12강. 상속

C++ 프로그래밍

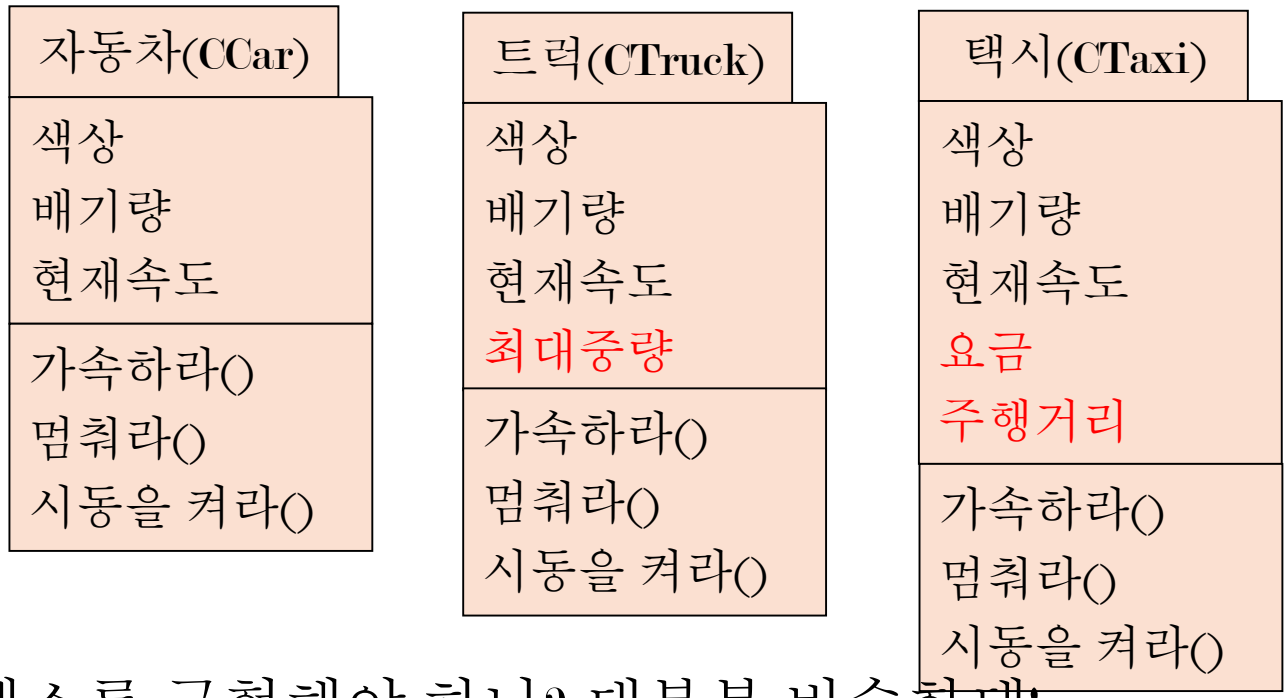
jhhwang@kumoh.ac.kr

컴퓨터공학과 황준하

목차

상속이란?

