CHAPTER 05
클래스 기본

학습 목표

- 클래스와 관련된 기본적인 용어를 이해한다.
- 클래스를 사용하는 방법을 익힌다.
- 클래스를 생성하는 방법을 익힌다.
- 인스턴스 변수와 클래스 변수를 만드는 방법을 익힌다.

연습문제

01 다음 문장이 맞다면 O, 틀리다면 X 하시오.

① 클래스 이름은 대문자로 시작하는 것이 관례이다. O

② List 클래스는 크기가 고정적인 배열을 쉽게 사용하기 위한 클래스이다. X→List 클래스는 배열과 다르게 크기가 고정적이지 않습니다.

③ Math.PI 메서드는 pi 값을 찾을 때 사용한다. X→List 클래스는 배열과 다르게 크기가 고정적이지 않습니다.

④ 클래스를 직접 만들 때는 Class 키워드를 사용한다. X→대소문자를 구분하므로, class 키워드입니다.

⑤ 클래스 내부에 클래스를 만들 수도 있다. O

⑥ 클래스 이름으로 곧바로 사용하는 변수와 메서드를 클래스 변수와 클래스 메서드라고 한다. O

⑦ 리스트에서 요소를 제거할 때는 일반적으로 역 반복문을 사용해야 한다. O

⑧ 윈도 폼에서 디자인에서 드래그해서 요소를 만드는 방법을 동적으로 요소를 생성한다고 표현하고, 코드에서 만드는 방법을 정적으로 요소를 생성한다고 표현한다. X→반대입니다.

02 다음 코드에서 클래스와 인스턴스를 구분해 표시하시오.

① List<int> list = new List() 클래스: List, 인스턴스: list

② Car car = new Car() 클래스: Car, 인스턴스: car

③ Product product = new Product() 클래스: Product, 인스턴스: product

④ Dictionary<int, string> dictionary = new Dictionary<int, string>()
클래스: Dictionary, 인스턴스: dictionary

03 다음 중 클래스를 선언하는 위치로 옳지 않은 곳은?

```
class Question { } ❶

class Program
{
    class Question { } ❷

    static void Main(string[] args)
    {
        class Question { } ❸
    }
}
```

04 다음 중에서 '복잡한 자료, 모듈, 시스템 등으로부터 핵심적인 개념 또는 기능을 간추려 내는 것을 말한다'에 해당하는 용어는 무엇인가?

① 상속 ② 다형성 ③ 캡슐화 ④ 추상화

05 다음과 같은 변수를 가지는 클래스를 만들고, 값을 넣어 인스턴스를 생성하시오. 변수의 이름과 자료형은 알맞다고 생각하는 방식으로 선언하시오. 클래스의 이름은 Unit이다.

변수	값
이름	건설 로봇
미네랄	50
보급품	1
생명력	45
공격력	5

```
class Unit {
  public string Name;
  public int Mineral;
  public int Supply;
  public int Hp;
  public int Attack;
}

Unit scv = new Unit() {
  Name = "건설 로봇",
  Mineral = 50,
  Supply = 1,
  Hp = 45,
  Attack = 5
};
```

06 다음과 같은 변수를 가지는 클래스를 만들고, 값을 넣어 인스턴스를 생성하시오. 변수의 이름과 자료형은 알맞다고 생각하는 방식으로 선언하시오. 클래스의 이름은 Book이다.

변수	값
이름	PHP 프로그래밍 입문
초판 발행	2013년 5월 20일
지은이	황재호
펴낸이	김태헌
펴낸곳	한빛아카데미(주
책임편집	김현용
기획	김이화
편집	김이화
디자인	여동일

```
class Book
{
 public string name;
 public DateTime publishedDate;
 public string author;
 public string owner;
 public string publisher;
 public string seniorEditor;
 public string producer;
 public string editor;
 public string designer;
}
Book book = new Book()
{
 name = "PHP 프로그래밍 입문",
 publishedDate = new DateTime(2013, 5, 20),
 author = "황재호",
 owner = "김태헌",
 publisher = "한빛아카데미(주)",
 seniorEditor = "김현용",
 producer = "김이화",
 editor = "김이화",
 designer = "여동일"
};
```

연습문제

07 다음과 같은 변수를 가지는 클래스를 만들고, 값을 넣어 인스턴스를 생성하시오. 변수의 이름과 자료형은 알맞다고 생각하는 방식으로 선언하시오. Person과 Pet이라는 클래스를 2개 만들고, 두 클래스가 연관 관계를 갖게 만든다. Pet 클래스의 인스턴스는 다음과 같이 2개 만든다.

변수	이름
이름	구름
나이	7

변수	이름
이름	별
나이	1

Person 클래스의 인스턴스는 다음과 같이 1개 만든다.

