**Вопрос 1** **Когда вызывается статический конструктор статического класса**

при вызове метода, в котором используется статический класс

никогда

перед созданием всех объектов нестатических классов в вызывающем методе

при запуске программы

перед первым обращением к классу в программе

при создании объекта, в котором используется статический класс

**Ответ:** перед первым обращением к классу в программе

**Вопрос 2** **Допустима ли операция для маccива чисел arr и почему**

**foreach(int d in arr)**

**d = 455;**

не допустима, т.к. синтаксис цикла foreach в примере ошибочный

не допустима, т.к. нет фигурных скобок

не допустима, т.к. изменять элементы массива в цикле foreach нельзя

допустима, т.к. переполнения переменной d не произойдет

допустима, т.к. синтаксис языка это позволяет

допустима, т.к. можно увеличить значение элемента массива

**Ответ:** не допустима, т.к. изменять элементы массива в цикле foreach нельзя

**Вопрос 3** **Можно ли изменить switch так, чтобы перейти от case 2 к default, не удаляя break (сам switch на if или что-то другое менять нельзя).**

**int v=2;**

**//...**

**switch(v){**

**case 1:{**

**Console.WriteLine("case 1");**

**break;**

**}**

**case 2:{**

**Console.WriteLine("case 2");**

**break;**

**}**

**default:{**

**Console.WriteLine("default");**

**break;**

**}**

**}**

можно так: в case 2 перед break написать "goto id;". А после "default:{" написать "id:", учитывая, что метка "id" уникальна (все без кавычек)

нельзя, т.к. использование оператора goto не допустимо в switch

нельзя, т.к. в C# нет оператора goto

нельзя, т.к. переходить от одной ветви switch к другой запрещено вравилами языка

можно так: в case 2 перед break написать "goto default;" (без кавычек)

нельзя, т.к. используется оператор области видимости

**Ответ:** можно так: в case 2 перед break написать "goto default;" (без кавычек)

**Вопрос 4** **Выберите правильный вариант получения символа из строки st**

st(0);

st[0];

st.Get(0);

st.get(0);

st.CharAt(0);

st.GetChar(0);

**Ответ:** st[0]

**Вопрос 5** **Как происходит исполнение .NET приложений**

исполняются CLI

исполняются java-машиной

исполняются Common Language Runtime

исполняются как и приложения на C++

исполняются WinRT

исполняются на Chromium

**Ответ:** исполняются Common Language Runtime

**Вопрос 6** **Какие типы данных по способу передачи входят в стандартную систему типов**

типы-указатели

логические типы

символьные типы

числовые типы

типы-значения

типы-ссылки

**Ответ:** типы-значения   
типы-ссылки

**Вопрос 7** **Какое значение по-умолчанию задается для элемента числового массива**

случайное значение

null

nil

ноль

минимальное значение для определенного типа (например, int.MinValue)

мусор из памяти

**Ответ:** ноль

**Вопрос 8** **Какой спецификатор доступа отсутствует в C#**

protected internal

internal

public

private

public internal

protected

**Ответ:** public internal

**Вопрос 9** **Что покажет компилятор или консоль, при условии, что остальной код соответствует стандарту С#**

**String str = "Africa";**

**string str2 = "africa";**

**if (str.Equals(str2, StringComparison.CurrentCultureIgnoreCase))**

**{**

**Console.WriteLine("equals");**

**}**

**else**

**Console.WriteLine("not equals");**

Ошибочный синтаксис для else

Ошибочный синтаксис для Equals

equals

Ошибочный синтаксис для объявления строки str2

Ошибочный синтаксис для объявления строки str

not equals

**Ответ:** equals

**Вопрос 10** **Что получается при первом построении Microsoft.NET приложения?**

исполняемый код на языке C++

go-code

HTML-код

код на MSIL

исполняемый код на языке C#

байт-код

**Ответ:** код на MSIL

**Вопрос 11** **Какая модель типизации используется в языке С#**

отсутствие типизации

динамическая типизация

нестрогая

строгая

ограниченная типизация

нестрогая и строгая одновременно

**Ответ:** строгая

**Вопрос 12** **Как называется среда исполнения приложений .NET**

NET.Framework

IDE

JDK

JVM

CLR

go runtime

**Ответ:** CLR

**Вопрос 13** **Что покажет компилятор или консоль**

**class Counter{**

**public static int instances;**

**public Counter(){**

**++instances;**

**}**

**}**

**public class Program{**

**static void Main(String []args){**

**for(int i=0; i<10; ++i) new Counter();**

**Console.Write(Counter.instances);**

**}**

**}**

неверный синтаксис для Counter.instances – нельзя так вызвать поле класса

неверный синтаксис для for – нет фигурных скобок

неверный синтаксис для for – нет объявления переменной для Counter

ошибка синтаксиса — переменная instances не инициализирована

10

неверный синтаксис — необходимо писать static publiс int instances;