- 자동차(CCar), 트럭(CTruck), 택시(CTaxi) 클래스를 만들어야 한다면?
  - 클래스 분석

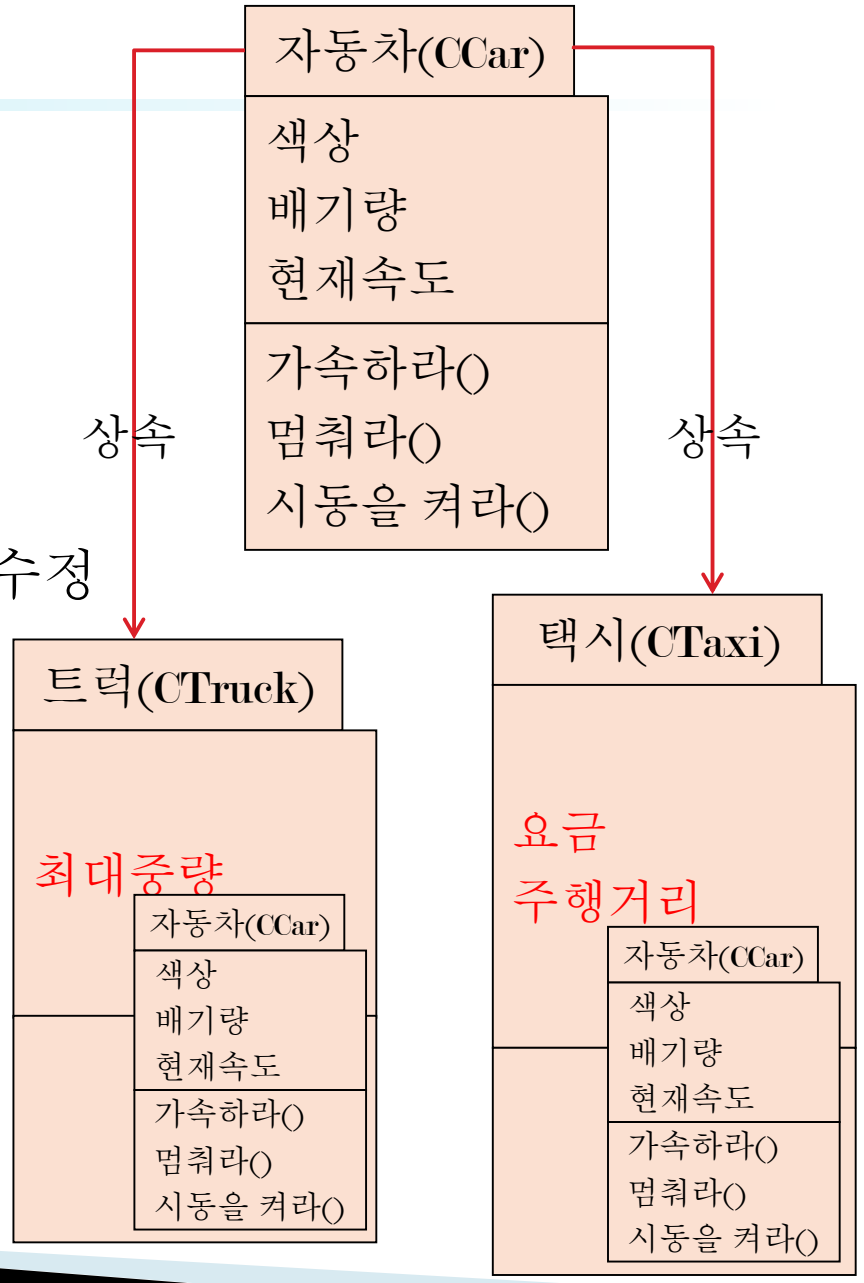


  - 모두 따로 클래스를 구현해야 하나? 대부분 비슷한데!
  - 코드의 중복, 비효율성을 제거하는 방법 → 상속

상속이란?

- 상속의 개념
  - CCar 클래스 작성
  - CTruck, CTaxi 클래스 작성
    - CCar 클래스 상속
    - CCar와 다른 부분만 추가 또는 수정
- 상속 문법

```
class CTruck : public CCar {
private :
        int MaxWeight;

};
```



상속이란?

- 용어 설명
  - CCar 클래스 : base 클래스, 부모 클래스, 기저 클래스
  - CTruck 클래스 : derived 클래스, 자식 클래스, 유도 클래스
  - base, derived 클래스 관계는 상대적인 개념

- 상속 관계가 자연스럽게 성립하는 경우
  - is-a 관계 : A는 B이다.
    - 트럭은 자동차이다 : 자동차-base 클래스, 트럭-derived 클래스
    - 사과는 과일이다 : 과일-base클래스, 사과-derived 클래스
  - 그 외에 기존 클래스의 모든 특성을 상속받고자 할 때

