컴퓨터정보과
C# 프로그래밍

2021년도/1학기/11주차
장은미

상속

- 클래스 사이의 부모 자식 관계를 정의하는 작업 p.335
  - 기본 클래스 : base class
  - 파생 클래스 : derived class
- base 키워드 p.338
- 접근제한자 protected p.337/339

```
class A
{
   private     int a;
   protected   int b;
   public      int c;
   public void PrintA()
   {
         Console.WriteLine( this.a );
         Console.WriteLine( this.b );
         Console.WriteLine( this.c );
   }

}
class B : A
{
   public void PrintB()
   {
         Console.WriteLine( base.a );
         Console.WriteLine( base.b );
         Console.WriteLine( base.c );
   }
}
class C
{
   public void PrintC()
   {
         B classb = new B();
         Console.WriteLine( classb.a );
         Console.WriteLine( classb.b );
         Console.WriteLine( classb.c );
   }

}
```

다형성

- 하나의 클래스가 여러 형태로 변환될 수 있는 성질 p.341
- 최상위 클래스 : Object p.344
- is 키워드 p.346
- as 키워드 p.349

```
List<Animal> animals = new List<Animal>();

animals.Add(new Cat());

animals.Add(new Dog());

foreach(var ani in animals)

{

    ani.Eat();

    ani.Sleep();

   //ani.Meow();

    if(ani is Cat)((Cat)ani).Meow();

    else if(ani is Dog)((Dog)ani).Bark();


    //ani.Bark();

 var dog = ani as Dog;

    if(dog != null)

    {

       dog.Bark();

    }else{

       var cat = ani as Cat;

       if(cat != null)

          cat.Meow();

    }}
```

base 클래스의 생성자

- derived 클래스의 인스턴스 생성시, base 클래스가 가지고 있는 멤버를 초기화 하기 위해 base 클래스의 생성자도 자동으로 호출 p.351
- base()

이름 충돌

- Overrloading
  - 동일한 메소드 명을 사용하는 방법
- **Overriding**
  - **base 클래스에서 상속받은 메소드를 재정의하는 방법 p.361**
- Shadowing
  - 특정한 영역에서 이름이 겹쳐서 다른 변수를 가리는 것
- **Hiding**
  - **base 클래스에서 상속 받은 메소드를 감추는 방법 p359/p.361**
- Obscuring
  - 변수/클래스/네임스페이스가 동일한 이름을 사용해 모호해지는 것

shadowing

```
class Test : Object
    {
        public int age = 10;
        public Test() : base()
        {
            int age = 20;
            Console.WriteLine("Test():" + age);
            Console.WriteLine("Test():" + this.age);
        }
    }
```

obscuring

```
public class Files
    {
        string System; //(1)
        public void Test()
        {
            System.Console.WriteLine(System)
             //-> (1)으로 인해 Namespace System에 접근이 안됨
        }
    }
```

```
namespace ConsoleApp2
{
    class Test : Object
    {
        public int age = 10;
        public Test() : base()
        {
            int age = 20;
            Console.WriteLine("Test():" + age);
        }
    }

    class Derived : Test
    {
        public new int age = 30;
        public Derived() : base()
        {
            Console.WriteLine("Derived():" + base.age);
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            Derived d = new Derived();

            object c = new List<string>();
            //c.Add("aaa");에러
            ((List<string>)c).Add("aaa");

            Test t = new Test();
            Console.WriteLine(t.ToString());

            object a = 10;
            Console.WriteLine(a.ToString());
        }
    }
}
```

sealed