JAVA 프로그램 실습

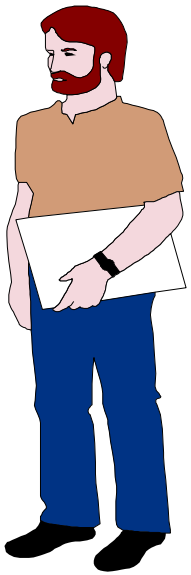
Class 사용하기

경복대학교
소프트웨어융합과
배희호 교수

Class 문제 1

- 다음 객체에 대한 설명에 맞는 클래스를 만들고 값을 출력해 보세요.

*나이*는 30살, *이름*은 홍길동이라는 *남자*가 있습니다.
이 남자는 *결혼*을 했고 *자식*이 셋 있습니다



홍길동

Class 문제 1

- 관심사항을 파악하자 (Data)
  - 명사 (속성)
- Class를 만들어보자
  - Member 변수
- People 클래스를 정의

People
-name : String -age : int -gender : char -isMarried : boolean -children : int
+toString() : String

Class 문제 1

- People.JAVA

```
public class People {
    private String name;
    private int age;
    private char gender;
    private boolean isMarried;
    private int children;

    public People(String name, int age, char gender, boolean isMarried,
                                                        int children) {
        this.name = name;
        this.age = age;
        this.gender = gender;
        this.isMarried = isMarried;
        this.children = children;
    }
```

Class 문제 1

- People.JAVA

```
    @Override
    public String toString() {
        String result = "";
        result = "이름 : " + name;
        result += "₩n나이 : " + age;
        result += "₩n성별 : " + gender + "자";
        result += "₩n결혼 여부 : " + (isMarried ? "기혼" : "미혼");
        result += "₩n자녀 수 : " + children+ '명';
        return result;
    }
}
```

Class 문제 1

- Main.JAVA

```
public static void main(String[] args) {
        People people = new People("홍길동", 40, '남', true, 3);

        System.out.println(people);
}
```

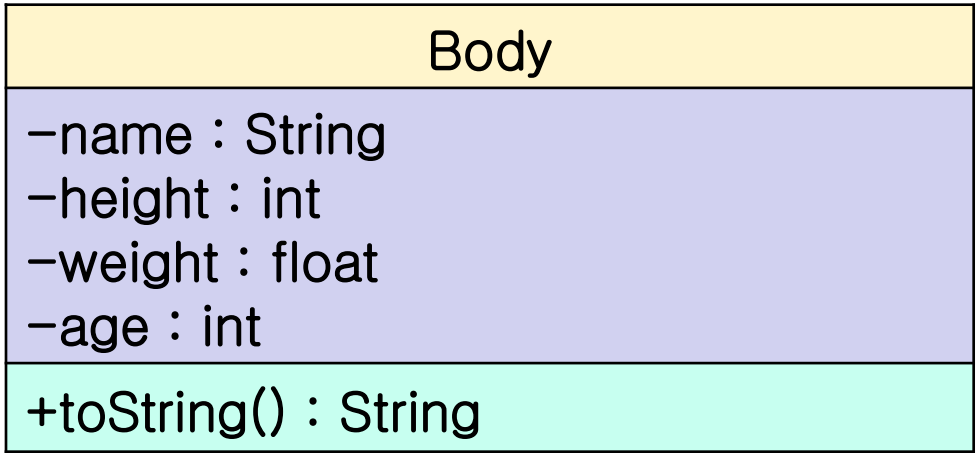
Class 문제 2

- 다음 객체에 대한 설명에 맞는 클래스를 만들고 값을 출력해 보세요.

*김철수*는 *키*가 183Cm이고, *몸무게*는 64.3Kg이고,
*나이*는 21살 입니다

Class 문제 2

- 관심사항을 파악하자 (Data)
  - 명사 (속성)
- Class를 만들어보자
  - Member 변수
- Body 클래스를 정의



Class 문제 2

- Body.JAVA

```
public class Body {
    private String name;
    int age;
    protected int height;
    private float weight;

    public Body(String name, int age, int height, float weight) {
        this.name = name;
        this.age = age;
        this.height = height;
        this.weight = weight;
    }

    public void setWeight(float weight){
        this.weight= weight;
    }
```

Class 문제 2

- Body.JAVA

```
    public float getWeight(){
        return weight;
    }

    @Override
    public String toString() {
        String result = "";
        result = "이름 : " + name;
        result += "₩n나이 : " + age;
        result += "₩n키 : " + height + "Cm";
        result += "₩n몸무게 : " + weight +"Kg";

        return result;
    }
}
```

Class 문제 2

- Main.JAVA

```
public static void main(String[] args) {
    Body kim = new Body("김철수", 21, 183, 64.3f);

    kim.setWeight(64.7f);
    kim.age++;

    System.out.print(kim);
}
```

Class 문제 3

- 쇼핑몰에 주문이 들어왔다. 주문 내용을 구현할 수 있는 클래스를 만들고 인스턴스로 생성한 후 해당 내용을 입력 받아 아래와 같은 형식으로 출력하는 프로그램을 작성해보자

```
주문 번호 : 202201234
주문자 아이디 : bae1234
주문 날짜 : 2022-03-24
주문자 이름 : 홍길동
주문 상품 번호 : PD-34-ABC
배송 주소 : 경기도 남양주시 진접읍 경복대로
주문 번호 : 202201234
주문자 아이디 : bae1234
주문 날짜 : 2022-03-24
주문자 이름 : 홍길동
주문 상품 정보 : PD-34-ABC
배송 주소 : 경기도 남양주시 진접읍 경복대로
```

Class 문제 3

- Shopping 클래스

Shopping
-orderNumber : String -id : String -date : String -name : String -productNumbe : String -address : String
+Setter/Getter +toString() : String

Class 문제 3

- Shopping.JAVA

```
public class Shopping {
    private String orderNumber;        //주문 번호
    private String id;                 //주문자 ID
    private String date;               //주문 날짜
    private String name;               //주문자 이름
    private String productNumber;      //주문 상품 번호
    private String address;            //배송 주소

    public void setOrderNumber(String orderNumber) {
        this.orderNumber = orderNumber;
    }
    public void setAddress(String address) {
        this.address = address;
    }

    public void setDate(String date) {
        this.date = date;
    }
```

Class 문제 3

- Shopping.JAVA

```
    public void setName(String name) {
        this.name = name;
    }
    public void setId(String id) {
        this.id = id;
    }
    public void setProductNumber(String productNumber) {
        this.productNumber = productNumber;
    }
    @Override
    public String toString() {
        return "주문 번호 : " + orderNumber +"₩n" +
                "주문자 아이디 : " + id + "₩n" +
                "주문 날짜 : " + date +"₩n" +
                "주문자 이름 : " + name + "₩n" +
                "주문 상품 정보 : " + productNumber + "₩n" +
                "배송 주소 : " + address;
    }
}
```

