1. Что понимается под термином .Net Framework?
2. Зависят ли приложения, разрабатываемые в .NET от платформы?
3. Возможно ли создание гетерогенных приложений в среде .NET? (ответ аргументировать)
4. Что означает аббревиатура CLR? (пояснить значение). Является ли среда CLR многоязычной?
5. Что такое принципы SOLID? Объясните значение каждого из принципов.
6. Объясните принцип подстановки Барбары Лисков? (приведите примеры)
7. Объясните принцип Inversion Of Control? (приведите примеры)
8. Объясните принцип Dependency Injection? (приведите примеры)
9. Приведите обобщенный синтаксис объявления переменной на языке C#.
10. Приведите обобщенный синтаксис инициализации переменной на языке C#.
11. Каковы основные категории типов в языке C# (приведите примеры для каждой категории).
12. Что понимается под областью видимости переменной в языке C#?
13. Как обозначается область видимости переменной в языке C#?
14. Как соотносится время жизни переменной и область видимости?
15. Приведите синтаксис оператора условного перехода в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
16. Приведите синтаксис оператора switch в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
17. Что понимается под термином «пространство имен»?
18. В чем состоит назначение пространств имен в языке C#?
19. Благодаря какому механизму удается избежать коллизий имен в языке C#?
20. Какое пространство имен использует системная библиотека .NET Framework?
21. В чем состоит назначение директивы using?
22. Приведите синтаксис директивы using в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
23. Приведите синтаксис описания пространства имен в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
24. Что понимается под термином «класс»?
25. Какие элементы определяются в составе класса?
26. Каково соотношение понятий «класс» и «объект»?
27. Что понимается под термином «члены класса»?
28. Какие члены класса Вам известны? Какие члены класса содержат код? Какие члены класса содержат данные? Перечислите пять разновидностей членов класса специфичных для языка C#.
29. Что понимается под термином «конструктор»? Сколько конструкторов может содержать класс языка C#?
30. Приведите синтаксис описания класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
31. Какие модификаторы типа доступа Вам известны (поясните их значение для способов применения)?
32. Какое ключевое слово языка C# используется при создании объекта (объяснить процесс создания нового объекта)?
33. Приведите синтаксис создания объекта в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
34. В чем состоит назначение конструктора? Каждый ли класс языка C# имеет конструктор (привести примеры)?
35. Каким значением инициализируются по умолчанию значения ссылочного типа?
36. В каком случае конструктор по умолчанию не используется?
37. Приведите синтаксис конструктора класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
38. Что понимается под термином «деструктор» (объяснить назначение и особенности использования)?
39. Приведите синтаксис деструктора класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#. Имеет ли деструктор параметры? Возвращает ли деструктор значение?
40. Что понимается под термином «наследование»? Какая классификация объектов соответствует наследованию?
41. Что общего имеет дочерний класс с родительским? В чем состоит различие между дочерним и родительским классами?
42. Приведите синтаксис описания наследования классов в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
43. Что понимается под термином «полиморфизм»?
44. В чем состоит основной принцип полиморфизма?
45. В чем состоит значение основного принципа полиморфизма?
46. Какие механизмы используются в языке C# для реализации концепции полиморфизма?
47. Что понимается под термином «виртуальный метод»? (привести примеры кода)
48. Какое ключевое слово языка C# используется для определения виртуального метода? В чем состоит особенность виртуальных методов в производных (дочерних) классах?
49. Какие условия определяют выбор версии виртуального метода?
50. Какое ключевое слово (модификатор) языка C# используется для определения виртуального метода в базовом (родительском) классе? Какое ключевое слово (модификатор) языка C# используется для определения виртуального метода в производном (дочернем) классе? Какие модификаторы недопустимы для определения виртуальных методов?
51. Что означает термин «переопределенный метод»?
52. Приведите синтаксис виртуального метода в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
53. Что понимается под термином «абстрактный класс»?
54. В чем заключаются особенности абстрактных классов? Какой модификатор языка C# используется при объявлении абстрактных методов?
55. Являются ли абстрактные методы виртуальными? Используется ли модификатор virtual языка C# при объявлении абстрактных методов?
