# 练习题-单项选择题参考答案

### 练习题1

（1）.NET Framework将 定义为一组规则，所有.NET语言都应该遵守这个规则才能创建可以与其他语言互操作的应用程序。A

A.CLR B.JIT C.MSIL D.ADO.NET

（2）在Visual Studio窗口中，在 窗口中可以察看当前项目的类和类型的层次信息。A

A.解决方案资源管理器 B.类视图 C.资源视图 D.属性

（3）在.NET Framework中，MSIL是指 。B

A.接口限制 B.中间语言 C.核心代码 D.类库

（4） 是独立于CPU的指令集，它可以被高效地转换为本机机器代码。C

A.CLR B.CLS C.MSIL D.Web服务

（5） 包含在.NET Framework的各语言之间兼容的数据类型。B

A.JIT B.CTS C.CLS D.MSIL

（6）C#源代码经过 次编译，才能在本机上执行。B

A.1 B.2 C.3 D.0

（7）所有.NET支持的编程语言编写的源代码经过一次编译后，被编译成 。D

A.机器代码 B.C#源代码 C.CLS代码 D.MSIL代码

（8）在.NET中，关于CLR和MSIL的、叙述中正确的是 。B

A.应用程序在CLR环境被编译成MSIL，MSIL能够被任何计算机执行

B..应用程序被编译两次，第一次生成MSIL，MSIL在本机运行时被CLR快速编译

C.应用程序被编译两次，但是第二次CLR编译比第一次慢

D.以上都不对

（9）CLR为.NET提供以下方面的功能或者服务，除了 。D

A.无用存储单元收集 B.代码验证和类型安全

C.代码访问安全 D.自动消除程序中的逻辑错误

（10）.NET Framework有两个主要组件，分别是 和.NET基础类库。A

A.公共语言运行环境 B.Web服务

C.命名空间 D. Main()函数

（11）控制台应用程序使用 命名空间中的类处理输入和输出。A

A.System.IO B.System.Web

C.System.Windows.Forms D.System.Data

（12）Windows应用程序使用 命名空间中的类处理输入和输出。C

A.System.IO B.System.Web

C.System.Windows.Forms D.System.Data

（13）以下 类型的应用程序适合于交互性操作较少的情况。B

A.Windows应用程序 B.控制台应用程序

C.Web应用程序 D.以上都不是

（14）以下对Read()和ReadLine()方法的叙述中正确的是 。A

A.Read()方法一次只能从输入流中读取一个字符

B.Read()方法可以从输入流中读取一个字符串

C.ReadLine()方法一次只能从输入流中读取一个字符

D.ReadLine()方法只有当用户按下回车键时返回，而Read()方法不是

（15）以下对Write()和WriteLine()方法的叙述中正确的是 。B

A.Write()方法在输出字符串的后面添加换行符

B.Write()方法输出字符串时，光标将会位于字符串的后面

C.Write()和WriteLine()方法输出数值变量时，必须要先把数值变量转换成字符串

D.使用不带参数的WriteLine()方法时，将不会产生任何输出

（16）以下的C#代码：

static void Main(string[] args)

{ Console.WriteLine("运行结果:{0}",Console.ReadLine());

Console.ReadLine();

}

代码运行结果为 。D

A.在控制台窗口显示“运行结果:”

B.在控制台窗口显示“运行结果:{0}”

C.在控制台窗口显示“运行结果:,Console.ReadLine”

D.如果用户在控制台输入“A”，那么程序将在控制台显示“运行结果:A”

（17）假设变量x的值为25，要输出*x*的值，以下正确的语句是 。C

A.System.Console.WriteLine(“x”) B.System.Console.WriteLine(“x”)

C.System.Console.WriteLine(“x={0}”,x) D.System.Console.WriteLine(“x={x}”)

（18）以下关于控制台应用程序和Windows应用程序的叙述中正确的是 。D

A.控制台应用程序中有一个Main静态方法，而Windows应用程序中没有

B.Windows应用程序中有一个Main静态方法，而控制台应用程序中没有

C.控制台应用程序和Windows应用程序中都没有Main静态方法

D.控制台应用程序和Windows应用程序中都有Main静态方法

### 练习题2

（1）在C#语言中，下列能够作为变量名的是 。C

A.if B.3ab C.a\_3b D.a-bc

（2）C#的数据类型分为 。B

A.值类型和调用类型 B.值类型和引用类型

C.引用类型和关系类型 D.关系类型和调用类型

（3）下列选项中， 是引用类型。C

A.enum类型 B.struct类型

C.string类型 D.int类型

（4）以下类型中，不属于值类型的是 。D

A.整数类型 B.布尔类型 C.字符类型 D.类类型

（5） 是将值类型转换成引用类型。A

A.装箱 B.拆箱 C.赋值 D.实例化

（6） 是将引用类型转换成值类型。B

A.装箱 B.拆箱 C.赋值 D.实例化

（7）C#中每个int类型的变量占用 个字节的内存。C

A.1 B.2 C.4 D.8

（8）在C#中，以下常量定义正确的是 。B

A.const double PI 3.1415926; B.const double e=2.7;

C.define double PI 3.1415926; D.define double e=2.7;

（9）在C#中，表示一个字符串的变量应使用以下 语句定义。B

A.CString str; B.string str;

C.Dim str as string; D.char \* str;

（10）在C#中，新建一个字符串变量str，并将字符串"Tom's Living Room"保存到串中，则应该使用 语句。B

A.string str = "Tom\\'s Living Room"; B.string str = "Tom's Living Room";

C.string str("Tom's Living Room"); D.string str("Tom"s Living Room");

（11）有以下的C#程序：

using System;

namespace aaa

{ public struct Person

{ string name;

int age;

}

class Program

{ static void Main(string[] args)

{ string a;

Person b;

//其他处理代码

}

}

}

以下说法正确的是 。A

A.a为引用类型的变量，b为值类型的变量

B.a为值类型的变量，b为引用类型的变量

C.a和b都是值类型的变量

D.a和b都是引用类型的变量

（12）在C#中可以通过装箱和拆箱实现值类型与引用类型之间相互转换，在下列代码中，有 处实现了拆箱。B

int age=5;

object o=age;

o=10;

age=(int)o;

object oAge=age;