Class 문제 3

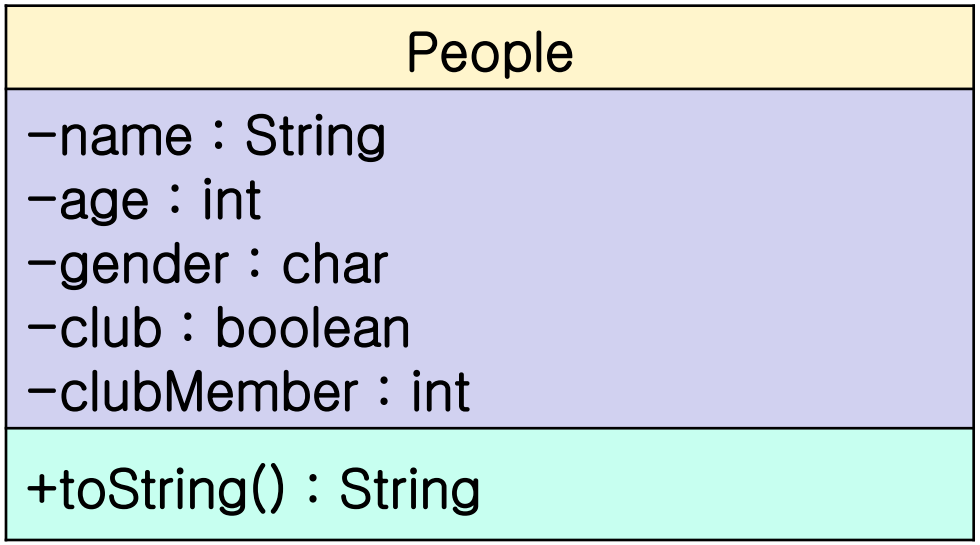
- Main.JAVA

```
public class Main {
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        Shopping user = new Shopping(); //객체 생성
        System.out.print("주문 번호 : ");
        user.setOrderNumber(keyboard.nextLine());
        System.out.print("주문자 아이디 : ");
        user.setId(keyboard.nextLine());
        System.out.print("주문 날짜 : ");
        user.setDate(keyboard.nextLine());
        System.out.print("주문자 이름 : ");
        user.setName(keyboard.nextLine());
        System.out.print("주문 상품 번호 : ");
        user.setProductNumber(keyboard.nextLine());
        System.out.print("배송 주소 : ");
        user.setAddress(keyboard.nextLine());
        System.out.println(user);
    }
}
```

Class 문제 4

- 해당 내용을 Class와 Object를 활용하여 생성하여 보자

나이(age)가 27살, 이름(name)이 '이순신'이라는 남자가 있다. 이 남자는 클럽(club)에 참여 하였고, 클럽 인원(clubMember)은 5명이다.



Class 문제 4

```
public class People {
    public int age; //나이
    public String name; //이름
    public boolean club; //클럽 참여 여부
    public int clubMember; //클럽 인원

    public void peoplePrint() {     //객체 정보 출력
        System.out.println("나이: " + age);
        System.out.println("이름: " + name);
        System.out.println("클럽 참여 여부: " + club);
        System.out.println("클럽인원:" + clubMember);
    }
}
```

Class 문제 4

```
public static void main(String[] args) {
    People people = new People(); //객체 생성
    people.age = 27;
    people.name = "peemang";
    people.club = true;
    people.clubMember = 5;

    people.peoplePrint(); //객체 정보 출력
}
```

Class 문제 5

- 다음과 같이 출력하는 프로그램을 만들어보자

나의(홍길동) 자동차는 2001년식 현대자동차에서
생산한 Grandure로 현재 23,000Km를 운행했습니다.

  - Car 클래스를 작성하여라.
  - CarModel 클래스를 작성하여라.

Car
-owner : String -CarModel model : CarModel -mileage : int
+toString() : String

CarModel
-modelName : String -company : String -year : int
+toString() : String

Class 문제 5

- CarModel.JAVA

```
public class CarModel {
    private String modelName;
    private String company;
    private int year;

    public CarModel(String modelName, String company, int year) {
        this.modelName = modelName;
        this.company = company;
        this.year = year;
    }

     @Override
    public String toString() {
        return "모델명 : " + modelName + "₩n" +
                "제조사 : " + company + "₩n" +
                "년식 : " + year;
    }
}
```

Class 문제 5

- Car.JAVA

```
public class Car {
    private String owner;
    CarModel model;
    private int mileage;

    public Car(String owner, CarModel model, int mileage) {
        this.owner = owner;
        this.model = model;
        this.mileage = mileage;
    }

     @Override
   public String toString() {
       return  "소유자 : " + owner +"₩n" +
               model.toString() + "₩n" +
                "운행거리 : " + String.format("%,d Km", mileage);
   }
}
```

Class 문제 5

- Main.JAVA

```
public class Main {

    public static void main(String[] args) {
        Car myCar = new Car("홍길동",
                            new CarModel("Grandure", "현대자동차", 2001), 23000);

        System.out.println(myCar);
    }
}
```

Class 문제 6

- 나의 니콘 사진기 판매점을 프로그램해보자
  - 우리 shop에서 취급하는 Camera는 “Nikon” 제품으로 400,000원이고, 재고수(numberOfStock)는 30개, 팔린 개수(sold)를 50개이다.
  - 이것을 모델링하여 JAVA의 클래스로 표현해보자

Class 문제 5

- Camera 하나를 표현하는 클래스 Camera를 작성
  - Camera 클래스는 4개의 필드를 갖음
    - String 타입의 name(상품 이름)
    - int 타입의 price(가격)
    - int 타입의 numberOfStock(재고 수)
    - int 타입의 sold(팔린 개수)
  - Shop 클래스
    - Camera 클래스로 만든 Object를 member로 함
    - 수입금 (int income)
    - Camera를 판매한다

Class 문제 5

- Camera 클래스

```
public class Camera {
    private String name;
    private int price;
    private int numberOfStock;
    private int sold;

    public Camera(String name, int price, int numberOfStock, int sold) {
        this.name = name;
        this.price = price;
        this.numberOfStock = numberOfStock;
        this.sold = sold;
    }

    public void setNumberOfStock(int numberOfStock) {
        this.numberOfStock = numberOfStock;
    }
```

Class 문제 5

- Camera 클래스

```
public void setSold(int sold) {
    this.sold = sold;
}

public int getPrice() {
    return price;
}

public int getNumberOfStock() {
    return numberOfStock;
}

public int getSold() {
    return sold;
}
```

Class 문제 5

- Camera 클래스

```
    @Override
    public String toString() {
        return  "이름 : " + name + '₩n' +
                String.format("가격 : %,d원₩n", price) +
                String.format("재고 수량 : %,d 개₩n", numberOfStock) +
                String.format("판매 수량 : %,d 개", sold);
    }
}
```

Class 문제 5

- Shop 클래스

```
public class Shop {
    private Camera camera;
    private int income;

    public Shop(Camera camera) {
        this.camera = camera;
        income = 0;
    }

    public void sale() {
        if (camera.getNumberOfStock() == 0)
            System.err.println("재고가 없습니다");
        else {
            camera.setNumberOfStock(camera.getNumberOfStock() - 1);
            camera.setSold(camera.getSold() + 1);
            income += camera.getPrice();
        }
    }
```

Class 문제 5

- Shop 클래스

```
    @Override
    public String toString() {
        return camera.toString() +"₩n" +
                String.format("수입금 : %,d원₩n", income);
    }
}
```

Class 문제 5

- Main.JAVA

```
public class Main {
    public static void main(String[] args) {
        Camera nikon = new Camera("Nikon", 350000, 10, 0);
        Shop shop = new Shop(nikon);

        shop.sale();
        shop.sale();

        System.out.println(shop);
    }
}
```

Class 문제 6

- 학생이 대중 교통(버스, 지하철)을 이용하여 학교를 오는 것을 프로그램 해보자
  - 단 버스 요금은 1,200원, 지하철은 1,500원

✓ 김철수는 100번 버스를 타고 등교하였습니다.
✓ 이영희는 4호선 지하철을 타고 와서, 100번 버스로 환승하여 등교하였습니다.

