1.Назовите принципы ООП. Поясните каждый из них.

а)наследование – процесс, благодаря которому один объект может наследовать(приобретать) свойства от других объектов

б)инкапсуляция-механизм, связывающий вместе данные и код, обрабатывающий эти данные, и сохраняющий их от внешнего воздействия и ошибочного использования

в)полиморфизм-способность вызвать метод потомка через экземпляр предка

г)абстракция данных- разделение и независимое рассмотрение интерфейса и реализации

абстракция-уровень описание/представления модели чего либо

2.

Назовите класс .NET, от которого наследуются все классы.

System.Object

3.

Охарактеризуйте открытые методы System.Object

|  |  |
| --- | --- |
| bool Equals() | Сравнивает две ссылки на объекты в период выполнения, чтобы определить, указывают ли они в точности один и тот же объект. Если две переменные ссылаются на один и тот же объект, возвращается true. В случае размерных типов (см. о них следующий раздел) этот метод возвращает true, если типы переменных идентичны и их значения равны. |
| int GetHashCodeO | Возвращает заданный для объекта хзш-код. Хэш-функции используются в реализации класса, когда хэш-код объекта нужно поместить в хэш-таблицу для повышения производительности. |
| Type GetType() | Используется с методами отражения (см. о них главу 16) для получения информации о типе данного объекта. |
| string ToString | Используется по умолчанию для получения имени объекта. Его можно переопределить в производных классах, чтобы они возвращали понятное пользователю текстовое представление объекта. |

4.Охарактеризуйте закрытые методы System.Object.

|  |  |
| --- | --- |
| Метод | Описание |
| void Finalize() | Вызывается в период выполнения для освобождение ресурсов перед сбором мусора. Этот метод можно вызывать, а можно и не делать этого. Поэтому не помещайте в него подлежащий исполнению код. Это правило выливается в нечто под названием детерминированное завершение (deterministic finalization), о котором подробнее см. главу 5. |
| Object MemberwiseClone | Представляет ограниченную копию (shallow copy) объекта. Под этим я понимаю копию объекта, содержащую ссылки на другие объекты, но не копии этих объектов. Если ваши классы должны поддерживать полную копию (deep copy), которая действительно включает копии объектов, на которые она ссылается, то вам нужно реализовать интерфейс ICloneable и самому вручную производить клонирование или копирование. |

5.Приведите пример определения класса.

Сlass имя\_класса

{

//члены \_класса

}

6.Какие ключевые слова можно использовать при определении класса?

- public –общедоступный член класса.

-private–член класса доступен только внутри данного класса.

-protected–член класса доступен только внутри данного класса и внутри

классов, производных от данного.

-internal–член класса доступен только внутри данной сборки (программы)???????????

7.В чем отличие между объектом и классом?

Класс это абстрактный тип данных, определяемый (программистом)

Объекты это экземпляры класса.

8.Что такое конструктор? Когда вызывается конструктор?

Конструктор–это метод класса, имеющий имя класса. Конструкторов

в классе может быть несколько или ни одного.

Каждый раз, когда создается класс или структура, вызывается конструктор

9.Перечислите свойства конструктора?

1) не имеет возвращаемого значения

2)имя такое же как и имя типа(класса)

3)не наследуются

4)нельзя применить модификаторы virtual, new, override, sealed и abstract

5)для класса без явно заданных конструкторов компилятор создает конструктор по умолчанию(без параметров)

6) для статистических классов компилятор не создает конструктор по умолчанию

7) можно определить несколько конструкторов, сигнатуры и уровня доступа к конструкторам обязательно должны отличаться

8)можно явно заставлять однин конструктор вызывать другой конструктор посрдством зарезервированного слова this

10.Что такое деструктор (destructor) ?

В языке C# имеется возможность определить метод, который будет вызываться непосредственно перед окончательным уничтожением объекта системой "сборки мусора". Такой метод называется **деструктором** и может использоваться в ряде особых случаев, чтобы гарантировать четкое окончание срока действия объекта.

11.Что такое this?

Обеспечивает доступ к текущему экземпляру класса. Ключевое слово this ссылается на текущий экземпляр класса, а также используется в качестве модификатора первого параметра метода расширения.