A.0 B.1 C.2 D.3

（13）在C#中，下列代码运行后，变量Max的值是 。C

int a=5,b=10,c=15,Max=0;

Max = a>b?a:b;

Max = c<Max?c:Max;

A.0 B.5 C.10 D.15

（14）以下程序的输出结果是 。B

using System;

namespace aaa

{ class Example1

{ static void Main(string[] args)

{ int a=5,b=4,c=6,d;

Console.WriteLine("{0}",d=a>b?(a>c?a:c):b);

}

}

}

A.5 B.4 C.6 D.不确定

（15）以下对枚举类型的声明中正确的是 。B

A.enum a={one,two,three}; B.enum b {a1,a2,a3};

C.num c={'1','2','3'}; D.enum d {"one","two","three"};

（16）有以下C#程序：

using System;

namespace aaa

{ class Program

{ static void Main(string[] args)

{ byte a = 2, b = 5;

Console.WriteLine("{0}",a^b);

}

}

}

该程序的输出结果是 。C

A.2 B.5 C.7 D.9

（17）有以下Ｃ#程序：

using System;

namespace aaa

{ class Program

{ static void Main()

{ String str;

str = Console.ReadLine();

bool a = str.Equals("a");

Console.WriteLine(a.ToString());

int b = str.Length;

Console.WriteLine(b.ToString());

}

}

}

在程序运行时输入为： AAAAA（3个空格加5个A）则程序输出为 。B

A.0 8 B.False 8 C.-1 8 D.False 5

### 练习题3

（1）if语句后面的表达式应该是 。B

A.字符串表达式 B.条件表达式 C.算术表达式 D.任意表达式

（2）有以下C#程序：

using System;

namespace aaa

{ class Program

{ static void Main()

{ int x = 2, y = -1, z = 2;

if (x < y)

if (y < 0) z = 0;

else z += 1;

Console.WriteLine("{0}",z);

}

}

}

该程序的输出结果是 。B

A.3 B.2 C.1 D.0

（3）有以下C#程序，在执行时从键盘上输入9，则输出结果是 。B

using System;

namespace aaa

{ class Program

{ static void Main()

{ int n;

n = int.Parse(Console.ReadLine());

if (n++ < 10)

Console.WriteLine("{0}", n);

else

Console.WriteLine("{0}",n--);

}

}

}

A.11 B.10 C.9 D.8

（4）有以下C#程序：

using System;

namespace aaa

{ class Example1

{ static void Main(string[] args)

{ int x=1,a=0,b=0;

switch(x)

{

case 0:b++;break;

case 1:a++;break;

case 2:a++;b++;break;

}

Console.WriteLine("a={0},b={1}",a,b);

}

}

}

该程序的输出结果是 。C

A.a=2,b=1 B.a=1,b=1 C.a=1,b=0 D.a=2,b=2

（5）有以下C#程序：

using System;

namespace aaa

{ class Program

{ static void Main()

{ int a = 15, b = 21, m = 0;

switch (a % 3)

{

case 0: m++; break;

case 1: m++;

switch (b % 2)

{

case 0: m++; break;

default: m++; break;

}

break;

}

Console.WriteLine("{0}",m);

}

}

}

该程序的输出结果是 。A

A.1 B.2 C.3 D.4

（6）以下叙述正确的是 。D

A.do-while语句构成的循环不能用其他语句构成的循环来代替

B.do-while语句构成的循环只能用break语句退出

C.用do-while语句构成的循环，在while后的表达式为true时结束循环

D.用do-while语句构成的循环，在while后的表达式应为关系表达式或逻辑表达式

（7）以下关于for循环的说法不正确的是 。A

A.for循环只能用于循环次数已经确定的情况

B.for循环是先判定表达式，后执行循环体语句

C.for循环中，可以用break语句跳出循环体

D.for循环体语句中，可以包含多条语句，但要用花括号括起来

（8）有以下C#程序：

using System;

namespace aaa

{ class Program

{ static void Main()

{ int i,j,s=0;

for(i=2;i<6;i++,i++)

{ s=1;

for(j=i;j<6;j++)

s+=j;

}

Console.WriteLine("{0}",s);

}

}

}

该程序的输出结果是 。D

A.9 B.1 C.11 D.10

（9）有以下C#程序：

using System;

namespace aaa

{ class Program

{ static void Main()

{ int i = 0, s = 0;

do

{ if (i%2==1)

{ i++;

continue;

}

i++;

s += i;

} while (i < 7);

Console.WriteLine("{0}",s);

}

}

}

该程序的输出结果是 。A

A.16 B.12 C.28 D.21

（10）有以下C#程序：

using System;

namespace aaa

{ class Program

{ static void Main()

{ int i = 0, a = 0;

while (i < 20)

{ for (; ; )

{ if (i % 10 == 0) break;

else i--;

}

i += 11;

a += i;

}

Console.WriteLine("{0}", a);

}

}

}

该程序的输出结果是 。B

A.21 B.32 C.33 D.11

### 练习题4

（1）在C#中定义一个数组，以下正确的是 。B

A.int arraya = new int[5]; B.int[] arrayb = new int[5];

C.int arrayc = new int[]; D.int[5] arrayd = new int;

（2）以下数组定义语句中不正确的是 。D

A.int a[]=new int[5]{1,2,3,4,5}; B.int[,] a=new inta[3][4];

C.int[][] a=new int [3][0]; D.int [] a={1,2,3,4};

（3）以下定义并初始化一维数组的语句中正确的是 。C

A.int arr1 [ ]={6,5,1,2,3}; B.int [ ] arr2=new int[ ];

C.int[ ] arr3=new int[ ]{6,5,1,2,3}; D.int[ ] arr4;arr4={6,5,1,2,3};

（4）以下定义并动态初始化一维数组的语句中正确的是 。D

A.int [ ] arr1=new int[ ]; B.int arr2=new int[4];

