Задачи с созданием классов(MathUtils, Counter, StringUtils, Circle)

Вопрос 1  
Почему следующая программа не компилируется:  
  
using System;  
  
namespace HelloApp  
{  
class Program  
{  
static void Main(string[] args)  
{  
Person tom = new Employee();  
Console.ReadKey();  
}  
}  
  
internal class Person  
{  
  
}  
public class Employee : Person  
{  
  
}  
}

Ответ:

Ошибка возникает из-за того, что класс Person имеет модификатор доступа internal, что означает, что он доступен только в пределах той же сборки (assembly). Однако, вы пытаетесь создать экземпляр класса Person в другом классе, который по умолчанию находится в другой сборке. Чтобы программа скомпилировалась, нужно изменить модификатор доступа Person на public, чтобы он был доступен и за пределами своей сборки.

Вопрос 2  
Даны следующие классы:  
  
class Person  
{  
string name;  
int age;  
  
public Person()  
{  
}  
public Person(string name) : this(name, 18)  
{  
}  
public Person(string name, int age)  
{  
[this.name](https://web.telegram.org/a/this.name) = name;  
this.age = age;  
}  
}  
class Employee : Person  
{  
string company;  
  
public Employee()  
{  
}  
public Employee(string name, int age, string company): base(name, age)  
{  
[this.company](https://web.telegram.org/a/this.company) = company;  
}  
public Employee(string name, string company) : base(name)  
{  
[this.company](https://web.telegram.org/a/this.company) = company;  
}  
}  
Допустим, мы создаем объект класса Employee следующим образом:  
  
1  
Employee tom = new Employee("Tom", "Microsoft");

Какие конструкторы и в каком порядке в данном случае будет выполняться?

В данном случае будет выполнено следующее:

Сначала вызывается конструктор Employee(string name, string company), который вызывает базовый конструктор Person(string name).

Конструктор Person(string name) вызывает конструктор Person() через this(name, 18) и передает имя "Tom", а возраст устанавливается равным 18 по умолчанию.

В конструкторе Employee будет установлен параметр company = "Microsoft". Итак, последовательность будет такой:

Вызов конструктора Employee(string name, string company)

Вызов конструктора Person(string name)

Вызов конструктора Person()  
  
  
Вопрос 3  
Как запретить наследование от класса?

Чтобы запретить наследование от класса в C#, необходимо использовать модификатор sealed. Класс, помеченный как sealed, не может быть наследован.  
  
Вопрос 4  
Что выведет на консоль следующая программа и почему?

class Auto // легковой автомобиль  
{  
public int Seats { get; set; } // количество сидений  
public Auto(int seats)  
{  
Seats = seats;  
}  
}  
class Truck : Auto // грузовой автомобиль  
{  
public decimal Capacity { get; set; } // грузоподъемность  
public Truck(int seats, decimal capacity)  
{  
Seats = seats;  
Capacity = capacity;  
}  
}  
class Program  
{   
static void Main(string[] args)  
{  
Truck truck = new Truck(2, 1.1m);  
Console.WriteLine($"Грузовик с грузоподъемностью {truck.Capacity} тонн");  
Console.ReadKey();  
}  
}

Ответ:

Auto has been created

Truck has been created

Truck with capacity 1.1

Это происходит потому, что в конструкторе Truck(decimal capacity) вызывается конструктор Auto(), и сначала выполняется вывод "Auto has been created", а затем выводится "Truck has been created". После этого выводится информация о грузоподъемности.

Вопрос 5  
Что выведет на консоль следующая программа и почему?  
  
  
class Auto // легковой автомобиль  
{  
public int Seats { get; set; } // количество сидений  
public Auto()  
{  
Console.WriteLine("Auto has been created");  
}  
public Auto(int seats)  
{  
Seats = seats;  
}  
}  
class Truck : Auto // грузовой автомобиль  
{  
public decimal Capacity { get; set; } // грузоподъемность  
public Truck(decimal capacity)  
{  
Seats = 2;  
Capacity = capacity;  
Console.WriteLine("Truck has been created");  
}  
}  
class Program  
{   
static void Main(string[] args)  
{  
Truck truck = new Truck(1.1m);  
Console.WriteLine($"Truck with capacity {truck.Capacity}");  
Console.ReadKey();  
}  
}

Ответ: Sam

В Person конструктор всегда присваивает Name = "Tim".

При создании Employee, Name сначала становится "Tim".

Затем используется инициализатор { Name = "Sam" }, который перезаписывает значение.

Вопрос 6  
Что выведет на консоль следующая программа и почему?  
  
class Person  
{  
public string Name { get; set; } = "Ben";  
  
public Person(string name)  
{  
Name = "Tim";  
}  
}  
  
class Employee : Person  
{  
public string Company { get; set; }  
  
public Employee(string name, string company)  
: base("Bob")  
{  
Company = company;  
}  
}  
  
class Program  
{   
static void Main(string[] args)  
{  
Employee emp = new Employee("Tom", "Microsoft") { Name = "Sam" };  
  
Console.WriteLine(emp.Name); // Ben Tim Bob Tom Sam  
Console.ReadKey();  
}  
}

Ответ:

Tim

Это происходит по следующим причинам:

Конструктор Employee(string name, string company) вызывает базовый конструктор Person("Bob"), и значение Name в родительском классе будет установлено в "Bob".

Внутри конструктора Employee значение Name будет переопределено в "Sam".

Однако, поскольку конструктор Person присваивает Name в "Tim", это значение будет записано в Name, и результат будет "Tim", а не "Sam".