상속 관련 문제 제기

- 상속을 통한 원(CCircle), 구(CSphere) 클래스 구현
  - 클래스 분석

원(CCircle)
중심 좌표 (x, y) 반지름
면적 계산

구(CSphere)
중심 좌표 (x, y, z) 반지름
면적 계산 (표면적) 부피 계산

  - 상속을 이용하지

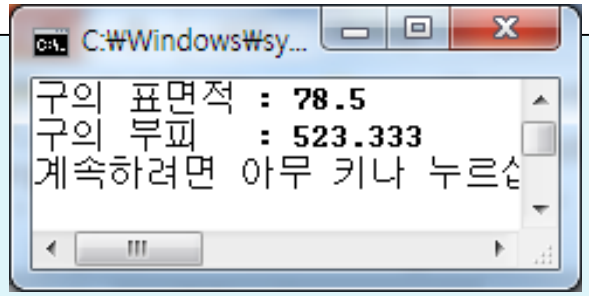
```
#define PI 3.14

class CCircle {
public :
        int x, y;
        double Radius;

public :
        double GetArea() { return (PI * Radius * Radius); }
};
```

일단 public으로 설정

상속 관련 문제 제기

- 상속을 이용한 구(CSphere) 구현

```
class CSphere : public CCircle {
public :
    int z;

public :
    double GetVolume() { return ((4.0/3.0) * PI * Radius * Radius * Radius); }
};

void main(void)
{
    CSphere Sph;
    Sph.x = 1; Sph.y = 1; Sph.z = 1; Sph.Radius = 5;

    cout << "구의 표면적 : " << Sph.GetArea() << endl;
    cout << "구의 부피   : " << Sph.GetVolume() << endl;
}
```



상속을 받았으므로 x, y, Radius, GetArea()
모두 Csphere에서 사용 가능

상속 관련 문제 제기

- 상속 관련 문제 제기
  - 액세스 지정자
    - CCircle 멤버 변수들을 private 영역에 포함시킨다면 → 에러
    - 액세스 지정자의 종류 : public, private, protected
  - 생성자와 소멸자
    - CCircle 클래스와 CSphere 클래스에 생성자와 소멸자가 필요하다면
    - 작성 방법, 호출 방법, 호출 순서
  - CCircle과 CSphere 클래스의 면적(GetArea) 함수
    - CCircle : pi * Radius * Radius
    - CSphere : 4 * pi * Radius * Radius
    - 함수 오버라이딩 : 상속에 있어서 함수명은 같지만 구현 내용 다름

protected 멤버

- 멤버(변수, 함수)에 대한 접근 가능 여부

영역	내부 접근	외부 접근	derived 클래스 접근
public	O	O	O
protected	O	X	O
private	O	X	X

  - protected 멤버
    - 외부 접근 허용하지 않음 : private과 동일
    - derived 클래스에서의 접근 허용 : public과 동일

액세스 지정자 : protected 멤버

- 액세스 지정자

```
class CSphere : public CCircle {  };
```

액세스 지정자 : public, protected, private

- 액세스 지정자와 멤버의 상속

base의 private 멤버 : 상속은 되나 derived에서 접근 불가

액세스 지정자 \ base 멤버	public 포함영역(접근가능)	protected 포함영역(접근가능)	private 포함영역(접근가능)
public 상속	public (O)	protected (O)	private (X)
protected 상속	protected (O)	protected (O)	private (X)
private 상속	private (O)	private (O)	private (X)

액세스 지정자 : protected 멤버

- protected 멤버
  - 상속 시 derived 클래스에서 바로 접근 가능

```
class CCircle {
protected :
    int x, y;
    double Radius;

public :
    double GetArea() { return (PI * Radius * Radius); }
};
```

private 영역에 있을 경우 에러 → derived에서 바로 접근 불가

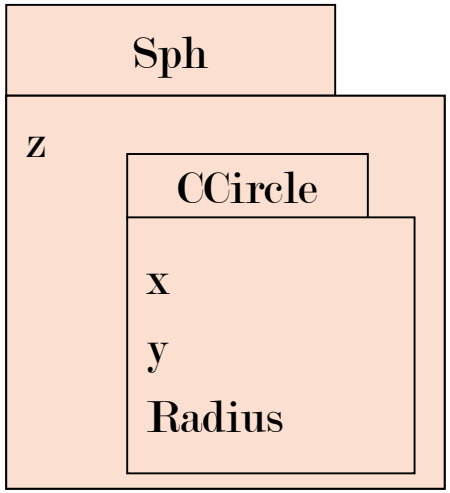
```
class CSphere : public CCircle {
public :
    int z;

public :
    double GetVolume() { return ((4.0/3.0) * PI * Radius * Radius * Radius); }
};
```

protected 멤버에 대한 접근 가능

생성자와 소멸자

- derived 클래스 객체 생성 시 메모리(변수) 생성 구조
  - CSphere Sph;