변수	이름
이름	윤인성
주소	서울특별시 강서구
반려 동물	List〈Pet〉 자료형으로서 위의 구름과 별을 가지게 구성

```
class Pet
{
  public string name;
  public int age;
}

class Person
{
  public string name;
  public string address;
  public List<Pet> pets;
}

static void Main(string[] args)
{
  Pet cloud = new Pet() { name = "구름", age = 7 };
  Pet star = new Pet() { name = "별", age = 1 };

  Person person = new Person()
  {
    name = "윤인성",
    address = "서울특별시 강서구",
    pets = new List<Pet>() { cloud, star }
  };
}
```

08 임의의 실수를 사용해 다음과 같은 숫자 맞히기 프로그램을 만들어보시오.

```
C:₩WINDOWS₩system32₩cmd.exe
숫자를 입력해주세요: 200
200보다는 작은 숫자입니다.

숫자를 입력해주세요: 100
100보다는 큰 숫자입니다.

숫자를 입력해주세요: 150
150보다는 작은 숫자입니다.

숫자를 입력해주세요: 125
125보다는 큰 숫자입니다.

숫자를 입력해주세요: 130
130보다는 작은 숫자입니다.

숫자를 입력해주세요: 126
정답입니다!
```

```
Random random = new Random();
int answer = random.Next();
while (true)
{
 Console.Write("숫자를 입력해주세요: ");
 int input = int.Parse(Console.ReadLine());
 if (input > answer)
 {
 Console.WriteLine(input + "보다는 작은 숫자입니다.");
 }
 else if (input < answer)
 {
 Console.WriteLine(input + "보다는 큰 숫자입니다.");
 }
 else
 {
 Console.WriteLine("정답입니다...!");
 break;
 }
 Console.WriteLine();
}
```

09 다음과 같은 프로그램을 만들 때에 필요하다고 생각되는 클래스를 추상화해보시오.

① 치킨 집 검색 프로그램

② 주차장 관리 프로그램

③ 도서 관리 프로그램

④ 소셜 네트워크 서비스

① 치킨집 클래스 - 이름, 주소, 전화번호, 좌표(위도와 경도), 영업 시간 등등
② 주차 정보 클래스 - 자동차 번호, 차 종류, 입차 시간, 출차 시간 등등
③ 책 클래스 - 책 제목, 저자, 출판사 등등
회원 클래스 - 회원 이름, 가입일 등등
대출 정보 클래스 - 책 제목, 회원 이름, 대출 일자 등등
④ 사용자 클래스 - 이름, 닉네임, 지역 등등
글 클래스 - 작성자, 작성 일자, 내용 등등
등등 해당 내용과 관련된 것이라면 답으로 처리

CHAPTER 06

메서드

학습 목표

- 메서드의 매개변수와 반환을 이해한다.
- 인스턴스 메서드와 클래스 메서드 생성 방법을 익힌다.
- 오버로딩을 이해한다.
- 접근 제한자를 이해한다.
- 생성자와 소멸자의 생성 방법을 익히고 호출 시점을 이해한다.
- 속성을 사용하는 이유를 이해하고 그 사용 방법을 익힌다.

01 다음 문장이 맞다면 O, 틀리다면 X 하시오.

① 메서드에서 아무 것도 리턴하지 않을 때는 반환형(리턴 자료형)을 Null로 입력한다. X→void입니다.

② 메서드는 여러 개의 값을 매개 변수로 받을 수도 있고, 반환할 수도 있다 X→매개변수는 여러 개 받을 수 있지만, 값은 하나만 반환할 수 있습니다.

③ 같은 이름으로 매개 변수가 다른 메서드를 여러 개 만드는 것을 오버라이드(override)라고 부른다. X→오버로딩(overloading)입니다.

④ private 접근 제한자가 붙은 변수와 메서드는 해당 클래스 내부에서도 외부에서도 어떤 경우에도 접근할 수 없다. X→해당 클래스 내부에서는 사용할 수 있습니다.

⑤ 생성자에는 public 접근 제한자만 붙일 수 있다. private 접근 제한자를 붙이면 오류가 발생한다. X→일반적으로 그런 것이지, private 접근 제한자를 붙일 수는 있습니다.

⑥ 객체를 소멸시키고 싶을 때, 소멸자를 직접 호출한다. X→소멸자는 자동적으로 호출됩니다.

⑦ 소멸자는 하나만 만들 수 있다. O

⑧ C# 개발자들은 일반적으로 클래스 변수의 이름은 대문자로 시작하게 하고, 속성 이름은 소문자로 시작하게 해서 구분한다. X→둘 다 대문자로 시작하게 합니다.

⑨ 윈도 폼의 이벤트 메서드에서 매개변수 sender는 이벤트를 발생시킨 객체를 의미한다. O

02 다음 중 생성자에 대한 설명으로 옳지 않은 것은?

