Chapter 06

1. 객체와 클래스에 대한 설명으로 틀린 것은 무엇입니까?

❶ 클래스는 객체를 생성하기 위한 설계도(청사진)와 같은 것이다.
❷ new 연산자로 클래스의 생성자를 호출함으로써 객체가 생성된다.
❸ 하나의 클래스로 하나의 객체만 생성할 수 있다.
❹ 객체는 클래스의 인스턴스이다.

2. 클래스의 구성 멤버가 아닌 것은 무엇입니까?

❶ 필드(field)
❷ 생성자(constructor)
❸ 메소드(method)
❹ 로컬 변수(local variable)

3. 필드, 생성자, 메소드에 대한 설명으로 틀린 것은 무엇입니까?

❶ 필드는 객체의 데이터를 저장한다.
❷ 생성자는 객체의 초기화를 담당한다.
❸ 메소드는 객체의 동작 부분으로, 실행 코드를 가지고 있는 블록이다.
❹ 클래스는 반드시 필드와 메소드를 가져야 한다.

4. 필드에 대한 설명으로 틀린 것은 무엇입니까?

❶ 필드는 메소드에서 사용할 수 있다.
❷ 인스턴스 필드 초기화는 생성자에서 할 수 있다.
❸ 필드는 반드시 생성자 선언 전에 선언되어야 한다.
❹ 필드는 초기값을 주지 않더라도 기본값으로 자동 초기화된다.

5. 생성자에 대한 설명으로 틀린 것은 무엇입니까?

❶ 객체를 생성하려면 생성자 호출이 반드시 필요한 것은 아니다.
❷ 생성자는 다른 생성자를 호출하기 위해 this()를 사용할 수 있다.
❸ 생성자가 선언되지 않으면 컴파일러가 기본 생성자를 추가한다.
❹ 외부에서 객체를 생성할 수 없도록 생성자에 private 접근 제한자를 붙일 수 있다.

확인문제

6. 메소드에 대한 설명으로 틀린 것은 무엇입니까?

1. 리턴값이 없는 메소드는 리턴 타입을 void로 해야 한다.
2. 리턴 타입이 있는 메소드는 리턴값을 지정하기 위해 반드시 return 문이 있어야 한다.
3. 매개값의 수를 모를 경우 "…"를 이용해서 매개변수를 선언할 수 있다.
4. 메소드의 이름은 중복해서 선언할 수 없다.

7. 메소드 오버로딩에 대한 설명으로 틀린 것은 무엇입니까?

1. 동일한 이름의 메소드를 여러 개 선언하는 것을 말한다.
2. 반드시 리턴 타입이 달라야 한다.
3. 매개변수의 타입, 수, 순서를 다르게 선언해야 한다.
4. 매개값의 타입 및 수에 따라 호출될 메소드가 선택된다.

8. 인스턴스 멤버와 정적 멤버에 대한 설명으로 틀린 것은 무엇입니까?

1. 정적 멤버는 static으로 선언된 필드와 메소드를 말한다.
2. 인스턴스 필드는 생성자 및 정적 블록에서 초기화될 수 있다.
3. 정적 필드와 정적 메소드는 객체 생성 없이 클래스를 통해 접근할 수 있다.
4. 인스턴스 필드와 메소드는 객체를 생성하고 사용해야 한다.

9. final 필드와 상수(static final)에 대한 설명으로 틀린 것은 무엇입니까?

1. final 필드와 상수는 초기값이 저장되면 값을 변경할 수 없다.
2. final 필드와 상수는 생성자에서 초기화될 수 있다.
3. 상수의 이름은 대문자로 작성하는 것이 관례이다.
4. 상수는 객체 생성 없이 클래스를 통해 사용할 수 있다.

10. 패키지에 대한 설명으로 틀린 것은 무엇입니까?

