|  |
| --- |
| , RD Dep. |
| Контрольный тест  NET.C#.11 Коллекции и обобщенные типы |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial version | Анжелика Кравчук |  |  |  |
| 1.1 | Review and corrections. | Владимир Тихон |  |  |  |

1. **Определена переменная c**

Color c = Color.Blue;

**где**

enum Color{ White, Red, Green, Blue, Orange}.

**При выполненеии какого оператора на консоль будет выведено целочисленное значение 3.**

* Console.WriteLine(c);
* Console.WriteLine(c.ToString());
* Console.WriteLine(c.ToString("D"));
* Console.WriteLine(c.ToString("G"));

Сложность: [1]

1. **В перечислении Season**

enum Season : sbyte {Spring = -3, Summer, Fall, Winter};

**литерал Winter равен**

* -1
* 0
* Перечисление не имеет смысла

Сложность: [1]

1. **Если при определении класса не определить конструктор по умолчанию, то**

* Конструктор по умолчанию гинерируется автоматически, только если класс не имеет других конструкторов.
* Конструктор по умолчанию для класса автоматически сгинерируется в любом случае.
* Код не скомпилируется.

Сложность: [1]

1. **Что произойдет, если при определении разделяемого метода, не обеспечить его реализацию?**

* Код не будет компилироваться.
* Код будет компилироваться, но не будет работать.
* Код будет компилироваться и работать, и можно будет вызвать разделяемыйметод.

Сложность: [1]

1. **Результатом выполнения следующего кода будет**

struct Book

{

public string title;

public string author;

public int copyrigth;

public Book()

{

author = "Author";

title = "Title";

copyrigth = 1000;

Console.WriteLine("This is a book.");

}

}

class Program

{

static void Main(string[] args)

{

Book book = new Book();

Console.WriteLine("This is not a book.");

}

}

* This is a book.

This is not a book.

* This is a book.
* Ошибка времени компиляции.
* Ошибка времени выполнения.

Сложность: [1]

1. **Отметьте операторы, которые генерируют идентичный IL код**

* int a = new int();
* int b = 0;
* Int16 с = 0;
* System.Int32 d = new System.Int32();
* Int32 e = 0;
* Byte f = 0;

Сложность: [2]

1. **Если Residence является классом**

public enum ResidenceType

{

House,

Flat,

Bungalow,

Apartment

};

public class Residence

{

public ResidenceType type;

public int numberOfBedrooms;

public Residence(ResidenceType type, int numberOfBedrooms)

{

this.type = type;

this.numberOfBedrooms = numberOfBedrooms;

}

}

**какое сообщение будет выведено на консоль при выполнении следующего кода?**

Residence myHouse = new Residence(ResidenceType.House, 2);

Residence anotherHouse = new Residence(ResidenceType.House, 2);

if (myHouse == anotherHouse)

{

Console.WriteLine("They are the same house");

}

else

{

Console.WriteLine("They are different houses");

}

* They are the same house
* They are different houses
* Код не будет компилироваться

Сложность: [1]

1. **Если Residence является структурой**

public enum ResidenceType

{

House,

Flat,

Bungalow,

Apartment

};

public struct Residence

{

public ResidenceType type;

public int numberOfBedrooms;

public Residence(ResidenceType type, int numberOfBedrooms)

{

this.type = type;

this.numberOfBedrooms = numberOfBedrooms;

}

}

**какое сообщение будет выведено на экран при выполнении следующего кода?**

Residence myHouse = new Residence(ResidenceType.House, 4);

Residence anotherHouse = new Residence(ResidenceType.House, 4);

if (myHouse == anotherHouse)

{

Console.WriteLine("They are the same house");

}

else

{

Console.WriteLine("They are different houses");

}

* They are the same house
* They are different houses
* Код не будет компилироваться

Сложность: [2]

1. **Пусь определена структура**

private struct Point : ICloneable

{

public Int32 x, y;

public override String ToString()

{

return String.Format("({0}, {1})", x, y);

}

public Object Clone()

{

return MemberwiseClone();

}

}

**создан экземпляр структуры**

Point p;

**и инициализированы ее поля**

p.x = 10;

p.y = 20;

Отметьте строки, в кроторых происходит упаковка.

* Console.WriteLine(p.ToString());
* Console.WriteLine(p.GetType());
* ICloneable c = p;
* Point p2 = (Point)p.Clone();

Сложность: [2]

1. **Отметьте правильные утверждения**

* Перечисления можно объявить в классе;
* Перечисления можно объявить в пространстве имен;
* Перечисления можно объявить в методе.

1. **Для какого типа используется Nullable <T>?**

* Ссылочные типы (reference types)
* Примитиные типы
* Типы значения (value types)
* Кассы и структуры
* Только int, bool, и DateTime

1. **Что из перечисленного верно**

* Переменные ссылочного типа (reference type) размещаются в стеке потока и хранят адрес объекта в управляемой куче.
* Переменные ссылочного типа (reference types) размещаются в управляемой куче.
* Переменные значимого типа (value type) размещаются в неуправляемой куче.
* Переменные значимого типа (value type) размещаются в стеке потока.
* Переменные ссылочного типа (reference types) размещаются в стеке потока и хранят адрес объекта в неуправляемой куче.

1. **Отметьте все верные утвержденеия о классе SomeClass**:

internal class SomeClass()

{

public void SomeClass

{

}

}

* SomeClass является значимым типом (value-type).
* SomeClass может использоваться внешними сборками.
* SomeClass наследуется от System.Object.
* экземпляр SomeClass нельзя создать вызовом new SomeClass(), поскольку в классе не определен открытый конструктор без параметров.
* SomeClass доступен для использования только внутри сборки, в которой он определён.