C.int[ ] arr3=new int[i]{6,5,1,2,3}; D.int[ ] arr4=new int[]{6,5,1,2,3};

（5）以下定义并初始化数组的语句中正确的是 。D

A.int arr1[ ][ ]=new int[4,5]; B.int [ ][ ] arr2=new int[4,5];

C.int arr3[,]=new int[4,5] D.int[, ] arr4=new int[4,5];

（6）有定义语句：int [,]a=new int[5,6];则下列正确的数组元素的引用是 。D

A. a(3,4) B. a(3)(4) C. a[3][4] D. a[3,4]

（7）假定int类型变量占用4个字节，若有定义：int[] x = new int[] {1,2,3,4,5,6};

则数组x在内存中所占字节数是 。C

A.4 B.12 C.24 D.48

（8）在C#中，关于Array和ArrayList的维数，以下说法正确的是 。A

A.Array可以有多维，而ArrayList只能是一维

B.Array只能是一维，而ArrayList可以有多维

C.Array和ArrayList 都只能是一维

D.Array和ArrayList都可以是多维

（9）以下程序的输出结果是 。C

using System;

namespace aaa

{ class Example1

{ static void Main()

{ int i;

int [] a=new int[10];

for(i=9;i>=0;i--)

a[i]=10-i;

Console.WriteLine("{0},{1},{2}",a[2],a[5],a[8]);

}

}

}

A.2,5,8 B.7,4,1 C.8,5,2 D.3,6,9

（10）以下程序的输出结果是 。B

using System;

using System.Collections;

namespace aaa

{ class Example1

{ static void Main()

{ int[] num =new int[]{1,3,5};

ArrayList arr=new ArrayList();

for(int i=0;i<num.Length;i++)

arr.Add(num[i]);

arr.Insert(1,4);

Console.WriteLine(arr[2]);

}

}

}

A.1 B.3 C.4 D.5

（11）以下程序的输出结果是 。D

using System;

using System.Collections;

namespace aaa

{ class Example1

{ static void Main()

{ int [] num = new int[5]{1,3,2,0,0};

Array.Reverse(num);

foreach(int i in num)

Console.Write("{0} ",i);

Console.WriteLine();

}

}

}

A.0 0 1 2 3 B.1 2 3 0 0 C.0 0 1 3 2 D.0 0 2 3 1

（12）以下程序的输出结果是 。D

using System;

using System.Collections;

namespace aaa

{ class Example1

{ static void Main()

{ int s = 0;

int[][] a = new int[2][]; //交错数组

a[0] = new int[3] { 1, 2, 3 };

a[1] = new int[4] { 4, 5, 6, 7 };

for (int i = 0; i < a.Length; i++)

for (int j = 0; j < a[i].Length; j++)

s += a[i][j];

Console.WriteLine(s);

}

}

}

A.1 B.6 C.22 D.28

### 练习题5

（1）在C#中， 访问修饰符修饰的字段只能由当前程序集访问。C

A.public B.protected C.internal D.private

（2）类ClassA有一个名称为M1的方法，在程序中有如下一段代码，假设该段代码是可以执行的，则修饰M1方法时一定使用了 修饰符。C

ClassA obj=new ClassA();

ClassA.M1();

A.public B.static C.public static D.virtual

（3）在C#中，从属性的读写特性上来进行分类，可以划分为以下三种，除了 。D

A.只读属性 B.只写属性 C.读写属性 D.不可读不可写的属性

（4）在类的定义中，类的 描述了该类的对象的行为特征。B

A.类名 B.方法 C.所属的命名空间 D.私有域

（5）在C#中，以下关于属性的描述正确的是 。B

A.属性是以public关键字修饰的字段，以public关键字修饰的字段也可称为属性

B.属性是访问字段值的一种灵活机制，属性更好地实现了数据的封装和隐藏

C.要定义只读属性只需在属性名前加上readonly关键字

D.在C#的类中不能自定义属性

（6）以下类MyClass的属性count属于 属性。A

class MyClass

{ int i;

int count

{ get{ return i; } }

}

A.只读 B.只写 C.可读写 D.不可读不可写

（7）以下关于C#中方法重载的说法正确的是 。C

A.如果两个方法名称不同，而参数的个数不同，那么它们可以构成方法重载

B.如果两个方法名称相同，而返回值的数据类型不同，那么它们可以构成方法重载

C.如果两个方法名称相同，而参数的数据类型不同，那么它们可以构成方法重载

D.如果两个方法名称相同，而参数的个数相同，那么它们一定不能构成方法重载

（8）以下 不是构造函数的特征。D

A.构造函数的函数名和类名相同 B.构造函数可以重载

C.构造函数可以带有参数 D.可以指定构造函数的返回值

D

（9）在C#中，以下有关索引器的参数个数的说法中正确的是 。B

A.索引器只能有1个参数 B.索引器可以有多个参数

C.索引器可以没有参数 D.索引器至少要有2个参数

（10）在类MyClass中有下列方法定义：

public void testParams(params int[] arr)

{ Console.Write ("使用Params参数!"); }

public void testParams(int x,int y)

{ Console.Write ("使用两个整型参数!"); }

请问上述方法重载有无二义性？若没有，则下列语句的输出为 。B

MyClass x = new MyClass();

x.testParams(0);

x.testParams(0,1);

x.testParams(0,1,2);

Console.WriteLine();

A.有语义二义性

B.使用Params参数!使用两个整型参数!使用Params参数!

C.使用Params参数!使用Params参数!使用Params参数!

D.使用Params参数!使用两个整型参数!使用两个整型参数!

