№ 5 Наследование, полиморфизм, абстрактные классы и интерфейсы

Задание

1) Определить иерархию и композицию классов (в соответствии с вариантом), реализовать классы. Если необходимо расширьте по своему усмотрению иерархию для выполнения всех пунктов л.р.

   Каждый класс должен иметь отражающее смысл название и информативный состав. При кодировании должны быть использованы соглашения об оформлении кода code convention.

   В одном из классов переопределите все методы, унаследованные от Object.
2) В проекте должен быть минимум один **интерфейс и абстрактный класс**. Использовать виртуальные методы и переопределение.
3) Сделайте один из классов **sealed**;
4) Добавьте в интерфейсы (интерфейс) и абстрактный класс **одноименные методы**.

   Например,

```
interface ICloneable
 {
     bool DoClone();
 }

 abstract class BaseClone
 {
     public abstract bool DoClone();
 }
  class UserClass:BaseClone, ICloneable
 {
     ....
  }
```

   Дайте в наследуемом классе им разную реализацию и вызовите эти методы.

5) Написать демонстрационную программу, в которой создаются объекты различных классов. Поработать с объектами **через ссылки на абстрактные классы и интерфейсы**. В этом случае для идентификации типов объектов использовать **операторы is или as.**
6) Во всех классах (иерархии) переопределить метод ToString(), который выводит информацию о типе объекта и его текущих значениях.
7) Создайте дополнительный класс Printer с полиморфным методом IAmPrinting(SomeAbstractClassorInterface someobj). Формальным параметром метода должна быть ссылка на абстрактный класс или наиболее общий интерфейс в вашей иерархии классов. В методе iIAmPrinting определите тип объекта и вызовите ToString(). В демонстрационной программе создайте массив, содержащий ссылки на разнотипные объекты ваших классов по иерархии, а также объект класса Printer и последовательно вызовите его метод IAmPrinting со всеми ссылками в качестве аргументов.

Далее приведен перечень классов:

Вариант 1	ПО, Набор операций, Текстовый процессор, Word, Вирус, CConficker Игрушка, Сапер, Разработчик.
Вариант *2*	Геометрическая фигура, Круг, Прямоугольник, Управление (интерфейс с методами show, input, resize и т.д.), Элемент управления, Checktbox, Radiobutton, Button.
Вариант 3	Транспортное средство, Управление авто, Машина, Двигатель, Разумное существо, Человек, Трансформер;
Вариант 4	Растение, Куст, Цветок, Роза, Гладиолус, Кактус, Бумага, Букет.
Вариант 5	Товар, Техника, Печатающее устройство, Сканер, Компьютер, Планшет.
Вариант 6	Журнал, Книга, Печатное издание, Учебник, Журнал, Персона, Автор, Издательство.
Вариант 7	Испытание, Тест, Экзамен, Выпускной экзамен, Вопрос;
Вариант 8	Телевизионная программа, Фильм, Новости, Худ. фильм, Мультфильм, Реклама, Режиссер.
Вариант 9	Продукт, Кондитерское изделие, Товар, Цветы, Торт, Часы, Конфеты;
Вариант 10	Квитанция, Накладная, Документ, Чек, Дата, Организация.
Вариант 11	Автомобиль, Поезд, Транспортное средство, Экспресс, Двигатель, Вагон.
Вариант 12	Инвентарь, Скамейка, Брусья, Мяч, Маты, Баскетбольный мяч, Теннис.
Вариант 13	Море, Континент, Государство, Остров, Суша, Вода;
Вариант 14	Млекопитающие, Птицы, Животное, Рыба, Лев, Сова, Тигр, Крокодил, Акула, Попугай.
Вариант 15	Транспортное средство, Корабль, Пароход, Парусник, Корвет, Капитан, Лодка;
Вариант 16	Кондитерское изделие, Конфета, Карамель, Шоколадная конфета, Печенюшка, Коробка конфет.
Вариант 17	Боец, Охотник, Действия, Шаман, Лучник, Экстрасенс;
Вариант 18	Товар, Камень, Драгоценный камень, Полудрагоценный камень, Рубин, Алмаз, Изумруд, Кремень, Нитка.
Вариант 19	Транспорт, Авиация, Грузовой самолет, Пассажирский, Военный, Ту134, Boeing.
Вариант 2*0*	Фигура, Прямоугольник, Круг, Изменение размера, Элемент управления (интерфейс с методами close, click и т.д.), Bitmap, PictureBox, Window
Вариант 2*1*	Персона, Служащий, Обучающийся (обучение), Работающий, Токарь, Студент, Студент-заочник, Программист
Вариант 22	Персона, Клиент, Адрес, Счет, Операции со счетом, Накопительный, Валютный, Расчетный, Общий, Карта дебетовая и т.д.
Вариант 23	Фигура, Прямоугольник, Оформление, Элемент управления, Кнопка, Меню, Окно.
Вариант 24	Фигура, Прямоугольник, Изменение размера, Элемент управления,Textbox, Окно, Окно просмотра, Кнопка.
Вариант 25	Товар, Техника, Монитор, ПК, Наушники, Проектор, Стол, Экран.
Вариант 26	Товар, Мебель, Диван, Кровать, Шкаф, Шкаф-купе, Вешалка, Тумба, Стул.