Class 문제 6

- 필요한 클래스

Student
-name : String -money : int
+takeOnBus() : void +takeOnSubway() : void +takeOnBus() : void +takeOnSubway() : void +toSting() : String

Bus
-busNo : String -passengerCount : int -money : int -passenger : ArrayList
+takeOn() : void +takeOff() : void +toString() : String

Subway
-lineNo : String -passengerCount : int -money : int -passenger : ArrayList
+takeOn() : void +takeOff() : void +toString() : String

Class 문제 6

- Bus 클래스

```
public class Bus {
    String busNo;          // 버스번호
    int passengerCount;    // 승객 수
    int money;             // 돈
    ArrayList<String> passenger;

    public Bus(String busNo) {
        passenger = new ArrayList<>();
        this.busNo = busNo;
        money = 0;
        passengerCount = 0;
    }

    public void takeOn(String name) { // 승객을 태움
        passenger.add(name);
        ++passengerCount;
        this.money += 1200;
    }
```

Class 문제 6

- Bus 클래스

```
    public boolean takeOff(String name) {
        if (passenger.contains(name))
            return true;
        else
            return false;
    }

    @Override
    public String toString() {
        return  String.format("busNo : %s, 탑승객 수 : %d 명, 수입금 : %,d 원,
                                                                승객명 = %s",
                busNo, passengerCount, money, passenger);
    }
}
```

Class 문제 6

- Subway 클래스

```
public class Subway {
    private String lineNo;
    private int passengerCount;
    private int money; // 돈
    private ArrayList<String> passenger;

    public Subway(String lineNo) {
        this.lineNo = lineNo;
        passengerCount = 0;
        money = 0;
        passenger = new ArrayList<>();
    }

    public void takeOn(String name) { // 승객을 태움
        passenger.add(name);
        passengerCount++;
        this.money += 1500;
    }
```

Class 문제 6

- Subway 클래스

```
    public boolean takeOff(String  name) {
        if (passenger.contains(name))
            return true;
        else
            return false;
    }

    @Override
    public String toString() {
        return String.format("%s호선 지하철, 승객 수 : %d 명, 수입금 : %,d 원,
                        승객명 : %s", lineNo, passengerCount, money, passenger);
    }
}
```

Class 문제 6

- Student 클래스

```
public class Student {
    private String name;
    private int money;

    public Student(String name, int money) {
        this.name = name;
        this.money = money;
    }

    public void takeOnBus(Bus bus) {
        bus.takeOn(name);
        money -= 1200;
    }
    public void takeOffBus(Bus bus) {
        bus.takeOff(name);
    }
```

Class 문제 6

- Student 클래스

```
    public void takeOnSubway(Subway subway) {
        subway.takeOn(name);
        money -= 1500;
    }

    public void takeOffSubway(Subway subway) {
        subway.takeOff(name);
    }

    @Override
    public String toString() {
        return String.format("%s님의 잔액은 %,d원 입니다.₩n", name, money);
    }
}
```

Class 문제 6

- Main.JAVA

```
public class Main {
    public static void main(String[] args) {
        Student kim = new Student("김철수", 15000);
        Student lee = new Student("이영희", 10000);
        Bus bus100 = new Bus("100");
        Subway subwayGreen = new Subway("4"); // 2호선
        kim.takeOnBus(bus100);
        kim.takeOffBus(bus100);
        lee.takeOnSubway(subwayGreen);
        lee.takeOffSubway(subwayGreen);
        lee.takeOnBus(bus100);
        lee.takeOffBus(bus100);

        System.out.print(kim);
        System.out.print(lee);
        System.out.println(bus100);
        System.out.println(subwayGreen);
    }
}
```

Class 문제 7

- 학생이 대중교통(버스, 지하철, Taxi)을 이용하여 학교를 오는 것을 프로그램 해보자
  - 단 버스 요금은 1,200원, 지하철은 1,500원
  - Taxi 요금은 meter기 요금

✓ 김철수는 100번 버스를 타고 등교하였습니다.
✓ 이영희는 4호선 지하철을 타고 와서, 100번 버스로 환승하여 등교하였습니다.
✓ 박정민이는 늦잠을 자서 4호선 지하철을 타고 와서, 경복 Taxi를 타고 등교하였습니다.

Class 문제 8

- 다음의 Box 객체를 정의할 Box 클래스를 작성해보자.

- Attribute(속성)와 Behavior(행동)을 구분해보자.
  - Attribute
    - Box의 크기 (가로, 세로, 높이)
    - Box의 색상
    - Box의 문구
  - Behavior
    - 부피를 구하다

Class 문제 8

- Class 생성
  - Class 이름은 File 이름과 같음
  - Class 이름은 대문자로 시작함
  - 속성 정의
    - 속성의 이름은 소문자로 시작함

- Data의 속성만 선언된 클래스 예

```
class Box {
    private int width;      // attribute(속성)을 Field로 구현
    private int height;
    private int depth;
}
```

Class 문제 8

- 생성자
  - Object를 생성하는 용도로 사용함
  - 생성자의 이름은 Class 이름과 같음
  - Class를 만들면 자동으로 Default 생성자가 만들어짐
  - 생성자를 정의하면 Default 생성자는 없어짐

```
class Box {
    private int width;
    private int height;
    private int depth;

    public Box() {    //default 생성자
    }
    public Box(int width, int height, int depth) {    //생성자
        this.width = width;
        this.height = height;
        this.depth = depth;
    }
}
```

Class 문제 8

- Setter 정의(변경 Method)
  - 속성값 설정
  - 일반적으로 메소드 이름이 'set~'로 시작함

```
class Box {
    private int width;
    private int height;
    private int depth;

    public void setWidth(int width) {
        this.width = width;
    }

    public void setHeight(int height) {
        this.height = height;
    }
}
```

Class 문제 8

- Getter 정의 (접근 Method)
  - 속성값 얻어 오기
  - 일반적으로 메소드 이름이 'get~'로 시작함

```
class Box {
    private int width;
    private int height;
    private int depth;

    public int getWidth(){
        return this.width;
    }

    public int getHeight(){
        return this.height;
    }
}
```

Class 문제 8

- Custom Method 정의
  - 메소드 이름은 소문자로 시작
  - 메소드의 반환 Data 형을 고려할 것
  - Data의 속성과 메소드를 가진 클래스 선언 예

```
class Box {
    private int width;
    private int height;
    private int depth;

    public int volume() {
        int volume;
        volume = width * height * depth;

        return volume;
    }
}
```

Class 문제 8

- toString( ) 메소드
  - Object를 String 타입으로 변환시켜주는 메소드
  - 상속 시 상위 클래스의 메소드 교체

```
@Override
public String toString() {
    return "Box{" +
            "width=" + width +
            ", height=" + height +
            ", depth=" + depth +
           '}';
}
```

Class 문제 8

- Object는 new 키워드를 이용하여 생성
  - new는 객체의 생성자 호출

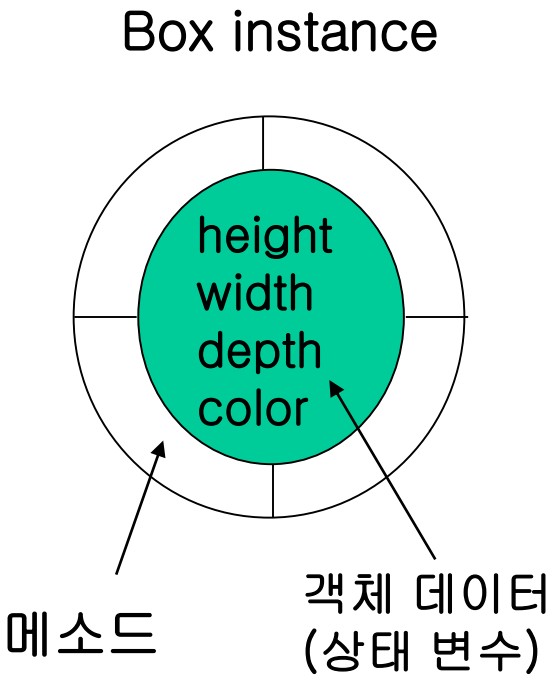
- Object 생성 과정
  1. 객체에 대한 레퍼런스 변수 선언
  2. 객체 생성

```
public static void main (String args[]) {
    Box brownBox;                    // 레퍼런스 변수 aPerson 선언
    brownBox  = new Box();           // Person 객체 생성

}
```

```
public static void main (String args[]) {
    Box brownBox = new Box( );

}
```