56. Возможно ли создание иерархии классов посредством абстрактного класса? (привести примеры кода)
57. Возможно ли создание объектов абстрактного класса?
58. Приведите синтаксис абстрактного класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
59. Что понимается под термином «интерфейс»?
60. Чем отличается синтаксис интерфейса от синтаксиса абстрактного класса?
61. Какое ключевое слово языка C# используется для описания интерфейса? Поддерживают ли реализацию методы интерфейса?
62. Какие объекты языка C# могут быть членами интерфейсов?
63. Каким количеством классов может быть реализован интерфейс? Может ли класс реализовывать множественные интерфейсы?
64. Необходима ли реализация методов интерфейса в классе, включающем этот интерфейс? (приведите примеры, в том числе EIMI)
65. Приведите синтаксис интерфейса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
66. Возможно ли создание ссылочной переменной интерфейсного типа?
67. Возможно ли наследование интерфейсов? Насколько синтаксис наследования интерфейсов отличается от синтаксиса наследования классов? Необходимо ли обеспечение реализации в иерархии наследуемых интерфейсов?
68. Что понимается под термином «делегат»?
69. В чем состоят преимущества использования делегатов?
70. В какой момент осуществляется выбор вызываемого метода в случае использования делегатов? Что является значением делегата?
71. Какое ключевое слово языка C# используется для описания делегатов? Приведите синтаксис делегата в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
72. Возможно ли использование делегата для вызова метода, соответствующего описанию делегата? Возможен ли вызов метода в том случае, если его определение не соответствует определению делегата?
73. Что понимается под термином «многоадресный» делегат? В чем состоит практическое значение многоадресности?
74. Каким образом осуществляется создание цепочки методов для многоадресных делегатов?
75. Какие операторы языка C# используются для создания цепочки методов для многоадресных делегатов? Каким образом осуществляется удаление цепочки методов для многоадресных делегатов?
76. Каким должен быть тип возвращаемого значения для многоадресных делегатов и почему?
77. Что понимается под термином «событие»?
78. Являются ли события членами классов?
79. Какое ключевое слово языка C# используется для описания событий? На каком механизме языка C# основана поддержка событий?
80. Приведите синтаксис описания события в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
81. Что понимается под термином «широковещательное событие»? На основе какого механизма языка C# строятся широковещательные события?
82. Приведите синтаксис описания события в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
83. Что понимается под термином «исключительная ситуация (исключение)»?
84. В чем состоит значение механизма исключений в языке C#?
85. Какие операторы языка C# используются для обработки исключений? (приведите примеры)
86. Приведите синтаксис блока try…catch в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
87. Необходимо ли обеспечивать соответствие типов исключения в операторе catch типу перехватываемого исключения?
88. Что происходит в случае неудачного перехвата исключения?
89. В каком случае возможно использование оператора языка C# catch без параметров?
90. Каким образом осуществляется возврат в программу после обработки исключительной ситуации?
91. Какой оператор языка C# используется для обеспечения возврата в программу после обработки исключения?
92. Приведите синтаксис блока finally (в составе оператора try…catch) в общем виде. Проиллюстрируйте его фрагментом программы на языке C#. Зависит ли вызов блока finally от наличия исключения?
93. Какие способы генерации исключений Вам известны?
94. Каким образом возможно осуществить явную генерацию исключений? (привести пример кода)
95. Приведите синтаксис оператора throw в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
96. Возможно ли создавать специализированные исключения для обработки ошибок в коде пользователя? Какой системный класс является базовым для создания исключений?
97. Необходима ли явная реализация классов, наследуемых от системных исключений?
98. Каким образом обеспечивается обращение к свойствам и методам системных исключений?
99. Что такое сборка «мусора», зачем она нужна?
100. К чему приведет операция присвоения если в левой и правой частях ссылки на объект?
101. Что такое инкапсуляция. Приведите примеры
102. Что такое перегрузка методов. Приведите примеры.
103. Дайте определение статического класса. Поясните особенности поведения таких классов. Что такое синглетон.
104. Объясните различие между виртуальными и абстрактными членами класса.
105. Что будет выведено в консоль