Вопросы

1. Для чего используют статические классы?
2. Что может содержать статический класс?
3. Что такое производный и базовый классы?
4. Как используют ключевое слово base?
5. В чем заключена основная задача наследования?
6. Пусть базовый класс содержит метод basefunc(), а производный класс не имеет метода с таким именем. Может ли объект производного класса иметь доступ к методу basefunc()? Если да, то при каких условиях?
7. Напишите объявление конструктора без аргументов для производного класса B, который будет вызывать конструктор без аргументов базового класса A.
8. Что такое полиморфизм? Приведите пример.
9. Определите назначение виртуальных функций.
10. Кому доступны переменные с модификатором protected?
11. Наследуются ли переменные с модификатором private?
12. As, is – что это, как применяется? В чем между ними отличие ?
13. Поддерживает ли C# множественное наследование?
14. Можно ли запретить наследование от класса?
15. Можно ли разрешить наследование класса, но запретить перекрытие метода?
16. Что такое абстрактный класс?
17. В каком случае вы обязаны объявить класс абстрактным?
18. В чем разница между абстрактными и виртуальными классами? Между виртуальными и абстрактными методами?
19. Какие компоненты класса могут быть виртуальными?
20. Что такое интерфейс?
21. Что может содержать интерфейс?
22. Как работать с объектом через унаследованный интерфейс?
23. Приведите пример явной реализации интерфейса.
24. Почему нельзя указать модификатор видимости для методов интерфейса?
25. Можно ли наследовать от нескольких интерфейсов?
26. Назовите отличия между интерфейсом и абстрактным классом.
27. Для чего используются стандартные интерфейсы ICloneable, IComparable, IComparer, lEnumerable?
28. В какой строке приведенного ниже фрагмента листинга не содержится ошибки?

```
class A
      {
      public virtual abstract void m() { } //1
      public virtual void g() { }//2
      public virtual new new void f() { }//3
      public static virtual void h() { }//4
       }
```

29. Что будет выведено на консоль в результате выполнения следующего фрагмента?

```
    class A
{
    public int x = 1;
}
class B : A
{
    public new int x = 2;
    public void m(int a, int b)
    {
        x = a;
        base.x = b;
        Console.Write(x + " " + base.x);
```

```
        }
    }
    class Test
    {
        static void Main(string[] args)
        {
            A a = new A();
            B b = new B();
            b.m(3, 4);

        }
}
```

30. Что будет выведено на консоль в результате выполнения следующего фрагмента кода?

```
      class A
    {
        public class B : A
        {
            public override void mA()
            {
                Console.WriteLine("B ");
            }
        }
        public virtual void mA()
              {
                  Console.WriteLine("A ");
              }
    }
    class Program
    {
        static void Main(string[] args)
        {
            A a = new A();
            A.B b = new A.B();
            a.mA();
            b.mA();

        }
     }
```

31. Чем может быть M4 если дано следующее определение:

```
public class C1 : M1, M2 { }
public struct S1 : M3, M4 { };
```

Варианты ответа:

1) M4 - только интерфейс
2) M4 - интерфейс или класс
3) M4 - только класс
4) M4 - только структура
5) M4 - делегат

32. Выберите верное присваивание для объектов, определенных в листинге.

```
class A { }
        class B : A { }
        class C : B { }
        class D { }
        class Test
        {
            static void Main(string[] args)
            {
                A a = new A();
                B b = new B();
                C c = new C();
                D d = new D();

            }
          }
```

Варианты ответа:

1) b = a;
2) a = b;
3) c = a;
4) d = a;
5) c = b;

33. Что будет выведено на консоль в результате выполнения следующего фрагмента, если раскомментировать строчку 1?

```
public abstract class A
        {
            public virtual void method()
            { Console.Write("A "); }
        }

        public class B : A
        {
            public override void method()
            {  // base.method(); // 1
                // this.method();  // 2
                Console.Write("B ");
            }
        }
        class Program2
        {
            static void Main(string[] args)
        {
            A my = new B();
            my.method();

        }
         }
```

Варианты ответа:

1) B
2) A
3) A B
4) B A
5) 0

34. В какой строке приведенного ниже фрагмента листинга содержится ошибка?

```
        public abstract class A
        {
            public virtual string m() { return "A"; }//1
        }
        public class B : A
        {
            public override new string m() { return "B"; }//2
        }
        public class C : B
        {
            public string m() { return "C"; }//3
        }
        class Program
        {
            static void Main(string[] args)
            {
                A ac = new C(); //4
                Console.WriteLine(ac.m());

            }
         }
```

35. Почему приведенный ниже фрагмент листинга содержит ошибку?

```
abstract class Student //1
        {
            public int Age { get; set; } //2
            public string Name { get; set; } //3
        }

        static void Main(string[] args)
        {
            Student Olga = new Student();//4

        }
```

36. В какой строке может быть ошибка компиляции?

```
class A{}
        class B : A { }
        class C : A { } //1
        class Program4
        {
            static void Main()
            {
                A one = new B(); //2
                A two = new C(); //3
                one = two;  //4
            }
        }
```

37. Что будет выведено на консоль в результате выполнения следующего фрагмента листинга:

```
interface Interface1
        {
            void f();
            void g();
        }
        class A
        {
            public void f() { System.Console.WriteLine("F"); }
            public void g() { System.Console.WriteLine("G"); }
        }
        class B : A, Interface1
        {
            new public void g() { System.Console.WriteLine("new G"); }
        }
        class Program5
        {
            static void Main(string[] args)
            {
                //Interface1 obj = new B();
                //obj.g();
                B obj = new B();
                obj.g();
            }
        }
```

.