Class 문제 9

- 다음의 Box 클래스를 정의해보자



- ✓ 속성
  - ✓ 높이, 깊이, 넓이, 색상
- ✓ 메소드
  - ✓ 상자의 크기를 설정하는 메소드
  - ✓ 상자의 색상을 설정하는 메소드
  - ✓ 상자의 각 속성을 반환하는 메소드
  - ✓ 상자의 정보를 반환하는 메소드

Class 문제 9

- 속성 정의
  - Class가 가지는 Data(Field)

```
class Box2 {
    private int width;
    private int height;
    private int depth;
    private String color;
}
```

클래스 이름은
대문자로 작성

Class 문제 9

- 생성자

```
class Box2 {
      ..................................
     public Box2() {
     }
    public Box2(int width, int height, int depth) {
        this.width = width;
        this.height = height;
        this.depth = depth;
    }
    public Box2(int width, int height, int depth, String color) {
         this.width = width;
        this.height = height;
        this.depth = depth;
        this.color = color;
    }
}
```

Class 문제 9

- Setter 정의

```
class Box2 {
    ..................................

    public void setWidth(int width) {
        this.width = width;
    }

    public void setDepth(int depth) {
        this.depth = depth;
    }

    public void setColor(String color) {
        this.color = color;
    }
}
```

Class 문제 9

- Getter 정의

```
class Box2 {
    ……….
    public int getWidth(){
        return this.width;
    }
    public int getHeight(){
        return this.height;
    }
    public int getDepth(){
        return this.depth;
    }
    public String getColor(){
        return this.color;
    }
}
```

Class 문제 9

- Custom 메소드 정의

```
public void setSize(int width, int height, int depth) {
    this.width = width;
    this.height = height;
    this.depth = depth;
}

public int volume() {
   int test = width * height * depth;
    return test;
}
```

Class 문제 9

- toString( ) 메소드
  - Object를 String 타입으로 변환시켜주는 메소드
  - 상속 시 상위 클래스의 메소드 교체

```
@Override
public String toString() {
    return "Box{" +
            "width=" + width +
            ", height=" + height +
            ", depth=" + depth +
            ", color='" + color + '₩'' +
            '}';
}
```

Class 문제 9

- 객체의 생성 및 사용

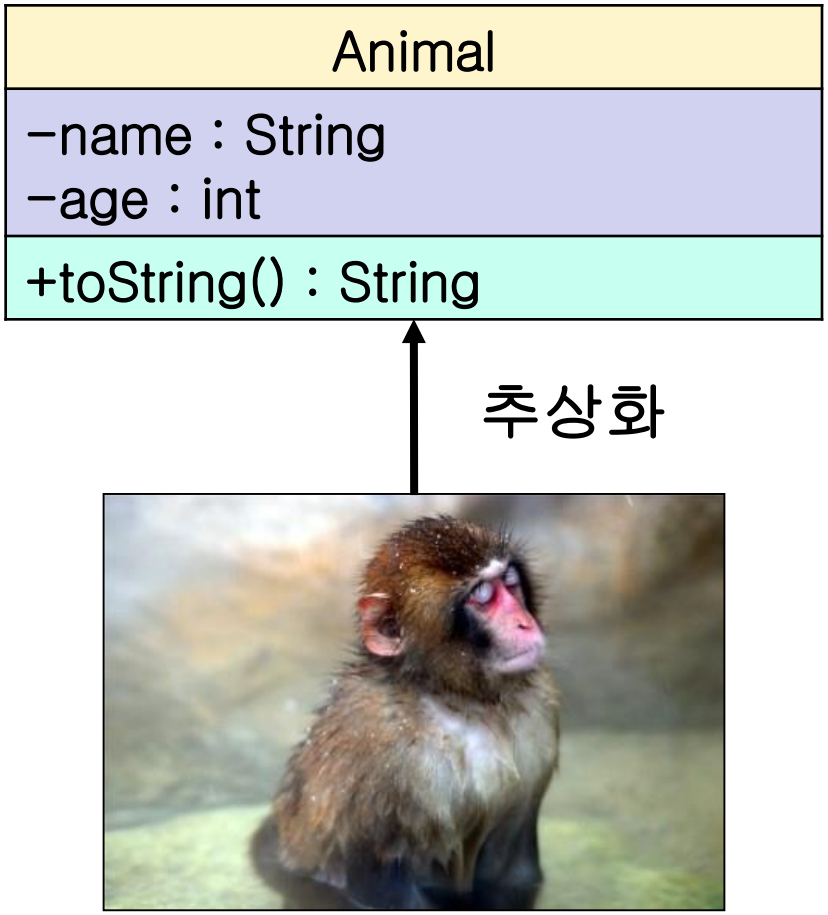
```
Box2 brownBox = new Box2();
Box2 whiteBox;

brownBox.setWidth(20);
brownBox.setHeight(40);
brownBox.setDepth(15);
brownBox.setColor("Brown");

whiteBox = new Box2(10, 20, 30, "White");

System.out.printf(" 첫번째 박스 " + brownBox);
System.out.printf(" 첫번째 박스의 부피는 %,d 입니다.₩n", brownBox.volume());
System.out.printf(" 첫번째 박스 " + whiteBox);
System.out.printf(" 두번째 박스의 부피는 %,d 입니다.₩n", whiteBox.volume());
```

Class 문제 10

- 동물(Animal)의 이름(name)과 나이(age)의 속성을 갖는 동물 Class를 만들고, 원숭이(Monkey) Object를 만들어 출력하여 보아라

Class 문제 10

- Class 설계



Class 문제 10

- Class 작성

```
//이름과 나이를 속성으로 갖는 Animal 클래스에 설계
class Animal {
	private String name;
	public int age;

	public Animal(String name, int age) {  // 생성자
		this.name = name;
		this.age = age;
	}

	public String toString() {                    // 메소드
		return String.format("동물 이름 : %s, 나이 = %d 살₩n", name,
                                                                age);
	}
}
```