  1. CCircle 객체 : x, y, Radius
     - CCircle 생성자 몸체 실행
  2. CSphere 변수 : z
     - CSphere 생성자 몸체 실행



- derived 클래스 객체 생성 시 생성자, 소멸자 수행 순서
  - CCircle 클래스 생성자 수행 → CSphere 클래스 생성자 수행
    - 소멸자는 역순 : CSphere 소멸자 → CCircle 소멸자
  - CCircle 클래스의 생성자를 명시적으로 호출하는 방법

생성자와 소멸자

- CCircle 클래스와 CSphere 클래스 생성자

```
class CCircle {
protected :
    int x, y;
    double Radius;

public :
    CCircle(int a, int b, double r) : x(a), y(b), Radius(r) { }
    double GetArea() { return (PI * Radius * Radius); }
};
```

```
void main(void)
{
    CSphere Sph(1, 2, 3, 4);
    cout << Sph.GetVolume() << endl;
}
```

```
class CSphere : public CCircle {
private :
    int z;

public :
    CSphere(int a, int b, int c, double r) : CCircle(a, b, r), z(c) { }
    double GetVolume() { return ((4.0/3.0) * PI * Radius * Radius * Radius); }
};
```

base 클래스 생성자 호출 : 멤버 초기화 구문
없다면 디폴트 생성자 수행 → 여기서는 에러

함수 오버라이딩

- CCircle과 CSphere 클래스의 면적(GetArea) 함수
  - CCircle : pi * Radius * Radius
  - CSphere : 4 * pi * Radius * Radius

- 문제점
  - CSphere에서 CCircle의 GetArea 함수를 상속받아 사용
    - CShpere 클래스에 맞지 않음

→ CSphere 클래스에 맞게 GetArea 함수를 재정의하라!

→ 함수 오버라이딩 (함수 재정의)

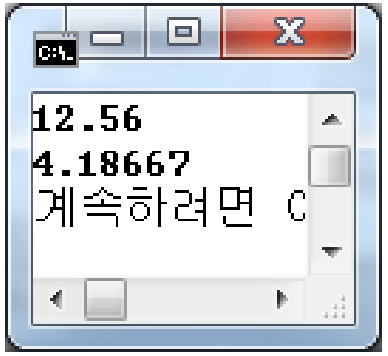
함수 오버라이딩

- CSphere 클래스의 GetArea 함수 오버라이딩

```
class CSphere : public CCircle {
private :
	int z;

public :
	CSphere(int a, int b, int c, double r) : CCircle(a, b, r), z(c) { }
	double GetArea() { return (4 * PI * Radius * Radius); }
	double GetVolume() { return ((4.0/3.0) * PI * Radius * Radius * Radius); }
};
```

```
void main(void)
{
	CSphere Sph(1, 1, 1, 1);

	cout << Sph.GetArea() << endl;
	cout << Sph.GetVolume() << endl;
}
```

CSphere의 GetArea 함수 수행



함수 오버라이딩

- CSphere 클래스의 경우 CCircle 클래스의 GetArea 함수는 사용할 수 없는가?
  - No! 클래스명(CCircle)을 통해 접근 가능

```
class CSphere : public CCircle {
public :
      // double GetArea() { return (4 * PI * Radius * Radius); }
      double GetArea() { return (4 * CCircle::GetArea()); }
};
```

```
void main(void)
{
      CSphere Sph(1, 1, 1, 1);

      cout << Sph.CCircle::GetArea() << endl;
      cout << Sph.GetVolume() << endl;
}
```