**Ответ:** 10

**Вопрос 14** **Можно ли создать собственный язык программирования, который будет исполняться на платформе .NET**

нельзя, - можно пользоваться только предоставленными языками. Этого не предусматривалось

можно, однако он будет исполняться только на ОС, на которой был создан

можно, однако у него будут ограниченные возможности, по сравнению с существующими языками .NET

можно создать любой язык с какими угодно типами и правилами

нельзя, так как механизм добавления нового языка в среду исполнения еще не добавлен разработчиками

можно создать свой язык, используя спецификацию Common Language Specification (CLS)

**Ответ:** можно создать свой язык, используя спецификацию Common Language Specification (CLS)

**Вопрос 15** **Есть ли разница между конкатенацией строк через плюс и использованием метода String.Format(...), и почему**

разница есть. Она заключается в скорости конкатенации. Метод String.Format(...) работает быстрее

разница есть. Если объединять более 2х строк, то для каждой их пары выделяется память на новую строку. При объединении новой строки со следующей, первая становится мусором. Таким образом, при сложении с помощью плюса, может образоваться много мусора. Мусора выделяется гораздо меньше при использовании String.Format(...)

нет разницы, памяти выделяется одинаково

разница есть. Она заключается в скорости конкатенации. Через плюс конкатенация происходит быстрее

нет разницы, оба варианта работают с одинаковой скоростью

разница есть. Для каждой их пары выделяется память на новую строку. При объединении новой строки со следующей, первая становится мусором. Таким образом, при сложении с помощью String.Format(...), может образоваться много мусора. Мусора выделяется гораздо меньше при использовании плюс

**Ответ:** разница есть. Если объединять более 2х строк, то для каждой их пары выделяется память на новую строку. При объединении новой строки со следующей, первая становится мусором. Таким образом, при сложении с помощью плюса, может образоваться много мусора. Мусора выделяется гораздо меньше при использовании String.Format(...)

**Вопрос 16** **В каком пространстве имен C# реализованы такие базовые типы как String, Boolean, Long**

System

System.Data

System.Types

System.Base

Types

System.Collections

**Ответ:** System

**Вопрос 17** **Как называется основная библиотека классов на .NET**

java.lang

Base Class Library

FCL

CLI

stl

std

**Ответ:** Base Class Library   
FCL

**Вопрос 18** **Почему .NET-приложение переводится компилятором на MSIL, а не в бинарный исполняемый файл**

такой подход позволяет при первом запуске скомпилировать приложение для каждой ЭВМ с учетом ее архитектуры

бинарный файл труднее прочитать среде исполнения

.NET-приложение переводится компилятором на байт-код

потому, что так оно будет работать быстрее

потому что так оно не может быть заражено вирусом

такой подход позволяет уменьшить размер файла

**Ответ:** такой подход позволяет при первом запуске скомпилировать приложение для каждой ЭВМ с учетом ее архитектуры

**Вопрос 19** **Какой синтаксис является правильным**

String[]ar = new String(10);

String ar[10];

String []ar;

String ar = new String[10];

String ar[];

String[10]ar;

**Ответ:** String []ar

**Вопрос 20** **Что покажет компилятор или консоль**

**class A{**

**static private A inst;**

**A(){}**

**public static A getInstance(){**

**if (inst == null) inst = new A();**

**return inst;**

**}**

**}**

**static void Main(){**

**A a1 = A.getInstance();**

**A a2 = A.getInstance();**

**Console.WriteLine(a1==a2? "equals":"not equals");**

**}**

ошибка синтаксиса для поля inst

not equals

ошибка — для Main пропущены параметры

ошибка — нельзя создавать объект класса внутри него же

ошибка при работе программы

equals

**Ответ:** equals

**Вопрос 21** **Каким образом происходит освобождение выделенной памяти на языке С#**

первый и второй вариант вместе

в деструкторе

с помощью оператора delete

в методе Finalize()

c помощью метода Dispose()