（11）以列程序的输出结果是 。C

using System;

namespace aaa

{ class Example1

{ static long sub(int x,int y)

{ return x\*x+y\*y; }

public static void Main()

{ int a=30;

sub(5,2);

Console.WriteLine("{0}",a);

}

}

}

A.0 B.29 C.30 D.无定值

（12）C#中MyClass为一个自定义类，其中有以下方法定义public void Hello(){…}。使用以下语句创建了该类的对象，并使变量obj引用该对象：MyClass obj=new MyClass();那么，可访问类MyClass的Hello方法的语句是 。A

A.obj.Hello(); B.obj::Hello();

C.MyClass.Hello(); D.MyClass::Hello();

（13）分析以下C#语句，注意类MyClass没有显式指定访问修饰符：

namespace aaa

{ class MyClass

{ public class subclass

{ int i; }

}

}

该类MyClass的默认访问修饰符是 。C

A.private B.protected C.internal D.public

（14）分析下列程序：

public class MyClass

{ private string \_sData = "";

public string sData

{ set { \_sData = value; } }

}

在Main函数中，在成功创建该类的对象obj后，以下 哪些语句是合法的。A

A.obj.sData = "It is funny!"; B.Console.WriteLine(obj.sData);

C.obj.\_sData = 100; D.obj.set(obj.sData);

（15）以下 关键字用于委托类型。A

A.delegate B.event C.this D.value

（16）以下关于委托和委托类型的叙述中正确的是 。D

A.委托不是一种类的成员 B.委托必须定义在类中

C.定义委托需要使用delegate关键字 D.委托类型是一种数据类型

（17）分析下列语句：

namespace NS

{ public delegate void Hello(string target); }

该语句的作用是 。D

A.在NS命名空间中定义了一个名称为Hello的全局方法

B.在NS命名空间中声明了函数Hello的原型

C.在NS命名空间中声明了一个名称为Hello的函数指针

D.在NS命名空间中声明了一个名称为Hello的委托类型

（18）有以下部分C#代码：

class App

{ public static void Main()

{ mydelegate p= new mydelegate(CheckStatus);

p("string...");

…

}

static void CheckSatus(string state)

{ Console.WriteLine(state); }

}

其中mydelegate是一个 。A

A.委托类型 B.结构类型 C.函数 D.类名

（19）以下C#程序的执行情况是 。B

using System;

namespace aaa

{ delegate void delep(int i);

class Program

{ public static void Main()

{ funb(new delep(funa)); }

public static void funa(int t)

{ funb(21); }

public static void funb(int i)

{ Console.WriteLine(i.ToString()); }

}

}

A.代码中存在错误，delegate void delep(int i);不能定义在名称空间或者类之外

B.代码中存在错误，代码行funb(new delep(funa))使用委托错误

C.程序正常运行，输出为0

D.程序正常运行，输出为21

（20）已知委托类型DoSomething定义如下：

public delegate void DoSomething();

a、b、c和d都是DoSomething的变量，分别有以下调用列表：

a：objA.Func1、objA.Func2

b：objA.Func1、Class1.StaticFunc

c：objA.Func1、Class2.StaticFunc

d：objB.Fun1

其中objA为类Class1的对象，objB为类Class2的对象。则执行b=b+c后，变量b关联的方法个数为 。C

A.2 B.3 C.4 D.5

（21）以下 关键字用于定义事件。B

A.delegate B.event C.this D.value

B

（22）将发生的事件通知其他对象（订阅者）的对象称为事件的 。C

A.广播者 B.通知者 C.发行者 D.订阅者

（23）已知类MyClass中事件MouseClicked定义如下：

public delegate void mydelegate();

public event mydelegate MouseClicked;

执行下列语句：

Method obj = new Method();

MyClass e = new MyClass();

e.MouseClicked += obj.fun;

e.MouseClicked += obj.fun;

其中Method类中包含事件处理方法fun。然后引发该MouseClicked事件，其结果为 。B

A.obj.fun方法被调用4次 B.obj.fun方法被调用2次

C.obj.fun方法被调用1次 D.obj.fun方法不会被调用

（24）在C#中，以下关于命名空间的叙述中正确的是 。D

A.命名空间不可以进行嵌套

B.任一个.cs文件中，只能存在一个命名空间

C.便用private修饰的命名空间，其内部的类也不允许访问

D.命名空间使得代码更加有条理、结构更清晰

### 练习题6

（1）在C#中，一个类 。B

A.可以继承多个类 B.可以实现多个接口

C.在一个程序中只能有一个子类 D.只能实现一个接口

（2）以下关于继承机制的叙述中正确的是 。D

A.在C#中任何类都可以被继承

B.一个子类可以继承多个父类

C.object类是所有类的基类

D.继承有传递性，如A类继承B类，B类又继承C类，那么A类也继承C类的成员

（3） 关键字用于在C#中从派生类中访问基类的成员。D

A.new B.super C.this D.base

（4）在定义类时，如果希望类的某个方法能够在派生类中进一步进行改进，以处理不同的派生类的需要，则应将该方法声明成 。C

A.sealed方法 B.public方法 C.virtual方法 D.override方法

（5）在C#语法中，在派生类中对基类的虚函数进行重写，要求在派生类的声明中使用 。A

A.override B.new C.static D.virtual

（6）有以下程序：

using System;

namespace aaa

{ class A

{ public A()

{ Console.Write("A "); }

}

class B : A

{ public B()

{ Console.WriteLine("B"); }

}

class Program

{ public static void Main()

{ B b = new B(); }

}

}

上述代码运行后，将在控制台窗口输出 。C

A.A B.B C.A B D.B A

（7）以下关于抽象类的叙述中错误的是 。D

A.抽象类可以包含非抽象方法

B.含有抽象类方法的类一定是抽象类

C.抽象类不能被实例化

D.抽象类可以是密封类

（8）在C#中接口与抽象基类的区别在于 。A

A.抽象类可以包含非抽象方法，而接口不包含任何方法的实现

B.抽象类可以被实例化，而接口不能被实例化

C.抽象类不能被实例化，而接口可以被实例化

D.抽象类中能够被继承，而接口不能被继承

（9）以下关于抽象类和接口的叙述中正确的是 。B

A.在抽象类中，所有的方法都是抽象方法

B.继承自抽象类的子类必须实现其父类（抽象类）中的所有抽象方法

C.在接口中，可以有方法实现，在抽象类中不能有方法实现

D.一个类可以从多个接口继承，也可以从多个抽象类继承

（10）以下类MyClass的定义中 是合法的抽象类。A

A.abstract class MyClass { public abstract int getCount(); }

B.abstract class MyClass { abstract int getCount(); }

C.private abstract class MyClass { abstract int getCount(); }

D.sealed abstract class MyClass { abstract int getCount(); }

（11）多态是指两个或多个属于不同对象，对于同一个消息作出不同响应的方式。C#中的多态不能通过 实现。D

A.接口 B.抽象类 C.虚方法 D.密封类

（12）分析下列程序中类MyClass的定义

class BaseClass

{

public int i;

