работающий в среде CLR. Содержит метаданные, в которых находится информация для среды выполнения - о типах, членах и ссылках, используемых в коде. **Неуправляемый код - это обычный машинный код**.

14. Как и для чего определен метод Main?

Это так называемая точка входа в программу.

15. Варианты использования директивы using( using Directive ) в C#.

Директива using используется в следующих трех целях.

* Для разрешения использования типов в пространстве имен, чтобы не нужно было квалифицировать использование типа в этом пространстве имен:

using System.Text;

* Для разрешения доступа к статическим членам и вложенным типам без необходимости квалифицировать доступ с помощью имени типа.

using static System.Math;

* Чтобы создать псевдоним для пространства имен или типа. Это называется *директивой using static*.

using Project = PC.MyCompany.Project;

16. Как связаны между собой сборки и пространства имен?

Пространство имен представляет собой логическую группировку типов (в основном, чтобы избежать столкновений имен). Сборка может содержать типы в нескольких пространствах имен и одно пространство имен может быть распространено по сборкам.

17. Что такое примитивные типы данных? Перечислите их.

Типы данных, которые поддерживаются компилятором напрямую, называются примитивными (primitive types) или встроенными у них существуют прямые аналоги в библиотеке классов .NET Framework Class Library.

Тип C# Размер в битах Тип System (FCL)

sbyte 8 System.Sbyte

short 16 System.Int16

int 32 System.Int32

long 64 System.Int64

byte 8 System.Byte

ushort 16 System.Uint16

uint 32 System.Uint32

ulong 64 System.Uint64

char 16 System.Char

bool 8 System.Boolean

float 32 System.Single

double 64 System.Double

decimal 128 System.Decimal

string - System.String

object - System.Object

18. Что такое ссылочные типы? Какие типы относятся к ним?

Типы- ссылки

• Классы

• Строки

• Массивы

• Делегаты

• Интерфейсы

**Ссылочные типы**(reference types) определяются class (в куче)

Делегат – это объект, инкапсулирующий ссылку на метод.   
Иными словами это тип, который ссылается на метод. Как только делегату назначен метод, он начинает работать точно также как и этот метод. Метод Делегат, может быть использован точно также как и любой другой метод с параметрами и возвращать значение.

19. Какие типы относятся к типам-значениям?

• Структуры

• Перечисления

• Простые типы

20. В чем отличие между ссылочными и значимыми типами данных?

**Типы значений** (value types) Размещение в стеке потока

**Ссылочные типы**(reference types) в куче

21. Что такое упаковка и распаковка значимых типов?

Упаковкой (boxing) называется процесс преобразования типа значения в тип System.Object или в тип интерфейса, который реализуется данным типом-значением

Int32 x = 5;

Object o = x;

// Упаковка x; o ссылается на упакованный объект

распаковка (unboxing) получение указателя на исходный значимый тип (поля данных), содержащийся в объекте.

byte m = (byte)(int)o; // Распаковка, а затем приведение типа

22. Для чего используется тип dynamic?

►Назначение :

► Отражения или коммуникации с другими компонентами

►Использование: для членов класса - поля, свойства/индексаторы, структур , для метода, делегата, или унарных/бинарных операторов

23. Что такое неявно типизированная переменная?

Как правило, при объявлении переменной сначала указывается тип, например int или bool, а затем имя переменной. Но начиная с версии C# 3.0, компилятору предоставляется возможность самому определить тип локальной переменной, исходя из значения, которым она инициализируется. Такая переменная называется неявно типизированной.

Неявно типизированная переменная объявляется с помощью ключевого слова **var** и должна быть непременно инициализирована. Для определения типа этой переменной компилятору служит тип ее инициализатора, т.е. значения, которым она инициализируется:

24. Для чего используют Nullable тип?

Значение null по умолчанию могут принимать только объекты ссылочных типов. Однако в различных ситуациях бывает удобно, чтобы объекты числовых типов данных имели значение null, то есть были бы не определены. Стандартный пример - работа с базой данных, которая может содержать значения null. И мы можем заранее не знать, что мы получим из базы данных - какое-то определенное значение или же null.

25. Как объявить строковый литерал? Какие операции можно выполнять со строкой?

Строковые литералы

String path;

path = "C:\\Windows\\regedit.exe"; //верно

path = @"C:\Windows\regedit.exe"; //верно,наиболее

path = "C:/Windows/regedit.exe"; //верно

path = "C:\Windows\regedit.exe"; //неверно, специального

► Поставить ещё один символ обратной косой черты (“\\”);

► Предварить строковый литерал символом ‘@’;

► В случае, если литерал есть путь к файлу (локальный или

сетевой), каталогу или web-странице, можно использовать

альтернативный символ '/' вместо “\\”.