12.Что будет выведено в результате выполнения

class

A

{

private

int

\_num;

public

A(

int

num) { Num = num; }

public

int

Num {

get

{

return

\_num; }

set

{ \_num =

value

; } }

}

static

void

Main(

string

[] args)

{

A

a =

new

A

(5);

A

b = a;

Console

.WriteLine(a.Num +

" "

+b.Num);

a.Num = 7;

Console

.WriteLine(a.Num +

" "

+ b.Num);

5 5

7 5

13.Какие спецификаторы доступа для класса и методов класса существуют в C#?

public: публичный, общедоступный класс или член класса. Такой член класса доступен из любого места в коде, а также из других программ и сборок.

private: закрытый класс или член класса. Представляет полную противоположность модификатору public. Такой закрытый класс или член класса доступен только из кода в том же классе или контексте.

protected: такой член класса доступен из любого места в текущем классе или в производных классах.

internal: класс и члены класса с подобным модификатором доступны из любого места кода в той же сборке, однако он недоступен для других программ и сборок (как в случае с модификатором public).

protected internal: совмещает функционал двух модификаторов. Классы и члены класса с таким модификатором доступны из текущей сборки и из производных классов.

14.Опишите модификатор protected internal.

protected internal: совмещает функционал двух модификаторов. Классы и члены класса с таким модификатором доступны из текущей сборки и из производных классов.

15.Зачем и как используются ref и out параметры функции?

Параметрыпредставляют собой переменные, которые определяются в сигнатуре метода и создаются при его вызове.

При передаче параметров по ссылке перед параметрами используется модификатор ref

Cделать параметр выходным, перед ним ставится модификатор out

16.Приведите пример необязательных и именованных параметров метода.

**Необязательные параметры**

C# позволяет использовать необязательные параметры. Для таких параметров нам необходимо объявить значение по умолчанию. Также следует учитывать, что после необязательных параметров все последующие параметры также должны быть необязательными:

|  |  |
| --- | --- |
| 1  2  3  4 | static int OptionalParam(int x, int y, int z=5, int s=4)  {      return x + y + z + s;  } |

Так как последние два параметра объявлены как необязательные, то мы можем один из них или оба опустить:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | static void Main(string[] args)  {      OptionalParam(2, 3);        OptionalParam(2,3,10);        Console.ReadLine();  } |

**Именованные параметры**

В предыдущих примерах при вызове методов значения для параметров передавались в порядке объявления этих параметров в методе. Но мы можем нарушить подобный порядок, используя именованные параметры:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | static int OptionalParam(int x, int y, int z=5, int s=4)  {      return x + y + z + s;  }  static void Main(string[] args)  {      OptionalParam(x:2, y:3);        //Необязательный параметр z использует значение по умолчанию      OptionalParam(y:2,x:3,s:10);        Console.ReadLine();  } |

17.Приведите пример полей класса –статические, константные, только для чтения.

const double PI = 3.14;

readonly double PI = 3.14;

static double PI = 3.14;

18.Приведите пример определения свойств класса.

Как свойства связаны с инкапуляцией?

Private int m=35;

Public int y

{

get

{

return m;

}

set

{

m = value;

}

Свойства инкапсуляции :

1)Совместное хранение данных в функции

2)сокрытие внутренней информации от пользователя

3)изоляция пользователя от особенностей реализации

19.Назовите явное имя параметра, передаваемого в метод set свойства класса?

??????

20.Что такое автоматические свойства?

Свойства управляют доступом к полям класса. Однако что, если у нас с десяток и более полей, то определять каждое поле и писать для него однотипное свойство было бы утомительно. Поэтому в фреймворк .NET были добавлены автоматические свойства.

21.Что такое индексаторы класса?

Какие ограничения существуют на индексатор?

Позволяет индексировать объект таким же способом, как массив или кллекцию

Индексаторы можно перегружать

Ограничения на индексаторы:

1)значение , выдаваемое индексатором, нельзя предавать методу в качестве параметра ref или out

2)индексатор не может быть объявлен как static

22.Что такое перегруженный метод?

Иногда возникает необходимость создать один и тот же метод, но с разным набором параметров. И в зависимости от имеющихся параметров применять определенную версию метода. Такая возможность еше называется перегрузкой методов

23.Что такое partial класс и какие его преимущества?