}

class MyClass:BaseClass

{

public new int i;

}

则下列语句在Console 上的输出为 。D

MyClass y = new MyClass();

BaseClass x = y;

x.i = 100;

Console.WriteLine("{0}, {1}",x.i,y.i);

A.0, 0 B.100, 100 C.0, 100 D.100, 0

（13）接口和类的区别中正确的是 。D

A.类可以继承而接口不可以

B.类不可以继承而接口可以

C.类可以多继承而接口不可以

D.类不可以多继承而接口可以

（14）接口Animal声明如下：A

public interface Animal

{

void Move();

}

则下列抽象类的定义中， 是合法的。

A.abstract class Cat: Animal

{ abstract public void Move(); }

B. abstract class Cat: Animal

{ virtual public void Move(){Console.Write(Console.Write("Move!");)} }

C. abstract class Cat: Animal

{ public void Move(){Console.Write(Console.Write("Move!");)}; }

D. abstract class Cat: Animal

{ public void Eat(){Console.Write(Console.Write("Eat!");)}; }

（15）已知接口IHello和类Base、Derived的声明如下：A

interface IHello

{

void Hello();

}

class Base : IHello

{ public void Hello()

{ System.Console.WriteLine("Hello in Base!"); }

}

class Derived : Base

{ public void Hello()

{ System.Console.WriteLine("Hello in Derived!"); }

}

则下列语句在控制台中的输出结果为 。

IHello x = new Derived();

x.Hello();

A.Hello in Base! B.Hello in Derived!

C.Hello in Base!Hello in Derived! D.Hello in Derived!Hello in Base!

（16）在C#程序中定义如下IPlay接口，实现此接口的代码正确的是 。C

interface IPlay

{ void Play();

void Show();

}

A.class Teacher : IPlay

{ void Play() { //省略部分代码 }

void Show() { //省略部分代码 }

}

B.class Teacher : IPlay

{ public string Play() { //省略部分代码 }

public void Show() { //省略部分代码 }

}

C.class Teacher : IPlay

{ public void Play() { //省略部分代码 }

public void Show() { //省略部分代码 }

}

D.class Teacher : IPlay

{ public void Play();

public void Show() { //省略部分代码 }

}

（17）以下程序的输出结果是 。B

using System;

namespace aaa

{ public class Vehicle

{ private int speed = 10;

public int Speed

{ get { return speed; }

set

{ speed = value;

Console.WriteLine("禁止驶入");

}

}

}

public class NewVehicle : Vehicle

{ public NewVehicle()

{ if (this.Speed >= 20)

Console.Write("机动车!");

else

Console.Write("非机动车!");

}

}

public class Program

{ public static void Main()

{ NewVehicle tong = new NewVehicle();

tong.Speed = 30;

}

}

}

A.禁止驶入非机动车! B.非机动车!禁止驶入

C.禁止驶入机动车! D.机动车禁止驶入!

（18）以下程序的输出结果是 。D

using System;

namespace aaa

{ public class Person

{ private int age = 0;

public int Age

{ get { return age; }

set

{ if (value>=18)

Console.WriteLine("成年人");

else

Console.WriteLine("非成年人");

age = value;

}

}

}

public class People:Person

{ public People()

{ Console.Write("不得入内"); }

}

public class Program

{ public static void Main()

{ People Shang = new People();

Shang.Age=25;

}

}

}

A.非成年人不得入内 B.成年人不得入内

C.不得入内非成年人 D.不得入内成年人

### 练习题7

（1）以下泛型集合声明中正确的是 。A

A.List <int> f= new List<int>(); B.List<int> f=new List();

C.List f=new List(); D.List<int> f=new List<int>;

（2）以下关于泛型的叙中错误的是 。D

A.泛型是通过参数化类型来实现在同一份代码上操作多种数据类型

B.泛型编程是一种编程范式，其特点是参数化类型

C.泛型类型和普通类型的区别在于泛型类型与一组类型参数或类型变量关联

D.以上都不对

（3）在C#程序中，关于反射的说法错误的是 。B

A.使用反射机制可以在程序运行时通过编程方式获得类型信息

B.使用反射机制，需要在程序中引入的命名空间是System.Runtime

C.通过反射可以查找程序集的信息

D.反射是一个运行库类型发现的过程

（4）在.NET Framework中，所有与反射应用相关的类都放在 命名空间中。D

A.System.Data B.System.IO

C.System.Threading D.System.Reflection

（5）在反射机制中，通过 类可以找到事件的信息。A

A.EventInfo B.FieldInfo

C.ParameterInfo D.PropertyInfo

（6）在反射机制中，通过 类可以找到方法的信息。B

A.FieldInfo B.MethodInfo

C.ParameterInfo D.PropertyInfo类

（7）在反射机制中，通过 类可以找到属性的信息。D

A.EventInfo B.FieldInfo

C.MethodInfo D.PropertyInfo

### 练习题8

（1）有如下代码：

int[] a = { 1, 2, 3, 4, 5 };

foreach (int i in a)

{

…

}

在foreach循环中，如果包含 语句，则会出现错误。A

A.i+=2; B. Console.WriteLine(i+2);

C.int j=i; Console.WriteLine(++j); D. Console.WriteLine(i>>2);