① 생성자 이름은 클래스 이름과 같다.

② 생성자는 반환과 관련된 선언을 하지 않는다.

③ 생성자는 인스턴스가 생성될 때 자동으로 호출되는 메서드이다.

④ 클래스는 반드시 개발자가 직접 정의해야 한다.

03 메서드를 오버로딩할 때 달라도 되는 메서드의 구성 요소가 아닌 것을 모두 고르시오.

① 반환형 ② 매개변수의 개수 ③ 매개변수의 자료형 ④ 메서드 이름

04 Sample이라는 클래스가 있을 때, 다음 중에서 올바른 소멸자의 이름을 고르시오.

① !Sample ② ~Sample ③ ^Sample ④ &Sample

연습문제

05 다음 중 소멸자에 붙여야 하는 올바른 접근 제한자를 고르시오.

① public ② private ③ protected ④ 없음

06 다음 코드의 실행 결과를 예측하시오.

```
class Program
{
    static int Test(int A)
    {
        return 10;
    }
    static int Test(long A)
    {
        return 20;
    }
    static int Test(float A)
    {
        return 30;
    }
    static int Test(double A)
    {
        return 40;
    }

    static void Main(string[] args)
    {
        Console.WriteLine(Test(52273));
        Console.WriteLine(Test(52.273));
    }
}
```

10
40

07 다음 코드의 실행 결과를 예측하시오.

```
class A
{
	public A()
	{
		Console.WriteLine("A의 생성자");
	}

	~A()
	{
		Console.WriteLine("A의 소멸자");
	}
}

class B : A
{
	public B()
	{
		Console.WriteLine("B의 생성자");
	}

	 ~B()
	{
		Console.WriteLine("B의 소멸자");
	}
}

class Program
{
	static void Main(string[] args)
	{
		new B();
	}
}
```

"A의 생성자"
"B의 생성자"
"B의 소멸자"
"A의 소멸자"

연습문제

08 다음 코드의 실행 결과를 예측하시오.

```
class MyMath
{
	static int Abs(int input)
	{
		return input > 0 ? input : -input;
	}

	 static double Abs(int input)
	{
		return input > 0 ? input : -input;
	}

	 static long Abs(long input)
	 {
		return input > 0 ? input : -input;
	 }

	 static double Abs(long input)
	{
		return input > 0 ? input : -input;
	}
}
```

같은 이름과 매개 변수를 가진 메서드를 만들었다.

CHAPTER 07

상속과 다형성

학습 목표

- 상속과 다형성을 사용하는 이유를 이해한다.
- 상속과 다형성으로 클래스 변환하는 방법을 익힌다.
- 섀도잉, 하이딩, 오버라이딩을 이해한다.
- 상속과 오버라이딩을 제한하는 방법을 익힌다.

01 다음 빈칸을 채우시오.

① (상속)은 클래스 사이에 부모 자식 관계를 정의하는 작업이다.

② C#에서 상속을 할 때는 (:) 기호를 사용한다.

③ 부모에서 (public)과 (protected) 접근 제한자를 지정한 멤버는 자식에서 접근할 수 있다.

④ 부모에서 (private) 접근 제한자를 지정한 멤버는 자식에서 접근할 수 없다.

⑤ 자식 클래스에서 부모 클래스를 가리킬 때는 (base) 키워드를 사용한다.

⑥ 특정한 영역에서 이름이 겹쳐서 다른 변수를 가리는 것을 (섀도잉)이라고 부른다.

⑦ 상속을 제한할 때는 (sealed) 키워드를 사용한다.

⑧ 메서드 오버라이드를 막을 때는 (sealed) 키워드를 사용한다.

⑨ (abstract) 키워드를 붙인 클래스는 인스턴스를 만들 수 없다.

02 다음 문장이 맞다면 O, 틀리다면 X 하시오.

① 상속을 할 때는 부모가 더 많은 기능을 갖는다. X: 일반적으로 자식이 더 많이 갖게 된다

② C#의 모든 객체는 최상위 객체 Object의 상속을 받는다. O

③ 자료형을 확인할 때는 is 키워드를 사용할 수 있다. O

④ 자료형을 변환할 때는 as 키워드를 사용할 수 있다. O

⑤ as 키워드로 자료형 변환을 하면, 변환에 실패했을 때 예외가 발생한다. X

⑥ 메서드를 하이딩 할 때는 new 키워드, 오버라이딩할 때는 override 키워드를 사용한다. O

⑦ 오버라이딩 할 때는 부모의 메서드에 virtual 키워드, 자식의 메서드의 override 키워드를 붙인다. O

03 다음 중 자료형 변환과 관련된 키워드는 무엇인가?