используется автоматическая сборка мусора

**Ответ:** используется автоматическая сборка мусора

**Вопрос 22** **Что покажет компилятор или консоль, при условии, что остальной код соответствует стандарту С#**

**string s = "123456789";**

**s.Remove(4);**

**Console.Write(s);**

56789

метод Remove принимает только строки

неверный синтаксис для string

1234

среди перечисленных нет правильного ответа

123456789

**Ответ:** 123456789

**Вопрос 23** **Вызывается ли статический метод быстрее, чем нестатический, и почему**

нет, т.к. вызов статического метода генерирует инструкцию вызова на MSIL

да, т.к. статический класс занимает меньше памяти

да, т.к. не добавляется проверка на ссылки со значением null

нет, т.к. вызов статического метода генерирует инструкцию callvirt

нет, т.к. он относится к классу, а не объекту класса

да, т.к. статический класс создается сразу при старте программы

**Ответ:** да, т.к. не добавляется проверка на ссылки со значением null

**Вопрос 24** **Что покажет компиллятор или консоль**

**string s = "lassen";**

**string s1 = "la\u00dfen";**

**Console.WriteLine(**

**s.CompareTo(s1) == 0 ?**

**"equals" : "not equals");**

Неверный синтаксис объявления строки s1

Неверный синтаксис для параметров Console.WriteLine

equals

Ошибка на этапе исполнения

CompareTo возвращает bool

not equals

**Ответ:** equals

**Вопрос 25** **В чем основное отличие классов String и StringBuilder**

в С# нет класса StringBuilder

объекты класса String изменяемы, а объекты класса StringBuilder - нет

класс StringBuilder имеет больше методов для работы со строками

класс StringBuilder – это новая версия класса String

объекты класса StringBuilder изменяемы, а объекты класса String - нет

методы объектов класса StringBuilder синхронизированы, а методы объектов класса String – нет

**Ответ:** объекты класса StringBuilder изменяемы, а объекты класса String - нет

**Вопрос 26** **Что покажет компиллятор или консоль**

**Console.WriteLine(**

**String.Format("Today {0:d} at {0:t} my program executed this code", DateTime.Now));**

Today {0:d} at {0:t} my program executed this code

Today my program executed this code

Консоль выведет сообщение с текущей датой и временем устройства

Компиллятор укажет, что произошла ошибка

Консоль выведет сообщение с датой и временем компилляции программы

При исполнении программы произойдет ошибка

**Ответ:** Консоль выведет сообщение с текущей датой и временем устройства

**Вопрос 27** **Выберите правильный синтаксис получения длины 2го измерения массива**

int len = ar.Length(1);

int len = ar.GetLength(1);

int len = ar.GetLength[2];

int len = ar.length[1];

int len = ar.getLength(2);

int len = ar.Length[1];

**Ответ:** int len = ar.GetLength(1)

**Вопрос 28** **Выберите правильный вариант объявления класса**

class A{ public int x; }

class A{ public int x = 0; }

class A{ public: int x; }

class A{ public: int x=0; }

class A{ public int x(0); }

class A{ public: int x(0); }

**Ответ:** class A{ public int x; }   
class A{ public int x = 0; }

**Вопрос 29** **Каким значением по-умолчанию инициализируются элементы рваного (разряженного) массива**

ноль

мусор из памяти

nil

случайное значение

null

минимальное значение для определенного типа (например int.MinValue)

**Ответ:** null

**Вопрос 30** **Что покажет компилятор или консоль**

**class StringPair{**

**public readonly string value1, value2;**

**public StringPair(string value1\_, string value2\_)**

**{**

**init(value1\_, value2\_);**

**}**

**private void init(string value1\_, string value2\_){**

**value1 = value1\_;**

**value2 = value2\_;**

**}**

**}**

**//код в Main**

**StringPair p = new StringPair("Hello", "World!");**

**Console.Write(p.value1 + "\_" + p.value2);**

неверный синтаксис при объявлении конструктора

выдаст сообщение об ошибке при исполнении программы

нельзя изменять значение публичной переменной в частном методе

не найдено ключевое слово readonly

"Hello\_World!"

присваивание значения переменной, доступной только для чтения, допускается только в конструкторе

**Ответ:** присваивание значения переменной, доступной только для чтения, допускается только в конструкторе