（2）执行foreach语句的前提是操作对象是 类型，而且存在相应的枚举器。D

A.整数 B.实数 C.object D.可枚举

（3）IEnumerator接口的命名空间是 。A

A. System.Collections B. System.Reflection

C. System.Threading D.以上都不对

（4）以下不属于IEnumerator 接口的public成员的是 。C

A.Current B.MoveNext C.公共方法由 XNA Framework 提供支持GoNext D.Reset

（5）以下属于yield 语句的形式是 。C

A.yield return B.yield break C.yield foreach D.以上都不对

（6）可以将迭代器作为属性，需要在属性的get访问器中使用 语句。A

A.yield return B.return C.foreach D.以上都不对

### 练习题9

（1）C#程序中为使变量myForm引用的窗体对象显示为对话框，必须 。A

A.使用myForm.ShowDailog()方法显示对话框

B.将myForm对象的isDialog属性设为true

C.将myForm对象的FormBorderStyle枚举属性设置为FixedDialog

D.将变量myForm改为引用System.Windows.Dialog类的对象

（2）在C#程序中，文本框控件的 属性用来设置其是否是只读的。A

A.ReadOnly B.Locked C.Lock D.Style

（3）设置文本框的 属性可以使其显示多行。C

A.PasswordChar B.ReadOnly C.Multiline D.MaxLength

（4）当用户单击窗体上的命令按钮时，会引发命令按钮控件的 事件。A

A.Click B.Leave C.Move D.Enter

（5）在Windows应用程序中，如果复选框控件的Checked属性值设置为True，表示 。A

A.该复选框被选中 B.该复选框不被选中

C.不显示该复选框的文本信息 D.显示该复选框的文本信息

（6）要获取ListBox控件当前选中项的文本，通过 属性得到。D

A.SelectedIndex B.SelectedItem C.Items D.Text

（7）要获取ComboBox控件所包含项的集合，通过 属性得到。C

A.SelectedItem B.SelectedText C.Items D.Sorted

（8）启动一个定时器控件的方法是 。C

A.Enabled B.Interval C.Start D.Stop

（9）已知在某Windows应用程序中，主窗口类为Form1，程序入口为静态方法From1.Main。如下所示：

public class Form1 : System.Windows.Forms.Form

{ //其他代码

static void Main()

{

//在此添加合适代码

}

}

则在Main方法中打开主窗口的正确代码是 。B

A.Application.Open(new Form1());

B.Application.Run(new Form1());

C.(new Form1()).Open();

D.(new Form1()).Run();

（10）在Windows应用程序中，可以通过以下 方法使一个窗体成为MDI窗体。C

A.改变窗体的标题信息

B.在工程的选项中设置启动窗体

C.设置窗体的IsMdiContainer属性为True

D.设置窗体的ImeMode属性

### 练习题10

（1）用鼠标右击一个控件时出现的菜单一般称为 。C

A.主菜单 B.菜单项 C.快捷菜单 D.子菜单

（2）为菜单添加快捷键的属性是 。A

A.ShortcutKeys B.keys C.MenuKeys D.MenuShortcutKeys

（3）变量openFileDialog1引用一个OpenFileDialog对象。为检查用户在退出对话框时是否单击了“确定”按钮，应检查openFileDialog1.ShowDialog()的返回值是否等于 。A

A.DialogResult.OK B.DialogResult.Yes

C.DialogResult.No D.DialogResult.Cancel

（4）展开一个TreeView控件中所有节点，使用 方法。B

A.CollapseAll B.ExpandAll

C.FindNode D.Indent

（5）列表视图ListView的View属性的默认值为 。A

A.大图标LargeIcon B.小图标SmallIcon

C.详细资料Details D.列表List

### 练习题11

（1）在GDI+的所有类中， 类是核心，在绘制任何图形之前，一定要先用它创建一个对象。A

A.Graphics B.Pen C.Brush D.Font

（2）在Windows应用程序中，在界面上绘制矩形、弧、椭圆等图像对象，可以使用System.Drawing命名空间的 类来实现。B

A.Brush B.Pen C.Color D.Image

（3）要设置Pen对象绘制线条的宽度，应使用它的 属性。B

A.Color B.Width C.DashStyle D.PenType

（4）通过HatchBush对象的 属性可设置HatchBush对象的阴影样式。C

A.BackgroundColor B.ForegroundColor

C.HatchStyle D.ColorStyle

（5）在界面上创建字体的类是 。D

A.Graphics B.Pen C.Brush D.Font

（6）在界面上绘制文本使用Graphics对象的 方法。C

A.DrawPie B.FillPie C.DrawString D.FillEllipse

### 练习题12

（1）以下不属于文件访问方式的是 。D

A.只读 B.只写 C.读写 D.不读不写

（2）以下 类提供了文件夹的操作功能。B

A.File B.Directory C.FileStream D.BinaryWriter

（3）用FileStream对象打开一个文件时，可用FileMode参数控制 。A

A.对文件覆盖、创建、打开等选项中的哪些操作

B.对文件进行只读、只写还是读写

C.其他FileStream对象对同一个文件所具有的访问类型

D.对文件进行随机访问时的定位点

（4）在使用FileStream对象打开一个文件时，通过使用FileMode枚举类型的 成员，来指定操作系统打开一个现有文件并把文件读写指针定位在文件尾部。A

A.Append B.Create C.CreateNew D.Truncate

（5）假设要使用C#设计一个日志系统，要求程序运行时，检查system.log文件是否存在，如果已经存在则直接打开，如果不存在则创建一个，为了实现这个目的，我们应该以FileMode的 方式创建文件流。C

A.CreateNew B.Open C.OpenOrCreate D. Create

（6）将文件从当前位置一直到结尾的内容都读取出来，应该使用 方法。A

A.StreamReader.ReadToEnd() B.StreamReader.ReadLine()

C.StreamReader.ReadBlock() D.StreamReader.WriteLine()

（7）FileStream类的 方法用于定位文件位置指针。B

A.Close B.Seek C.Lock D.Flush

（8）关于以下C#代码的说法正确的是 。D

FileStream fs = new FileStream("D:\\music.txt",FileMode.OpenOrCreate);

A.如果D盘根目录中没有文件music.txt，则代码运行时出现异常

B.如果D盘根目录中存在文件music.txt，则代码运行时出现异常

C.该语句存在语法错误

D.代码执行后，D盘根目录一定存在文件music.txt

（9）以下语句定义和初始化一个整型数组*a*：A

int[] a = new int[400];

for(int i = 0;i<400;i++) a[i]=i;