① as　② to　③ select　④ from

04 다음 접근 제한자를 허용 순서가 좁은 것부터 넓은 순서로 나열하시오. ②①③

① protected　② private　③ public

05 다음 코드에서 예외가 발생하는 부분을 고르시오.

```
class Unit { }
class Tank : Unit { }

class Program
{
    static void Main(string[] args)
    {
        Unit unit = new Unit();
        Tank tank = new Tank();

        /* ❶ */ Unit a = (Unit)unit;
        /* ❷ */ Unit b = (Unit)tank;
        /* ❸ */ Tank c = (Tank)unit;
        /* ❹ */ Tank d = (Tank)tank;
    }
}
```

06 다음 코드는 10을 출력하는가? 20을 출력하는가?

```
class Parent
{
    public int question = 10;
}

class Child : Parent
{
    public string question = "20";
}

class Program
{
    static void Main(string[] args)
    {
        Child child = new Child();
        Console.WriteLine(child.question);
    }
}
```

20

07 위의 06번 문제에서 출력되지 않은 값을 출력하게 하려면 코드를 어떻게 수정해야 하는가?

```
Child child = new Child();
Console.WriteLine(((Parent) child).question);
또는
Parent child = new Child();
Console.WriteLine(child.question);
```

08 다음 코드의 실행 결과를 예측하시오.

```
class Parent
{
    public int Question() { return 10; }
}

class Child : Parent
{
    public int Question() { return 20; }
}

class Program
{
    static void Main(string[] args)
    {
        Child child = new Child();
        Console.WriteLine(child.Question());
    }
}
```

20

09 다음 코드의 실행 결과를 예측하시오.

```
class Parent
{
    public int Question() { return 10; }
}

class Child : Parent
{
    public new int Question() { return 20; }
}

class Program
{
    static void Main(string[] args)
    {
        Child child = new Child();
        Console.WriteLine(child.Question());
    }
}
```

20

연습문제

10 다음 코드의 실행 결과를 예측하시오.

```
class Parent
{
    public virtual int Question() { return 10; }
}

class Child : Parent
{
    public new int Question() { return 20; }
}

class Program
{
    static void Main(string[] args)
    {
        Child child = new Child();
        Console.WriteLine(child.Question());
    }
}
```

20

11 다음 코드의 실행 결과를 예측하시오.

```
class Parent
{
    public virtual int Question() { return 10; }
}

class Child : Parent
{
    public override int Question() { return 20; }
}

class Program
{
    static void Main(string[] args)
    {
        Child child = new Child();
        Console.WriteLine(child.Question());
    }
}
```

20

CHAPTER 08

클래스 심화

학습 목표

- 제네릭을 사용하는 방법을 익힌다.
- 인덱서를 사용하는 방법을 익힌다.
- out 키워드를 사용하는 방법을 익힌다.
- 구조체를 생성하고 사용하는 방법을 익힌다.

연습문제

01 다음 문장이 맞다면 O, 틀리다면 X 하시오.

① 한 클래스에서 제네릭은 하나만 사용할 수 있다. X→여러 개 사용할 수 있습니다.

② 인덱서의 대괄호 안에서는 int 자료형만 사용할 수 있다. X→원하는 자료형을 사용할 수 있습니다.

③ 구조체와 클래스는 차이가 없다. X→여러 가지 차이가 있습니다.

④ 구조체는 매개 변수가 없는 생성자를 만들 수 없다. O

⑤ 구조체의 생성자는 내부에서는 반드시 모든 변수를 초기화해야 한다. O

02 다음 중 제네릭과 관련된 설명 중에 틀린 것을 고르시오.

① 제네릭은 클래스를 선언할 때 이름 옆에 〈〉 괄호를 사용해서 넣을 수 있다.

② of 키워드를 사용해서 제네릭 자료형에 제한을 걸 수 있다.

③ 제네릭은 일반적으로 T라는 식별자를 사용한다.

④ 제네릭은 자료형의 별칭을 지정하는 기능이다.

03 다음 중 out 키워드와 관련된 설명 중에 옳은 것을 고르시오.

① out 키워드를 사용하면, 여러 개의 값을 리턴하는 것과 같은 메서드를 만들 수 있다.

② int.Parse( ), float.Parse( ) 메서드는 out 키워드를 활용한 대표적인 메서드이다.

③ out 키워드는 한 메서드에 하나만 사용할 수 있다.

④ out 키워드를 사용하면 오버로드가 불가능하다.

04 클래스와 구조체의 차이로 옳지 않은 것은?

① 클래스는 매개변수 없는 생성자를 만들 수 있지만, 구조체는 만들 수 없다.

② 클래스는 값 복사가 일어나고, 구조체는 참조 복사가 일어난다.

③ 클래스는 class 키워드로 생성하고, 구조체는 struct 키워드로 생성한다.

④ 클래스는 상속할 수 있지만, 구조체는 상속할 수 없다