24.Что такое анонимный тип в C#?

Анонимные типы позволяют создать объект с некоторым набором свойств без определения класса. Анонимный тип определяется с помощью ключевого слова var и инициализатора объектов:

|  |  |
| --- | --- |
| 1  2 | var user = new { Name = "Tom", Age = 34 };  Console.WriteLine(user.Name); |

В данном случае user - это объект анонимного типа, у которого определены два свойства Name и Age. И мы также можем использовать его свойства, как и у обычных объектов классов. Однако тут есть ограничение - свойства анонимных типов доступны только для чтения.

25.Для чего делают статические классы?

Ранее, чтобы использовать какой-нибудь класс, устанавливать и получать его поля, использовать его методы, мы создавали его объект. Однако если данный класс имеет статические методы, то, чтобы получить к ним доступ, необязательно создавать объект этого класса.

26.В чем отличие статического поля от экземплярного?

Статический метод - вызывается на уровне типа (вызов: Имя\_типа точка Имя\_метода)  
Экземплярный метод - вызывается на уровне объявленного экземпляра типа (вызов: Имя\_переменной точка Имя\_метода).

27.Поясните работу статических конструкторов.

татические конструкторы выполняются при самом первом создании объекта данного класса или первом обращении к его статическим членам (если таковые имеются)

28.Какая разница между поверхностным (shallow) и глубоким (deep)

копированием?

При поверхностном копировании копируются значения полей класса, включая значения любых указателей или ссылок. При этом скопированные значения этих указателей и ссылок указывают на одни и те же объекты, что и в оригинальном объекте, что зачастую ведет к ошибкам. Отсюда и название такого метода копирования: мы копируем только указатели/ссылки, вместо того, чтобы делать копии этих внутренних объектов и ссылаться на них, собственно не углубляемся во внутреннюю структуру объекта. При глубоком копировании мы копируем значения полей не только на первом "уровне", но и заходим глубже, копируя все значения.

29.В чем разница между равенством и тождеством объектов?

Равенство — операция, позволяющая определить совпадают или различаются [объекты](https://dic.academic.ru/dic.nsf/ruwiki/73197).

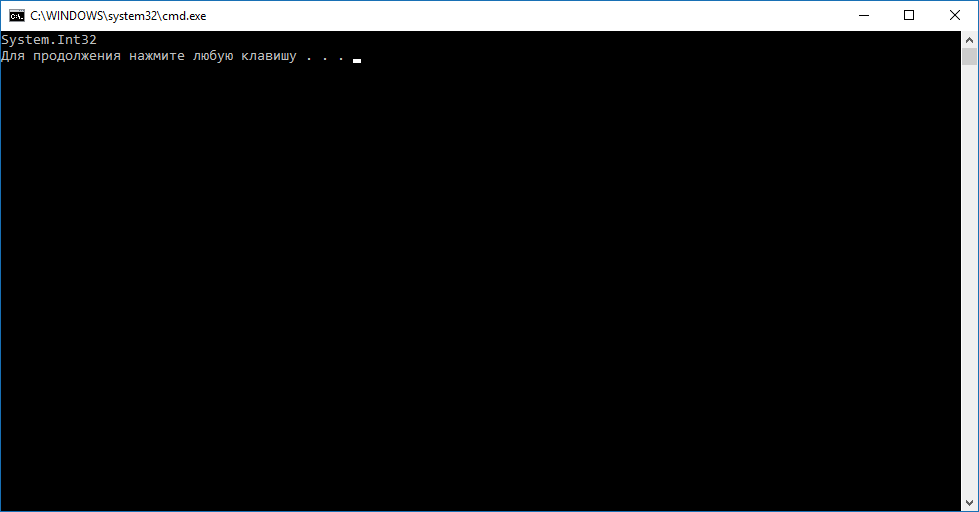
При тождестве левая и правая части равны

30.Что такое частичные классы и частичные методы?