为了将数组*a*的所有元素值写入FileStream流中，可创建 类的实例对该流进行写入。

A.BinaryWriter B.StreamWriter C.TextWriter D.StringWriter

（10）在C#中类上加 关键字来标记该类支持序列化。A

A.[Serializable] B.[Formatable] C.[Stream] D.[STAThread]

### 练习题13

（1）程序执行过程中发生的错误称为 。C

A.版本 B.断点 C.异常 D.属性

（2）以下关于程序的各种错误中说法错误的是 。D

A.只通过测试无法确保程序运行完全正常

B.通过异常处理可以捕获运行错误

C.逻辑错误编译时不能被发现，但是可以通过测试发现

D.语法错误容易在运行时发现

（3）程序调试的基础是设置断点，以下叙述中正确的是 。B

A.在一个程序中只能设置一个断点

B.在一个程序中可以设置一个或多个断点

C.在一个程序中必须设置两个或两个以上的断点

D.以上都不对

（4）以下关于C#的异常处理的叙述中正确的是 。A

A.try块后面必须跟catch块或finally块组合使用，不能单独使用

B.一个try块后面只能跟随一个catch块

C.throw语句中必须指出抛出的异常

D.在try…catch…finally块中，当发生异常时不会执行finally块

（5）在C#程序中可以使用try..catch机制来处理程序出现的 错误。B

A.语法 B.运行 C.逻辑 D.拼写

（6）在C#中，在方法MyFunc内部的try…catch语句中，如果在try代码块中发生异常，并且在当前的所有catch块中都没有找到合适的catch块，则 。B

A.系统运行时忽略该异常

B.系统运行时马上强制退出该程序，指出未处理的异常。

C.系统运行时继续在MyFunc的调用堆栈中查找提供该异常处理的过程

D.系统抛出一个新的“异常处理未找到”的异常

（7）以下关于try…catch…finaly语句的叙述中不正确的是 。D

A.catch块可以有多个

B.finaly块最多只有一个

C.catch块也是可选的

D.可以只有try块，没有catch块和finaly块

（8）为了能够在程序中捕获所有的异常，在catch语句的括号中使用的类名为 。A

A.Exception B.DivideByZeroException

C.FormatException D.以上三个均可

（9）假设给出下面代码：

try

{

throw new OverflowException();

}

catch(FileNotFoundException e) { Console.Write("1 "); }

catch(OverflowException e){Console.Write("2 ");}

catch(SystemException e){Console.Write("3 ");}

catch {Console.Write("4 ");}

finally {Console.Write("5 ");}

执行时的输出结果是 。B

A.1 5 B.2 5 C.3 5 D.2 3

（10）有以下的C#程序：

using System;

namespace aaa

{ public class Program

{ public static void ThrowException()

{ throw new Exception(); }

public static void Main()

{ try

{ Console.Write("try ");

ThrowException();

}

catch (Exception e)

{ Console.Write("catch "); }

finally

{ Console.Write("finally "); }

}

}

}

该程序的运行结果是 。A

A.try catch finally B.try

C.try catch D.try finally

### 练习题14

（1）在.NET Framework中，所有与多线程机制应用相关的类都放在 命名空间中。C

A.System.Data B.System.IO

C.System.Threading D.System.Reflection

（2）在C#中，通过调用Thread类的Sleep(int *x*)方法来设置禁止线程运行的时间，其中*x*代表 。B

A.禁止线程运行的微秒数

B.禁止线程运行的毫秒数

C.禁止线程运行的秒数

D.禁止线程运行的CPU时间数

（3）以下实现线程互斥的叙述中正确的是 。D

A.可以用lock语句实现线程互斥

B.可以用Mutex类实现线程互斥

C.可以用Monitor类实现线程互斥

D.以上都正确

（4）异步方法在被调用之后 以便程序在被调用方法完成其任务的同时执行其他操作。A

A.立即返回 B.等待而不返回

C.不等待也不返回 D.以上都不对

（5）一个异步方法内部通常有 await 表达式。B

A. 一个 B.一个或多个  C. 多个 D.以上都不对

（6）若要获得异步任务的返回值，应该使用 类型。C

A.Task B.void C.Task<TResult> D.以上都不对

（7）异步方法头部中应该包含 修饰符。D

A.public B.private C.internal D.async

（8）以下不属于异步方法的特征的是 。D

A.方法头部中包含async修饰符。

B.包含一个或多个await表达式，表示可以异步完成的任务。

C.异步方法的参数可以任意个数任意类型，但不能为out或ref参数。

D. 异步方法的返回类型必须是Task<T>

### 练习题15

（1）在.NET Framework中可以使用 对象来连接和访问数据库。C

A.MDI B.JIT C.ADO.NET D.System.ADO

（2）以下 是ADO.NET的两个主要组件。C

A.Command和DataAdapter B.DataSet和DataTable

C..NET数据提供程序和DataSet D..NET数据提供程序和DataAdapter

（3）在ADO.NET中，OleDbConnection类所在的命名空间是 。C

A.System B.System.Data

C.System.Data.OleDb D.System.Data.SqlClient

（4）Connection对象的 方法用于打开与数据库的连接。C

A.Close B.ConnectionString C.Open D.Database

（5）以下 类的对象是ADO.NET在非连接模式下处理数据内容的主要对象。D

A.Command B.Connection C.DataAdapter D.DataSet

（6）在ADO.NET中，以下关于DataSet类的叙述中错误的是 。D

A.可以向DataSet的表集合中添加新表

B.DataSet支持ADO.NET的不连贯连接及数据分布

C.DataSet就好象是内存中的一个临时数据库

D.DataSet中的数据是只读的并且是只进的

（7）在ADO.NET中，下列代码运行后的输出结果是 。B

DataTable dt=new DataTable();

dt.Columns.Add("编号",typeof(System.Int16));

dt.Columns.Add("成绩",typeof(System.Single));