1. 패키지는 클래스들을 그룹화시키는 기능을 한다.
2. 클래스가 패키지에 소속되려면 패키지 선언을 반드시 해야 한다.
3. import 문은 다른 패키지의 클래스를 사용할 때 필요하다.
4. com.mycom 패키지에 소속된 클래스는 com.yourcom에 옮겨 놓아도 동작한다.

11. 접근 제한에 대한 설명으로 틀린 것은 무엇입니까?

❶ 접근 제한자는 클래스, 필드, 생성자, 메소드의 사용을 제한한다.

❷ public 접근 제한은 아무런 제한 없이 해당 요소를 사용할 수 있게 한다.

❸ default 접근 제한은 해당 클래스 내부에서만 사용을 허가한다.

❹ 외부에서 접근하지 못하도록 하려면 private 접근 제한을 해야 한다.

12. 다음 클래스에서 해당 멤버가 필드, 생성자, 메소드 중 어떤 것인지 ( ) 안에 적어보세요.

```
public class Member {
    private String name; ──────────→ (          )
    public Member(String name) { … } ──────────→ (          )
    public void setName(String name) { … } ──────────→ (          )
}
```

13. 현실 세계의 회원을 Member 클래스로 모델링하려고 합니다. 회원의 데이터로는 이름, 아이디, 패스워드, 나이가 있습니다. 이 데이터들을 가지는 Member 클래스를 선언해보세요.

데이터 이름	필드 이름	타입
이름	name	문자열
아이디	id	문자열
패스워드	password	문자열
나이	age	정수

14. 13번 문제에서 작성한 Member 클래스에 생성자를 추가하려고 합니다. 다음과 같이 name 필드와 id 필드를 외부에서 받은 값으로 초기화하도록 생성자를 선언해보세요.

```
Member user1 = new Member("홍길동", "hong");
```

확인문제

15. login() 메소드를 호출할 때에는 매개값으로 id와 password를 제공하고, logout() 메소드는 id만 매개값으로 제공하려고 합니다. 다음 조건과 예제 코드를 보고 MemberService 클래스에서 login(), logout() 메소드를 선언해보세요.

❶ login() 메소드는 매개값 id가 "hong", 매개값 password가 "12345" 일 경우에만 true로 리턴

❷ logout() 메소드는 id + "님이 로그아웃 되었습니다"가 출력되도록 할 것

리턴 타입	메소드 이름	매개변수(타입)
boolean	login	id(String), password(String)
void	logout	id(String)

```
MemberService memberService = new MemberService();
boolean result = memberService.login("hong", "12345");
if(result) {
  System.out.println("로그인 되었습니다.");
  memberService.logout("hong");
} else {
  System.out.println("id 또는 password가 올바르지 않습니다.");
}
```

16. println() 메소드는 매개값을 콘솔에 출력합니다. println() 메소드의 매개값으로는 int, boolean, double, String 타입 값을 줄 수 있습니다. 다음 조건과 예제 코드를 보고 Printer 클래스에서 println() 메소드를 오버로딩하여 선언해보세요.

리턴 타입	메소드 이름	매개변수(타입)
void	logout	value (int, boolean, double, String)

```
Printer printer = new Printer();
printer.println(10);
printer.println(true);
printer.println(5.7);
printer.println("홍길동");
```

17. 16번 문제에서는 Printer 객체를 생성하고 println() 메소드를 호출했습니다. 이번에는 Printer 객체를 생성하지 않고도 다음과 같이 호출할 수 있도록 Printer 클래스를 수정해보세요.

```
Printer.println(10);
Printer.println(true);
Printer.println(5.7);
Printer.println("홍길동");
```

18. 다음 예제 코드가 실행되면 "같은 ShopService 객체입니다."라는 메시지가 출력되도록, 싱글톤 패턴을 사용해서 ShopService 클래스를 작성해보세요.

```
ShopService obj1 = ShopService.getInstance();
ShopService obj2 = ShopService.getInstance();

if(obj1 == obj2) {
  System.out.println("같은 ShopService 객체입니다.");
} else {
  System.out.println("다른 ShopService 객체입니다.");
}
```

19. 은행 계좌 객체인 Account 객체는 잔고(balance) 필드를 가지고 있습니다. balance 필드는 음수값이 될 수 없고, 최대 백만 원까지만 저장할 수 있습니다. 외부에서 balance 필드를 마음대로 변경하지 못하도록 하고, 0 <= balance <= 1,000,000 범위의 값만 가질 수 있도록 Account 클래스를 작성해보세요.