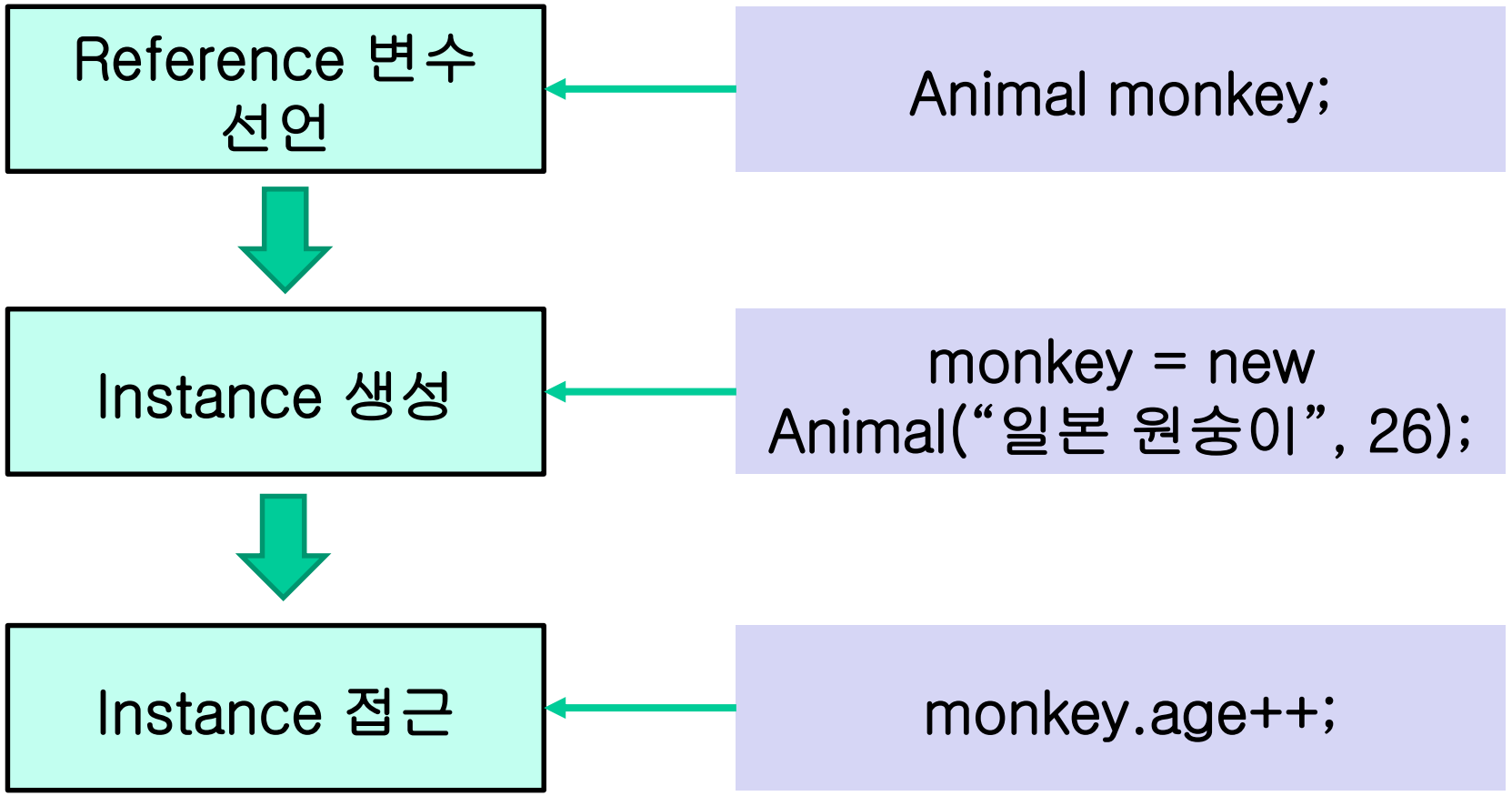
Class 문제 10

- 원숭이 Object 생성

```
public class AmimalTest{
      public static void main(String[] args) {
         Animal monkey;                                  //Reference 변수 선언

         monkey = new Animal("일본 원숭이", 26);   //객체 생성
         monkey.age++;
         System.out.println(monkey);
      }
}
```

Class 문제 10

- Object 생성



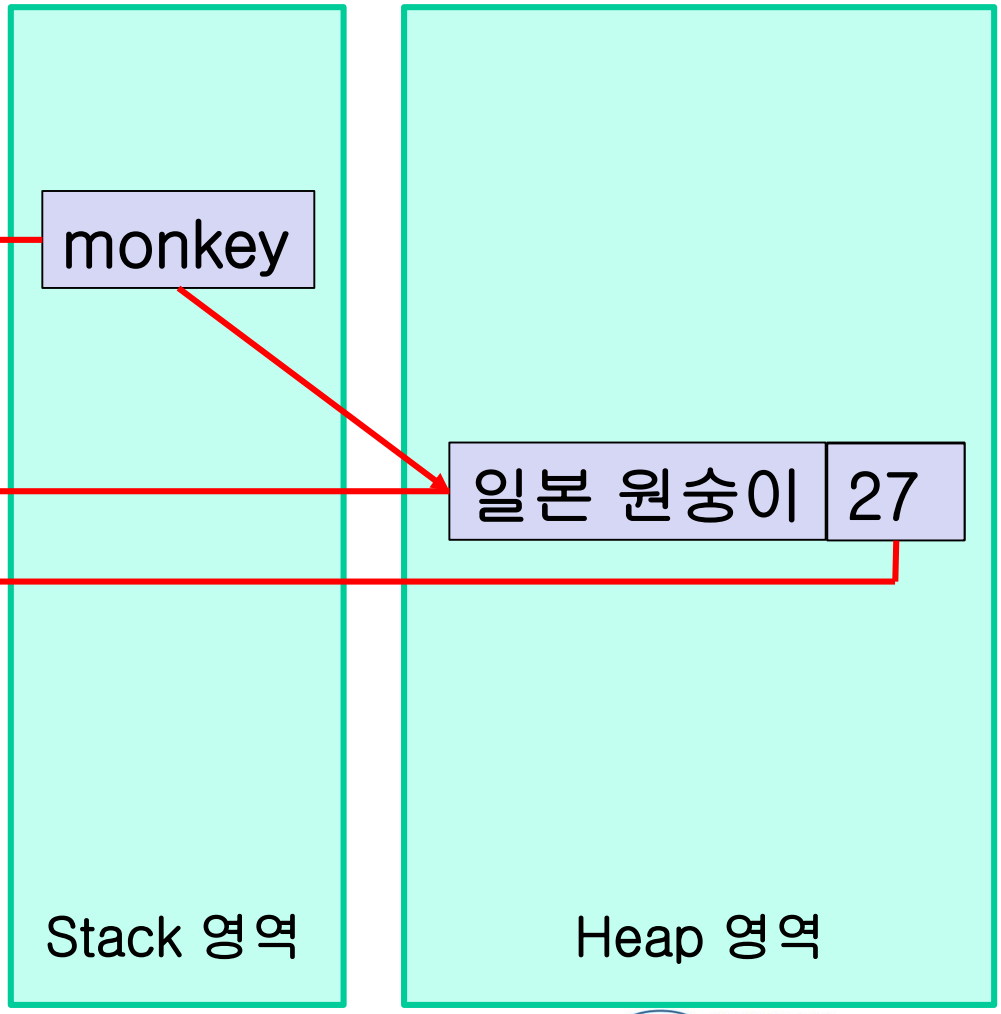
Class 문제 10

- 원숭이 Object 생성

Animal monkey;

monkey = new
Animal("일본 원숭이", 26);

monkey.age++;



Class 문제 10[심화]

- Animal 객체를 하나 더 생성하여 Reference 변수 penguin로 접근하도록 하고 이름은 “황제 펭귄" 나이는 2를 저장한 후 출력하시오.

Class 문제 10[심화]

- 펭귄 Object 생성

```
public class AmimalTest {
      public static void main(String[] args) {
          ……….
         Animal penguin = new Animal(“황제 펭귄”, 2);   //객체 생성
         System.out.println(penguin);              //객체의 멤버에 저장된 값 출력
      }
}
```

Class 문제 10[심화]

```
class Animal {
    private String name;
    private int age;

    public Animal() {
    }

     public Animal(String name, int age) {
        this.name = name;
        this.age = age;
    }

     public void setAge(int age) {              // setter()와 getter()
        this.age = age;
    }

    public int getAge(){
         return age;
    }

    public toString( ) {
        return String.format("동물 이름 : %s, 나이 = %d 살₩n", name, age);
    }
}
```

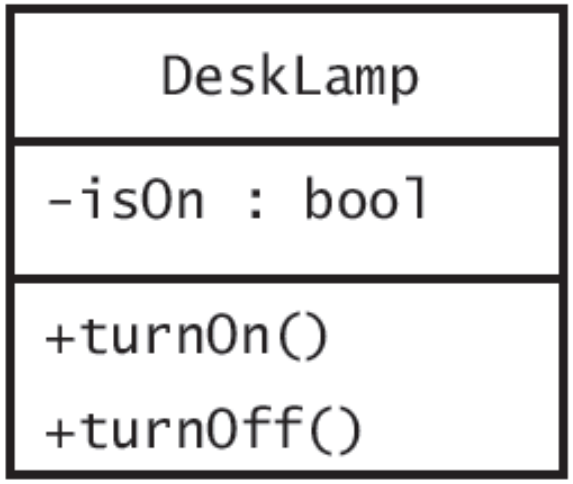
Class 문제 10[심화]

