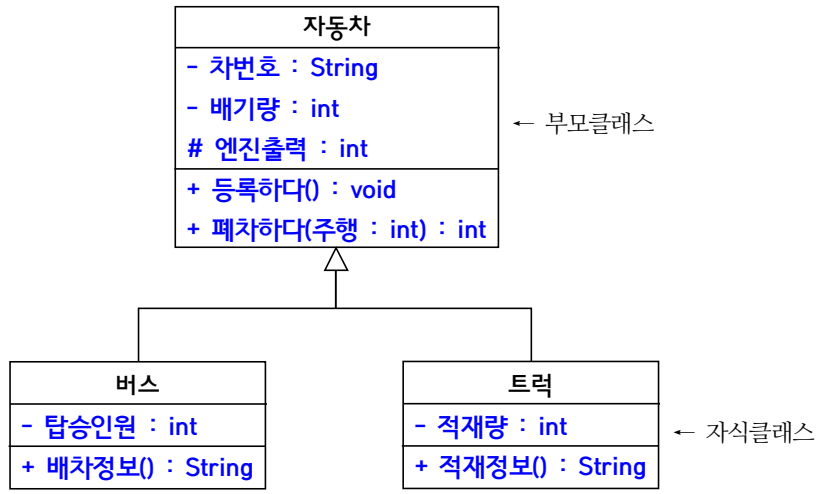
4. 일반화(generalization)

〈일반화(generalization)/세분화(specialization)〉



↓Java 원시코드 생성

〈Java 원시코드〉

```
public class 자동차                          //부모클래스 자동차
{
  private String 차번호;                     //클래스 자동차의 속성
  private int 배기량;
  protected int 엔진출력;
  public void 등록하다() {      }            //클래스 자동차의 메서드
  public int 폐차하다(int 주행) {      }
}
public class 버스 extends 자동차             //자식클래스 버스(자동차를 상속 받음)
{
  private int 탑승인원;                      //클래스 버스의 속성
  public String 배차정보() {      }          //클래스 버스의 메서드
}
public class 트럭 extends 자동차             //자식클래스 트럭(자동차를 상속 받음)
{
  private int 적재량;                        //클래스 트럭의 속성
  public String 적재정보() {      }          //클래스 트럭의 메서드
}
```

- **일반화(generalization)** : 자식클래스가 주체가 되어 자식클래스를 부모클래스로 일반화한다.
- **세분화(specialization)** : 부모클래스가 주체가 되어 부모클래스를 자식클래스로 구체화한다.
- 부모클래스는 자식클래스의 공통 속성이나 연산을 제공하는 틀이다.

기출문제 분석

1. 유사한 클래스들 사이의 공유되는 속성과 동작을 묶어주는 추상화 기법으로 가장 옳은 것은? [2018년 서울 7급]

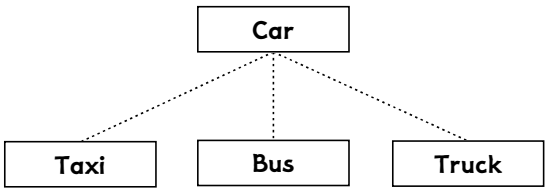
① 집단화(aggregation)　② 추상화(abstraction)
③ 캡슐화(encapsulation)　④ 일반화(generalization)

♤ **일반화(generalization)**

- 유사한 클래스들 사이의 공유되는 속성과 동작을 묶어주는 추상화 기법 : 일반화

정답 : ④

2. UML의 다이어그램에서 관계를 완성하고자 한다. 다음 관계의 표현으로 가장 적합한 것은? [2010년 국가 7급]



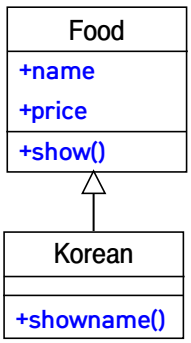
① 연관관계(association)　② 일반화관계(generalization)
③ 집단화관계(aggregation)　④ 의존관계(dependency)

♤ **상속관계 : 세분화(specialization) / 일반화(generalization)**

세분화	• 어떤 개체집합에서 하나 이상의 하위 개체를 표시하는 과정을 세분화라 한다. • 즉, 하나의 개체집합을 다수의 하위 레벨 개체집합으로 분리하는 것이다. • 하향식(top-down) 설계 방식으로 is-a 관계에 기반을 둔다. • **세분화**는 특수화, 구체화, 상세화, 전문화라고도 한다.
일반화	• 일반화는 개체집합은 몇 가지 공통된 특성을 가진다는 것에서 시작한다. • 일반화는 공통성을 근거로 개체집합에서 하나의 상위 개체집합으로 통합한다. • 공통된 속성들이 하위 개체집합에서 반복 표현되지 않도록 설계한다. • 상향식(bottom up) 설계 방식으로 is-a 관계에 기반을 둔다. • 일반화는 세분화의 역방향 설계이다. • 상위클래스는 하위 객체의 공통 특징을 가진다.

정답 : ②

3. 다음 클래스 다이어그램을 파이썬 코드로 표현한 것이다. (가)~(라)에 들어갈 내용으로 옳은 것은? [2022년 국가 7급]




```
class (가):
    def __init__(self, name, price):
        self.name = name
        self.price = price
    def (나)(self):
        print("{}'s price = {}".format(self.name, self.price))
class (다)( (가) ):
    def __init__(self, name, price):
        Food.__init__(self, name, price)
    def (라)(self):
        print("{}".format(self.name))
```

	(가)	(나)	(다)	(라)
①	Food	show	Korean	showname
②	Korean	showname	Food	show
③	Food	showname	Korean	show
④	Korean	show	Food	showname

♤ 클래스 다이어그램 – 파이썬 코드

```
class Food:                                  #부모클래스
    def __init__(self, name, price):        #초기화(객체 생성 시 자동호출) - 생성자 역할
        self.name = name
        self.price = price
    def show(self):                          #메서드 정의
        print("{}'s price = {}".format(self.name, self.price))
class Korean(Food):                          #상속 : 자식클래스(부모클래스)
    def __init__(self, name, price):        #초기화(객체 생성 시 자동호출) - 생성자 역할
        Food.__init__(self, name, price)     #부모 생성자 호출
    def showname(self):                      #메서드 정의
        print("{}".format(self.name))
```

정답 : ①