- 별도 명시가 없으면 필드는 private 으로 설정, 메소드는 public으로 설정
- 필요한 메소드는 추가 작성 가능
- 요구된 기능 외에 추가적인 기능들 구현 가능
- Test 클래스에서 사용하지 않더라도 생성자와 접근자,설정자 메소드, toString 들은 작성해 두도록 함.

Lab 3. Chapter5

실습 1> BankAccount 클래스 작성 (BankAccount.java, BankAccountTest.java)

- 은행 계좌를 나타내는 BankAccount 클래스는 잔액을 나타내는 balance 필드를 가지고 있다. 생성자, 입금(deposit), 출금(withdraw), toString 메소드를 추가하라.
  - 입금/출금할 때 매개변수로 들어온 금액이 양수일 때만 처리되도록 한다.
  - 출금할 때 매개변수로 들어온 금액보다 잔액이 크거나 같을 때만 출금 처리되도록 한다.
  - 테스트 프로그램에 출금/입금 기능을 사용하지 않더라도 입금/출금 메소드를 작성하도록 한다.
- 이 클래스에 계좌간 이체 기능을 수행하는 메소드 transfer(int amount, BankAccount otherAccount)를 추가하라.
  - transfer()는 현재 객체의 잔액에서 amount 만큼을 otherAccount 계좌로 송금한다.
  - 이 때 잔액이 부족하면 이체를 하지 않고 잔액 부족 메시지를 출력한다.

실습 1> BankAccount 클래스 작성

```
<terminated> BankAccountTest (1) [Java Application
계좌 1 잔액 입력:10000
계좌 2 잔액 입력:2000
myAccount1: BankAccount [balance=10000]
myAccount2: BankAccount [balance=2000]

계좌 1 --> 계좌 2 이체 금액 입력:3000
transfer 호출 후
myAccount1: BankAccount [balance=7000]
myAccount2: BankAccount [balance=5000]
```

```java
public class BankAccountTest {
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        System.out.print("계좌 1 잔액 입력:");
        int a1 = scan.nextInt();
        BankAccount myAccount1 = new BankAccount(a1);
        System.out.print("계좌 2 잔액 입력:");
        int a2 = scan.nextInt();
        BankAccount myAccount2 = new BankAccount(a2);
        System.out.println("myAccount1: " + myAccount1);
        System.out.println("myAccount2: " + myAccount2+"\n");
        System.out.print("계좌 1 --> 계좌 2 이체 금액 입력:");
        int t = scan.nextInt();
        myAccount1.transfer(t, myAccount2);
        System.out.println("transfer 호출 후");
        System.out.println("myAccount1: " + myAccount1);
        System.out.println("myAccount2: " + myAccount2);
    }
}
```

실습 2> Car 클래스 작성 (Car.java, CarTest.java)

- 자동차 회사에서 지금까지 생산한 자동차의 대수를 정적 변수를 이용하여 계산하려고 한다. Car 클래스에 모델 이름(model), 생산자(make) 등의 필드를 정의하고, 생성자, 설정자, 접근자도 적절하게 추가하라.
- 그리고 지금까지 생산된 자동차의 대수를 나타내는 정적 변수 numberOfCars를 추가한다. 이 정적 변수를 외부로 반환해주는 정적 메소드 getNumberOfCars()도 정의해서 사용해보자.

```
public class CarTest {
    public static void main( String[] args ) {
        new Car("3SERIES", "BENZ");
        new Car("3SERIES", "BENZ");
        new Car("3SERIES", "BENZ");
        System.out.println("총 "+Car.getNumberOfCars() + "대의 자동차가 생산되었습니다.");
    }
}
```

```
<terminated> CarTest [Java Appli
총 3대의 자동차가 생산되었습니다.
```

실습 3> Circle 클래스 작성 (Circle.java, CircleTest.java)

- 크기가 3인 Circle 객체 배열을 생성하고 여기에 Circle 객체를 3개 생성하여 배열에 저장한다.
- Circle 객체 생성할 때 매개변수로 전달되는 원의 반지름은 0부터 100 사이의 난수를 사용하여 생성한다.
- 배열에 저장된 객체들을 꺼내서 화면에 출력한다.

```
class Circle {
    int radius;

    public Circle(int d) {
        radius = d;
    }
    @Override
    public String toString() {
        return "Circle [radius=" + radius + "]";
    }
}
```

```
<terminated> CircleTest [Ja
Circle [radius=69]
Circle [radius=16]
Circle [radius=5]
```

- 위의 Circle 클래스를 대상으로 객체 배열을 생성하여 테스트하는 CircleTest 클래스를 작성해 보자.