- 원숭이 Object 생성

```
public class AmimalTest{
      public static void main(String[] args) {
         Animal monkey;                                  //Reference 변수 선언

         monkey = new Animal("일본 원숭이", 26);   //객체 생성
         monkey.setAge(monkey.getAge() + 1);
         System.out.println(monkey);
      }
}
```

Class 문제 11

- 집에서 사용하는 Desk Lamp를 클래스로 작성하여 보자



Class 문제 11

- 클래스 작성

```
class DeskLamp {
    private boolean isOn;



    public void turnOn( ) {
        isOn = true;
    }

    public void turnOff( ) {
        isOn = false;
    }

    public String toString( ) {
        return “현재 상태는 “ + (isOn ? “켜짐” : “꺼짐”);
    }
}
```

Class 문제 11

- 객체 생성

```
public  class DeskLamp {
    public static void main(String args[ ]) {
            DeskLamp myLamp = new DeskLamp( );

            myLamp.turnOn();
            System.out.println(myLamp);
            myLamp.turnOff();
            System.out.println(myLamp);
    }
}
```

Class 문제 12

- 날자 클래스를 작성해보자

Date
-year : int -month : string -day : int
+setDate() +printDate()

Class 문제 12

- Class 생성

```
class Date {
    private int year;                          // 년도
    private String month;                      // 월 이름
    private int day;                           // 일

    public Date(int year, String month, int day) {    // 날자 초기화
        this.year = year;
        this.month = month;
        this.day = day;
    }
```

Class 문제 12

- Class 생성

```
public int getYear() {
    return year;
}

public void setYear(int year) {
    this.year = year;
}

public String getMonth() {
    return month;
}

public void setMonth(String month) {
    this.month = month;
}
```

Class 문제 12

- Class 생성

```
    public int getDay() {
        return day;
    }
    public void setDay(int day) {
        this.day = day;
    }
    public void setDate(int year, String month, int day) {  // 날자 지정
        this.year = year;
        this.month = month;
        this.day = day;
    }
    public void printDate() {
        System.out.println(year + "년 " + month + " " + day + "일");
    }
}
```

Class 문제 12

- 객체 생성및 사용

```
public  class DateTest {
    public static void main(String args[ ]) {
            Date date = new Date(2017, “5월”, 15);

            date.printDate( );
            date.setDate(2022, “3월”, 20);
            date.printDate( );
            date.setYear(2018);
            date.printDate( );
    }
}
```

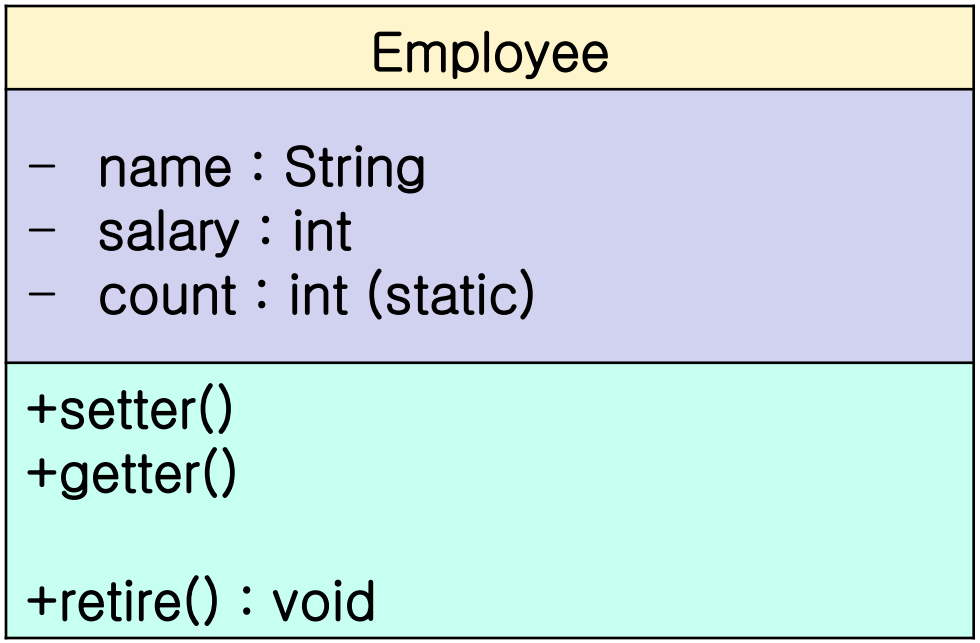
Class 문제 12[심화]

- 시간을 나타내는 Time 클래스

Time
- hour : int - minute : int - second : int
+Time() +Time(hour, minute, second) +setter() +getter() +toString()

직원 클래스 작성하기

- 직원(Employee)을 나타내는 클래스에서 직원들의 수를 카운트하는 예를 살펴보자. 직원의 수를 정적 변수로 나타낸다



직원 클래스 작성하기

- Employee.JAVA

```
public class Employee {
    private String name;
    private int salary;
    static int count = 0; // 정적 변수

    public Employee(String name, int salary) {    // 생성자
        this.name = name;
        this.salary = salary;
        count++; // 정적 변수인 count를 증가
    }

    public String getName() {
        return name;
    }

    public int getSalary() {
        return salary;
    }
```

직원 클래스 작성하기

- Employee.JAVA

```
    public static int getCount() {
        return count;
    }

    // 객체가 소멸될 때 호출된다.
    protected void retire() {
        count--; // 직원이 하나 줄어드는 것이므로 count를 하나 감소
    }
}
```

직원 클래스 작성하기

- Employee.JAVA

```
public class Main {
    public static void main(String[] args) {
        Employee employee1 = new Employee("김철수", 35000);
        Employee employee2 = new Employee("최수철", 50000);
        Employee employee3 = new Employee("김철호", 20000);

        int count = Employee.getCount();
        System.out.println("현재의 직원수 = " + count);
    }
}
```

과일 장수

- Object를 이루는 것은 Data와 기능(동작)
  - 예) 과일 장수 Objet의 표현

과일 장수는 과일을 팝니다. 행위
과일 장수는 사과 20개, 오렌지 10개를 보유하고 있습니다. 상태
사과는 개당 1000원이고, 오렌지는 개당 500원이다.
과일 장수의 과일 판매수익은 50,000원입니다. 상태

  - 과일 장수의 Data 표현

✓ 보유하고 있는 사과의 수 : int numOfApple;
✓ 보유하고 있는 오렌지의 수 : int numOfOrange;
✓ 판매수익 : int myMoney;

과일 장수

- Object
  - 사전적 의미 : 물건 또는 대상
- Object Oriented Programming

객체지향 프로그래밍에서는 나, 과일장수, 사과라는 객체를 등장시켜서 두 개의 사과 구매라는 행위를 실체화 함

과일 장수

- Object를 이루는 것은 Data와 기능(동작)
  - 과일 장수의 판매 행위 표현 (Method)

```
int saleApple(int money) {      // 사과 구매액이 메소드의 인자로 전달
      int num = money / 1000; // 사과가 개당 1000원이라고 가정
      numofApple -= num;        // 사과의 수가 줄어든다
      myMoney += money;         // 판매 수익 발생
      return num;                    // 실제 구매가 발생한 사과의 수 반환
}
```

과일 장수

- '과일장수' 클래스 정의와 키워드 final

```
public class FruitSeller {
    final static int APPLE_PRICE = 1000;
    final static int ORANG_PRICE = 500;
    private int numOfApple;
    private int numOfOrange;
    private int myMoney;

    public FruitSeller(int numOfApple, int numOfOrange, int myMoney) {
        this.numOfApple = numOfApple;
        this.numOfOrange = numOfOrange;
        this.myMoney = myMoney;
    }

    public int saleApple(int count) {
        int num = count * APPLE_PRICE;
        numOfApple -= count;
        myMoney += num;
        return count;
    }
```