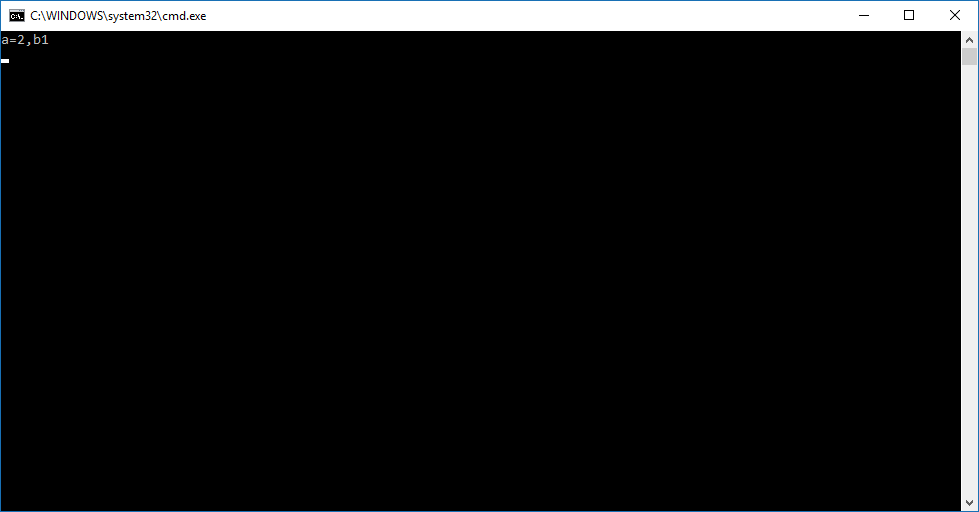
Частичные классы (partial class) представляют возможность разделить функционал одного класса на несколько файлов.

Частичные методы обычно используются кодогенераторами. Они объявляются в автоматически сгенерированной части кода и оттуда же вызваются, а в части кода, которая пишется вручную, их можно реализовать. Если метод не реализован, при компиляции его вызовы автоматически удаляются из кода. Соответственно, сигнатура этих методов должна быть такой, чтобы их удаление никак не повлияло на собираемость кода. Отсюда описанные ограничения.

31.Что будет выведено на консоль результате выполнения следующего кода:



32.Что будет выведено на консоль результате выполнения следующего кода:



33.Пусть задан следующий класс.

Какой из конструкторов задан неверно?

2) public int A() { }

34.Пусть задан следующий класс.

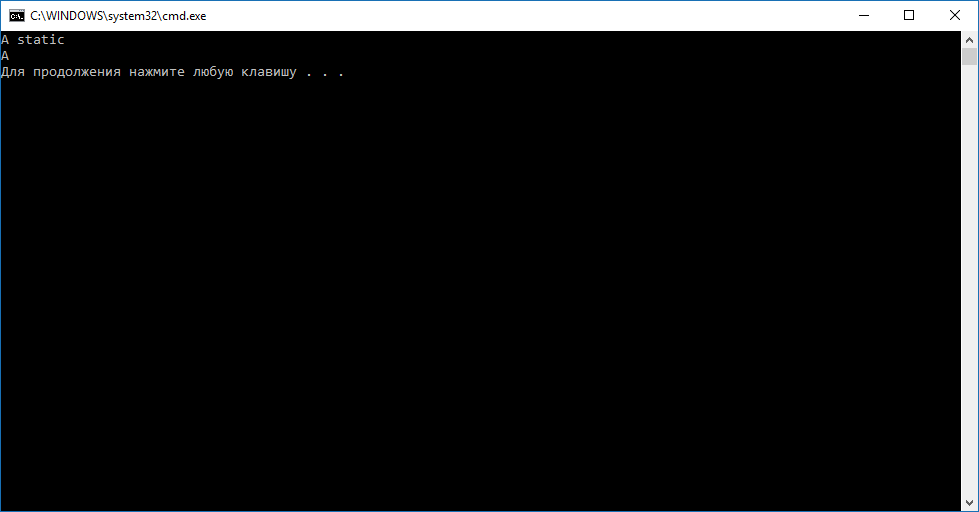
Сколько аргументов может быть задано при вызове конструктора

данного класса?

35.Почему не удается создать объект класса A?

Из за уровня защиты

36.Что будет выведено в консоль при выполнении данной программы?



37.Какая строка приведенного далее класса вызовет ошибку компиляции?

3) b=30 Присваивание значений доступному только для чтения статическому полю допускается только в статическом конструкторе и в инициализаторе переменных.