실습 4> Contacts 클래스 작성 (Contacts.java, ContactsTest.java)

- 지인들의 연락처(최대 100)를 저장하고 검색하는 프로그램을 객체 배열을 이용하여 작성해 보자.
  - Contacts 클래스는 이름(name), 전화번호(tel), 이메일(email)등의 필드를 가진다.
    - 여기에 지인들의 수를 저장하는 정적 변수 count를 추가하라.
  - ContactsTest 클래스에서는 Contacts 객체를 저장하는 객체 배열을 생성한 후 사용자로부터 연락처 정보를 입력 받는다.
  - 실행 결과를 참고해서 ContactsTest 클래스를 작성하라.

```
<terminated> ContactsTest (1) [Java Application] C:₩Program Files₩Java₩jdk-1
연락처를 입력하시오(종료 -1)
이름과 전화번호, 이메일을 입력하시오:Kim 010-1111-2222 kim@java.com
이름과 전화번호, 이메일을 입력하시오:Lee 010-2222-3333 lee@java.com
이름과 전화번호, 이메일을 입력하시오:Kang 010-5555-7777 kang@java.com
이름과 전화번호, 이메일을 입력하시오:Choe 010-1234-5678 choe@java.com
이름과 전화번호, 이메일을 입력하시오:-1
지인들의 수는 4입니다
검색할 이름을 입력하시오:Kang
Kang의 전화번호: 010-5555-7777이메일: kang@java.com
```

문자열 a와 b가 같은지 비교할 때 a.equals(b)가 참인지로 판단한다.

실습 5> Book 클래스 작성 (Book.java, BookTest.java) << 교재 Mini Project

- 사용자가 읽은 책과 평점을 저장하는 객체 배열을 생성해보자. 다음과 같은 메뉴가 제공된다.

```
==============================
1. 책 등록
2. 책 검색
3. 모든 책 출력
4. 종료
==============================
메뉴를 선택하시오:1
책 제목:자바 프로그래밍
책 평점:8
```

- Book 클래스는 제목(title), 평점(score)를 필드를 가진다.
  - 필요한 생성자, 접근자, 설정자 등의 메소드는 알아서 작성
- 현재까지의 등록된 책의 수는 정적 필드로 구현해보자. 객체 배열을 정적이나 동적으로 생성해본다.

```
BookTest [Java Application] C:₩Program Files
============================
1. 책 등록
2. 책 검색
3. 모든 책 출력
4. 종료
============================
메뉴를 선택하시오: 1
책 제목:자바 프로그래밍
책 평점:8
============================
1. 책 등록
2. 책 검색
3. 모든 책 출력
4. 종료
============================
메뉴를 선택하시오:1
책 제목:Python 기초
책 평점:9
============================
1. 책 등록
2. 책 검색
3. 모든 책 출력
4. 종료
============================
메뉴를 선택하시오:3
Book [title=자바 프로그래밍, score=8]
Book [title=Python 기초, score=9]
============================
1. 책 등록
2. 책 검색
3. 모든 책 출력
4. 종료
============================
메뉴를 선택하시오:
```

```
메뉴를 선택하시오:3
Book [title=자바 프로그래밍, score=8]
Book [title=Python 기초, score=9]
============================
1. 책 등록
2. 책 검색
3. 모든 책 출력
4. 종료
============================
메뉴를 선택하시오:2
책 제목:자바 프로그래밍
Book [title=자바 프로그래밍, score=8]
============================
1. 책 등록
2. 책 검색
3. 모든 책 출력
4. 종료
============================
메뉴를 선택하시오:4
```

String a = new String("Hello");
String b = new String("Hello");
이 때 두 문자열의 내용이 같은지 비교를 하기 위해서는 a.equals(b) 로 비교합니다.

```
if( a.equals(b) )
    // 동일할 때 실행할 문장
```