var s1 = new string(new[] { 'T', 'e', 's', 't' });

var s2 = new string(new[] { 'T', 'e', 's', 't' });

var o1 = (object)s1;

var o2 = (object)s2;

Console.WriteLine(s1 == s2);

Console.WriteLine(Equals(s1, s2));

Console.WriteLine(o1 == o2);

Console.WriteLine(Equals(o1, o2));

|  |  |  |  |
| --- | --- | --- | --- |
| А. ⃝ True  True  False  False | B. ⃝ True  True  False  True | C. ⃝ True  True  True  True | D. ⃝ True  False  False  False |

1. Напишите потоко-безопасную реализацию события. Также каждый обработчик события не должен зависеть от возможных ошибок (исключений) в других обработчиках.
2. "MyKey" - класс, который является кастомной реализацией ключа для словаря, со следующими членами:

public class MyKey

{

private readonly string \_code;

public string Code { get { return \_code; } }

public MyKey(string code)

{

if (code == null) throw new ArgumentNullException("code");

\_code = code;

}

public override int GetHashCode()

{

return \_code.GetHashCode();

}

}

что будет выведено в консоль при выполнении следующего кода?

var myMap = new Dictionary<MyKey, string>();

var key = new MyKey("KeyA");

myMap.Add(key, "Value1");

key = new MyKey("KeyB");

myMap.Add(key, "Value2");

myMap[key] = "Value3";

key = new MyKey("KeyA");

myMap[key] = "Value3";

myMap.Remove(new MyKey("KeyB"));

Console.WriteLine("size = {0}", myMap.Count);

|  |  |  |
| --- | --- | --- |
| А. ⃝ size = 2 | B. ⃝ size = 3 | С. ⃝ size = 3 |

1. Что будет выведено в консоль в результате выполнения следующей программы?

interface I

{

int P { get; }

}

class A : I

{

virtual public int P { get { return 0; } }

}

class B : A

{

public override int P { get { return 1; } }

}

class C : B, I

{

public int P { get { return 2; } }

}

..........................

A a = new A(), b = new B(), c = new C();

I ia = new A(), ib = new B(), ic = new C();

Console.WriteLine("{0}, {1}, {2}, {3}, {4}, {5}", a.P, b.P, c.P, ia.P, ib.P, ic.P);

|  |  |  |  |
| --- | --- | --- | --- |
| А. ⃝ 0, 1, 2, 0, 1, 2 | B. ⃝ 0, 1, 2, 0, 1, 1 | C. ⃝ 0, 1, 1, 0, 1, 2 | D. ⃝ 0, 1, 1, 0, 1, 1 |

1. Объясните что такое «обобщения» (generics). Приведите примеры обобщенных классов, и членов классов.
2. Объясните практическое значение «обобщений».
3. Что такое «Атрибут». Объясните практическое значение атрибутов.
4. Объясните, что такое значимые типы, допускающие присвоение значения null (nullable types). В чем и практическое назначение.
5. Объясните что такое многопоточность, каковы ее плюсы и минусы.
6. Что такое «поток» (thread), объясните особенности его реализации на платформе .Net
7. Что такое пул потоков (thread pool). Что такое backgroundworker? Приведите примеры использования данных классов.
8. Для чего нужна синхронизация потоков? Какие способы синхронизации потоков вы знаете на платформе .Net, в чем их отличие?
9. Что такое асинхронная модель программирования (APM – asynchronous programming model) на платформе .Net?
10. Что такое асинхронный делегат? Приведите примеры использования.