присваивание (=);

► проверка на равенство содержимого (==);

► проверка на неравенство (!=);

► обращение по индексу ([]);

► сцепление (конкатенация) строк (+)

► <,>, >=,<= - сравнивают ссылки!!!!!!!!

26. Какие есть способы для задания и инициализации строк?

Создание строки:

char[] a = { '0', '0', '0' };

// создание массива символов:

string s;

// инициализация отложена

string t = "qqq";

// инициализация строковым литералом

string u = new string(' ', 20);

// с пом. конструктора

string v = new string(a);

// создание из массива символов

27. Какие методы есть у типа String?

**Compare** Сравнение двух строк в алфавитном порядке. Разные

реализации метода позволяют сравнивать строки и

подстроки с учетом и без учета регистра и

особенностей национального представления дат и т. д.

**CompareOrdinal** Сравнение двух строк по кодам символов. Разные

реализации метода позволяют сравнивать строки и

подстроки

**CompareTo** Сравнение текущего экземпляра строки с другой

строкой

**Concat** Конкатенация строк. Метод допускает сцепление

произвольного числа строк

**Copy** Создание копии строки

**Format** Форматирование в соответствии с заданными

спецификаторами формата

**IndexOf,**

**LastIndexOf,**

**…**

Определение индексов первого и последнего вхождения

заданной подстроки или любого символа из заданного

набора

**Insert** Вставка подстроки в заданную позицию

**Join** Слияние массива строк в единую строку. Между

элементами массива вставляются разделители (см.

далее)

**Length** Длина строки (количество символов)

**Remove** Удаление подстроки из заданной позиции

**Replace** Замена всех вхождений заданной подстроки или символа

новой подстрокой или символом

**Split** Разделение строки на элементы, используя заданные

разделители. Результаты помещаются в массив строк

**Substring** Выделение подстроки, начиная с заданной позиции

28. В чем отличие пустой и null строки?

**Пустая строка** — экземпляр объекта System.String, содержащий 0

символов:

string s = "";

Для пустых строк можно вызывать методы.

► Строки со значениями **null** не ссылаются на экземпляр объекта

System.String, попытка вызвать метод для строки **null** вызовет

исключение NullReferenceException. строки **null** можно использовать в

операциях объединения и сравнения с другими строками.

29. Как можно выполнить сравнение строк?

**Compare, CompareOrdinal, CompareTo,** <,>, >=,<= - сравнивают ссылки!

30. В чем отличие типов String и StringBuilder?

Данные, хранящиеся в классе System.String и есть неизменяемые (immutable). Класс System.StringBuilder разрабатывался так, чтобы над изменяемой строкой можно было проделать множество операций. То есть при каждой операции над объектом класса System.String происходит перенос данных в новую область памяти, что влияет на производительность программы.

31. Поясните явные преобразования переменных с помощью команд Convert.

32. Как выполнить консольный ввод/вывод?

Redline, writeline

33. Приведите примеры определения и инициализации одномерных и двумерных массивов.

Одномерные массивы

(single-dimensional)

тип[] имя;

тип[] имя = new тип [ размерность ];

тип[] имя = { список инициализаторов };

тип[] имя = new тип [] { список инициализаторов };

тип[] имя = new тип [ размерность ] {список инициализаторов };

Прямоугольные (многомерные)

массивы

тип[,] имя;

тип[,] имя = new тип [ разм\_1, разм\_2 ];

тип[,] имя = { список инициализаторов };

тип[,] имя = new тип [,] { список инициализаторов };

тип[,] имя = new тип [ разм\_1, разм\_2 ] { список инициализаторов };

34. Что такое ступенчатый массив? Как его задать?

Ступенчатые,зубчатые jagged массивы

тип[][] имя;

int[][] a = { new int[3], new int[5], new int[4] };

foreach (int[] x in a)

{

foreach (int b in x)

Console.Write("\t" + b);

Console.WriteLine();

}



35. Какие типы можно использовать в foreach? Приведите пример.

Цикл foreach предназначен для перебора элементов в контейнерах, в том числе в массивах.

foreach (тип\_иден. название\_иден. in контейнер)

{ операторы }

int[] numbers = new int[] { 1, 2, 3, 4, 5 };

foreach (int i in numbers)

{

Console.WriteLine(i);

}

36. Что такое кортеж? Для чего и как он используется?

**Кортежи (tuple)** комбинируют объекты различных типов (от одного до восьми).

создается один раз и остается неименным

(все свойства доступны только для

чтения)

►позволяют использовать методы

CompareTo, Equals, GetHashCode и

ToString, свойство Size

►реализуют интерфейсы

IStructuralEquatable, IStructuralComparable

и IComparable (можно сравнивать)

37. Что такое локальная функция?

вспомогательная функция – внутри метода, в котором вызывается. Аргументы внешнего метода и его локальные переменные доступны для локальной функции.