과일 장수

- '과일장수' 클래스 정의와 키워드 final

```
    public int saleOrange(int count) {
        int num = count * ORANG_PRICE;
        numOfOrange -= count;
        myMoney += num;
        return count;
    }

    public void showSaleResult( ) {
        System.out.println("남은 사과 : " + numOfApple);
        System.out.println("남은 오렌지 : " + numOfOrange);
        System.out.println("판매 수익 : " + myMoney);
    }
}
```

과일 장수

- 과일 구매자 관점에서의 나(me) 클래스 표현!

```
public class FruitBuyer {
    private int myMoney;
    private int numOfApple;

    public FruitBuyer(int myMoney) {
        this.myMoney = myMoney;
        numOfApple = 0;
    }

    public void buyApple(FruitSeller seller, int count) {
        numOfApple += seller.saleApple(count);
        myMoney -= count * FruitSeller.APPLE_PRICE;
    }

    public void showBuyResult( ) {
        System.out.println("현재 잔액 : " + myMoney);
        System.out.println("사과 개수 : " + numOfApple);
    }
}
```

과일 장수

- 사과장수 시뮬레이션

```
public static void main(String[] args) {
    FruitSeller seller = new FruitSeller(20, 10, 50000);
    FruitBuyer buyer = new FruitBuyer(5000);

    buyer.buyApple(seller, 2);

    seller.showSaleResult();
    buyer.showBuyResult();
}
```

Message 전달은 두 객체 간의 대화 방법이다.
위 예제에서의buyApple() 메소드가 의미하는 바는
"아저씨 사과 2개 주세요!"

Counter

- 계수기를 만들어보자
  - 키는 up, reset

Counter

- Counter.JAVA

```
public class Counter {
   private int count;

   public Counter() {
      count = 0;
   }

    int getCount() {
      return count;
   }

   void reset() {
      count = 0;
   }

   void up() {
      count++;
   }
}
```

Counter

- Main.JAVA

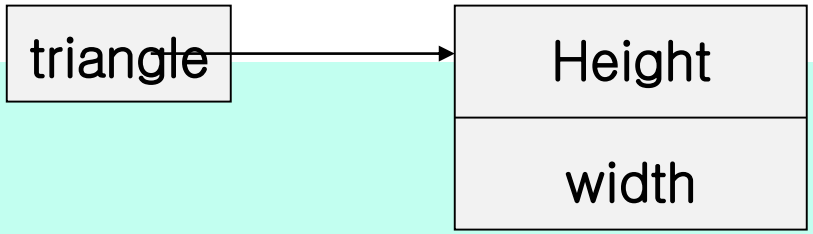
```
public static void main (String[] args) {
	Counter counter = new Counter();

	counter.up();
	counter.up();
	counter.up();
	System.out.println("Counter 값은 : " + counter.GetCount());
	counter.reset();
	System.out.println("Counter 값은 : " + counter.GetCount());
}
```

삼각형의 면적

- Data와 Information을 구분해보자
- 삼각형 클래스 정의
  - 속성
    - 밑변(width) : double
    - 높이(height) : double
  - 메소드
    - 면적을 계산하다(area()) : double
    - Data와 Information을 출력하다(toString()) : void
- 객체 생성 및 활용

삼각형의 면적

- Triangle.JAVA




```
public class Triangle {
    private double width;
    private double height;

    public Triangle(double width, double height) {
        this.width = width;
        this.height = height;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    public void setHeight(double height) {
        this.height = height;
    }
```

삼각형의 면적

- Triangle.JAVA

```
    public double area() {
        double area = (height * width) / 2.0;

        return area;
    }

    @Override
    public String toString() {
        return String.format("₩n 삼각형의 면적 "+
                "₩n₩t 밑변 : %.2f Cm₩n₩t 높이 : %.2f Cm₩n" +
                "₩t 면적 : %.2f Cm2₩n", width, height, area());
    }
}
```

삼각형의 면적

- Main.JAVA

```
public static void main(String[] args) {
    Triangle triangle = new Triangle(3.5, 5);

    triangle.setHeight(6.0);
    triangle.setWidth(5.6);
    System.out.println(triangle);
}
```

```
삼각형의 면적
    밑변 : 5.60 Cm
    높이 : 6.00 Cm
    면적 : 16.80 Cm2
```

두수 더하기

- 덧셈(Add)을 하는 클래스를 만들어 보자
  (단, 클래스 내에 변수 2개를 선언하여 내부에 저장 후 값을 출력하는 방식으로 하자)

```
class Add {

}
```

두수 더하기

- 덧셈(Add)을 하는 클래스를 만들어 보자
  (단, 클래스 내에 변수 2개를 선언하여 내부에 저장 후 값을 출력하는 방식으로 하자)

```
class Add {
        private int num1;
        private int num2;
}
```

두수 더하기

- 생성자를 이용하는 방법
  - 클래스 이름과 “동일한” 이름을 가진 메소드
  - 반환형이 없는 메소드

```
class Add {
      private int num1
      private int num2;

      public Add(int num1, int num2) {
           this.num1 = num1;
           this.num2 = num2;
     }
}
```

두수 더하기

- Getter() 만들기

```
class Add {
        private int num1
        private int num2;

        public int getNum1() {
            return num1;
        }

        public int getNum2() {
            return num2;
       }
}
```

두수 더하기

- Custom 메소드

```
class Add {
        int num1, num2;
        ……….

        public void input(int num1, int num2) {
             this.num1 = num1;
             this.num2 = num2;
        }

         public int sum(int num1, int num2) {
                return num1 + num2;
        }
}
```

두수 더하기

- Object 생성 사용하기

```
public static void main(String[] args) {
    Scanner keyboard = new Scanner(System.in);

    System.out.print("정수 2개를 입력 : ");
    int num1 = keyboard.nextInt();
    int num2 = keyboard.nextInt();

    Add add = new Add(num1, num2);
    add.input(num1, num2);

    System.out.printf(" %d + %d = %d₩n" + num1, num2,
                                                num1 + num2);
    System.out.printf(" %d + %d = %d₩n" + add.getNum1(),
                                add.getNum2(), add.sum(num1, num2));
}
```

두수 더하기[심화]

```
public class Main {
    public int add(int a, int b){
        return a + b;
    }

    public static void main(String[] args) {
        System.out.println(add(1, 2));    // error!
    }
}
```

Cannot make a static reference to the non-static method add(int, int) from the type Main

두수 더하기[심화]

```
public class Main {
    public int add(int a, int b) {
        return a + b;
    }

    public static void main(String[] args) {
        Main adder = new Main();

        System.out.println(adder.add(1, 2));     // OK!
    }
}
```

두수 더하기[심화]

```
public class Main {
    public static int add(int a, int b){
        return a + b;
    }

    public static void main(String[] args) {
        System.out.println(Main.add(1, 2));      // OK!
    }
}
```

Class 문제 13[심화]

```
public class Main {
    public static int add(int a, int b){
        return a + b;
    }

    public static void main(String[] args) {
        System.out.println(Main.add(1, 2));      // OK!
        System.out.println(add(1, 2));           // OK!
                                                 // 클래스 내부에서는
                                                 // 클래스 이름 생략 가능!
    }
}
```

자연수의 합

- N에서 M까지의 자연수의 합을 구하여 보자
  - 단, n,과 m은 0보다 큼

  - 구조화 Programming
  - 객체 지향 Programming

Structured Programming

```
public class Main {
    public static void main(String[] args) throws IOException {
        Scanner keyboard = new Scanner(System.in);
        int start, last, result = 0;
        while(true) {
            System.out.print("어디서부터 더할까요? ");
            start = keyboard.nextInt();
            if (start > 0)
                break;
            else {
                System.err.print("입력 오류");
                System.in.read();
            }
        }
        do {
            System.out.print("어디까지 더할까요? ");
            last = keyboard.nextInt();
        } while (last <= 0);
```