❶ Setter와 Getter를 이용

❷ 0과 1,000,000은 MIN_BALANCE와 MAX_BALANCE 상수를 선언해서 이용

❸ Setter의 매개값이 음수이거나 백만 원을 초과하면 현재 balance 값을 유지

```
Account account = new Account();

account.setBalance(10000);
System.out.println("현재 잔고: " + account.getBalance());    //현재 잔고: 10000

account.setBalance(-100);
System.out.println("현재 잔고: " + account.getBalance());    //현재 잔고: 10000
```

확인문제

```
account.setBalance(2000000);
System.out.println("현재 잔고: " + account.getBalance());    //현재 잔고: 10000

account.setBalance(300000);
System.out.println("현재 잔고: " + account.getBalance());    //현재 잔고: 300000
```

20. 다음은 키보드로부터 계좌 정보를 입력받아 계좌를 관리하는 프로그램입니다. 계좌는 Account 객체로 생성되고 BankApplication에서 길이 100인 Account[ ] 배열로 관리됩니다. 실행 결과를 보고, Account와 BankApplication 클래스를 작성해보세요(키보드로 입력받을 때는 Scanner의 nextLine() 메소드를 사용).

[계좌생성 실행 결과] ❶

```
-----------------------------------------------------------
1.계좌생성 | 2.계좌목록 | 3.예금 | 4.출금 | 5.종료
-----------------------------------------------------------
선택> 1
-------------
계좌생성
-------------
계좌번호: 111-111
계좌주: 홍길동
초기입금액: 10000
결과: 계좌가 생성되었습니다.
-----------------------------------------------------------
1.계좌생성 | 2.계좌목록 | 3.예금 | 4.출금 | 5.종료
-----------------------------------------------------------
선택> 1
-------------
계좌생성
-------------
계좌번호: 111-222
계좌주: 강자바
초기입금액: 20000
결과: 계좌가 생성되었습니다.
```

[계좌목록 실행 결과] ❷

```
-----------------------------------------------------------
1.계좌생성 | 2.계좌목록 | 3.예금 | 4.출금 | 5.종료
-----------------------------------------------------------
선택> 2
-------------
계좌목록
-------------
111-111    홍길동    10000
111-222    강자바    20000
```

[계좌목록 실행 결과] ❸

```
-----------------------------------------------------------
1.계좌생성 | 2.계좌목록 | 3.예금 | 4.출금 | 5.종료
-----------------------------------------------------------
선택> 3
-------------
예금
-------------
계좌번호: 111-111
예금액: 5000
```

[계좌목록 실행 결과] ❹

```
-----------------------------------------------------------
1.계좌생성 | 2.계좌목록 | 3.예금 | 4.출금 | 5.종료
-----------------------------------------------------------
선택> 4
-------------
출금
-------------
계좌번호: 111-222
출금액: 3000
결과: 출금이 성공되었습니다.
```

[계좌목록 실행 결과] ❺

```
-----------------------------------------------------------
1.계좌생성 | 2.계좌목록 | 3.예금 | 4.출금 | 5.종료
-----------------------------------------------------------
선택> 2
-------------
계좌목록
-------------
111-111    홍길동    15000
111-222    강자바    17000
-----------------------------------------------------------
1.계좌생성 | 2.계좌목록 | 3.예금 | 4.출금 | 5.종료
-----------------------------------------------------------
선택> 5
프로그램 종료
```