Console.WriteLine(dt.Columns[1].DataType);

A.System.Int16 B.System.Single

C.编号 D.成绩

（8） 方法执行指定为Command对象的命令文本的SQL语句，并返回受SQL语句影响或检索的行数。A

A.ExecuteNonQuery B.ExecuteReader

C.ExecuteQuery D.ExecuteScalar

（9）看代码回答问题

建表的SQL语句如下：

CREATE TABLE stuInfo

( stuId int not null,

stuName varchar(30) not null,

stuAddress varchar(30),

)

有以下C#代码，其功能是要读取表中的第一列数据（已知myreader为OleDbDataReader对象），请选择填空为 。D

while (myreader. ① )

{

Console.WriteLine(myreader. ② );

}

A.①Read() ②GetString[0] B.①Next() ②GetString[0]

C.①Read() ②GetInt32[1] D.①Read() ②GetInt32[0]

（10）在窗体上拖放一个DataAdapter 对象后，可使用 来配置其属性。A

A.数据适配器配置向导 B.数据窗体向导

C.服务器资源管理器 D.对象浏览器

（11）在ADO.NET中，执行数据库的某个存储过程，则至少需要创建 并设置它们的属性，并调用合适的方法。A

A.一个Connection对象和一个Command对象

B.一个Connection对象和DataSet对象

C.一个Command对象和一个DataSet对象

D.一个Command对象和一个DataAdapter对象

（12）在使用ADO.NET编写连接到SQL Server数据库的应用程序时，从提高性能角度考虑，应创建 类的对象，并调用其Open方法连接到数据库。B

A.Connection B.SqlConnection

C.OleDbConnection D.OdbcConnection

（13）在ADO.NET中，DataAdapter对象的 属性用于将DataSet中的新增记录保存到数据源。B

A.SelectCommand B.InsertCommand

C.UpdateCommand D.DeleteCommand

（14）ADO.NET中的DataView控件可以用来筛选数据集中的数据项，以下的代码用来选择数据集中年龄小于24的员工：

DataView mydv = new DataView(mydataSet.Tables[0]);

①

则①处应该填写的正确代码为 。A

A.mydv.RowFilter = "age <24";

B.mydv.RowFilter = "SELECT age FROM mydv WHERE age < 24";

C.mydv.Excute("SELECT age FROM mydv WHERE age < 24");

D.mydv.Excute("age < 24");

### 练习题16

（1）XML是 。B

A.一种标准标记语言 B.一种扩展标记语言

C.一种超文本标记语言 D.都不是

（2）以下关于XML的叙述中错误的是 。A

A.XML是HTML的升级产品 B.XML文件中的节点区分大小写

C.XML节点是成对出现的 D.XML节点是可扩展的

（3）XML文档操作类的命名空间是 。B

A.System.Data B.System.Xml

C.System.Data.Xml D.System.Xml.Data

（4）以下关于XPath的叙述中错误的是 。D

A.XPath表达式是指符合W3C XPath 2.0建议的字符串表达式

B.XPath表达式用于在匹配XML文档结构树时能够准确地找到节点元素

C.可以将XPath比作文件管理路径

D.以上都不对

（5）将XML文档转换成DataSet数据时需使用DataSet类的 方法。C

A.Open B.Read C.ReadXml D.WriteXml

### 练习题17

（1）LINQ to Objects提供程序的命名空间是 。C

A.System B.System.IO C. System.Linq D.Linq

（2）使用LINQ可以查询许多不同的数据源，但这些数据源不包括 。A

A.Access数据库 B.SQL Server数据库 C.数组 D.List<T>集合

（3）LINQ查询由多个子句构成，其中 用于指定数据源。A

A.from子句 B.where子句 C.select子句 D.orderby子句

（4）LINQ查询由多个子句构成，其中 用于指定条件。B

A.from子句 B.where子句 C.select子句 D.orderby子句

（5）LINQ查询由多个子句构成，其中 用于选择字段。C

A.from子句 B.where子句 C.select子句 D.orderby子句

（6）LINQ查询由多个子句构成，其中 用于排序查询结果。D

A.from子句 B.where子句 C.select子句 D.orderby子句

（7）在LINQ查询表达式中group子句的作用是 。D

A.指定数据源 B.对查询结果排序 C.数据连接 D.数据分组

（8）在LINQ查询表达式中join子句的作用是 。C

A.指定数据源 B.对查询结果排序 C.数据连接 D.数据分组

### 练习题18

（1）Web窗体文件的扩展名为 。B

A.ASP B.ASPX C.ASCX D.HTML

（2）以下关于ASP.NET网页结构的叙述中正确的是 。B

A.单文件页模型是将标记保存在.aspx文件中，编程代码保存在.cs文件中

B.代码隐藏页模型是将标记保存在.aspx文件中，编程代码保存在.cs文件中

C.单文件页模型的所有代码都保存在.cs文件中

D.代码隐藏页模型的所有代码都保存在.aspx文件中

（3）关于ASP.NET窗体应用程序，下列说法正确的是 。A

A.Web窗体是在浏览器中进行解释执行的

B.Web窗体程序中脚本和代码必须严格分开

C.Web窗体程序产生的HTML网页只能运行于IE浏览器

D.Web窗体程序的代码只能用C#语言编写

（4）以下关于服务器控件的叙述中正确的是 。C

A.服务器控件就是HTML标记

B.服务器控件只能在客户机上运行

C.服务器控件只能在服务器上运行

D.一个网页上的控件只能是服务器控件

（5）Web标准服务器控件的命名空间是 。C

A.System.Web B.System.Web.UI

C.System.Web.UI.WebControls D.System.Web.UI.Control

（6）DataGridView和GridView控件的区别是 。A

A.DataGridView控件用于设计Windows窗体，而GridView控件用于设计Web网页

B.DataGridView和GridView控件都可以用设计Windows窗体

C.DataGridView和GridView控件都可以用设计Web网页

D.DataGridView和GridView控件的功能完全相同

（7）所有的数据验证控件都要设置 属性。D

A.Data B.DataSource

C.ControlToCompare D.ControlToValidate