Structured Programming

```
        for (int i = start; i <= last; i++)
            result += i;
        System.out.printf("%d + … + %d = %d₩n", start, last, result);
    }
}
```

Structured Programming

- 같은 Class 안에 있는 static 메소드를 호출할 때

```
public class Main {
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        int start, last, result = 0;
        do {
            System.out.print("어디서부터 더할까요? ");
            start = keyboard.nextInt();
        } while (start <= 0);
        do {
            System.out.print("어디까지 더할까요? ");
            last = keyboard.nextInt();
        } while (last <= 0);
        result = add(start, last);

        System.out.printf("%d + … + %d = %d₩n", start, last, result);
    }
```

Structured Programming

- 같은 클래스 안에 있는 static 메소드를 호출할 때

```
    private static int add(int start, int last) {
        int result = 0;
        for (int i = start; i <= last; i++)
            result += i;
        return result;
    }
}
```

Structured Programming

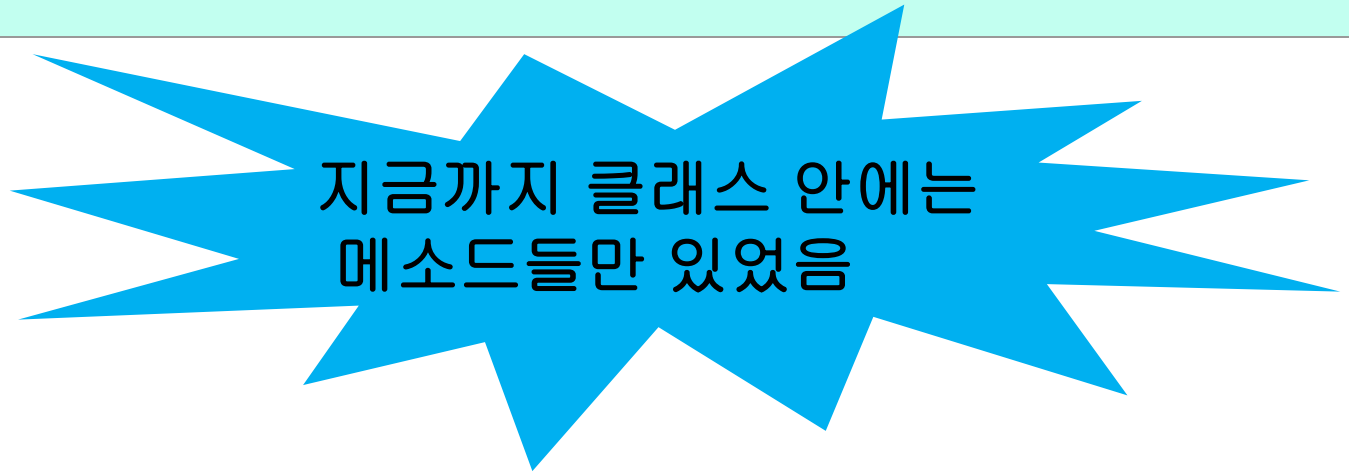
- 다른 클래스에 있는 static 메소드를 호출할 때

```
public class Main {
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        int start, last, result = 0;
        do {
            System.out.print("어디서부터 더할까요? ");
            start = keyboard.nextInt();
        } while (start <= 0);
        do {
            System.out.print("어디까지 더할까요? ");
            last = keyboard.nextInt();
        } while (last <= 0);
        result = Adder.add(start, last);

        System.out.printf("%d + … + %d = %d₩n", start, last, result);
    }
}
```

Structured Programming

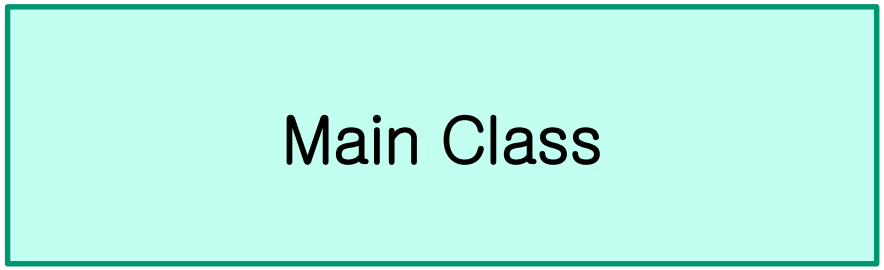
- 다른 클래스에 있는 static 메소드를 호출할 때

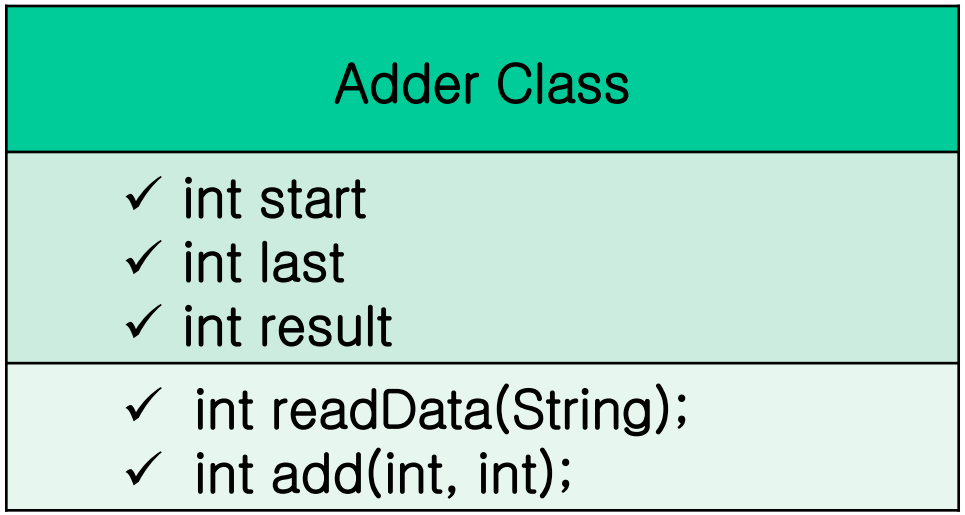
```
public class Adder {
    public static int add(int start, int last) {
        int result = 0;
        for (int i = start; i <= last; i++)
        result += i;
        return result;
    }
}
```



Object Oriented Programming

- 추상화





Object Oriented Programming

- Adder Class

```
public class Adder {
    private int start;          // 필드
    private int last;
    private int result;

    public Adder() {      // 생성자
        result = 0;
    }

    public Adder(int start, int last) {    // 생성자
        this.start = start;
        this.last = last;
        result = 0;
    }
```

Object Oriented Programming

- Adder Class

```
public void setStart(int start) {      //setter
    this.start = start;
}

public void setLast(int last) {
    this.last = last;
}

public int getStart() {                //getter
    return start;
}

public int getLast() {
    return last;
}
```

Object Oriented Programming

- Adder Class

```
public int readData(String prompt) throws IOException {
    Scanner keyboard = new Scanner(System.in);
    int data = 0;
    while(true) {
        System.out.print(prompt);
        data = keyboard.nextInt();
        if (data > 0)
            break;
        else {
            System.err.print("입력 오류");
            System.in.read();
        }
    }
    return data;
}
```

Object Oriented Programming

- Adder Class

```
    public void add() {
        for (int i = start; i <= last; i++)
            result += i;
    }

    @Override
    public String toString() {
        return start + " + ..... + " + last + " = " + result;
    }
}
```

Object Oriented Programming

- Main Class

```
public static void main(String[] args) throws IOException {
        Adder test = new Adder();

        test.setStart(test.readData("어디서부터 더할까요? "));
        test.setLast(test.readData("어디까지 더할까요? "));

        test.add();

        System.out.print(test);
}
```