**[ 8 ] 클래스 – 객체지향 프로그램의 기본 코딩방법**

(cf) 캡슐화(Encapsulation) : 객체가 포함한 속성과 메서드는 객체간의 관계에 있어서 감추거나 권한에 따라 접근이 가능하게 처리하는 것을 말한다. 여기에 사용되는 keyword로 접근제어자(access modifier)가 있다

목표 : 클래스의 기본적인 코딩을 구현할 수 있다.

1. 클래스 제작 (ExClass.java)

package com.lec.ex;

public class ExClass { // class명 반드시 대문자로 시작, 알파벳,숫자,\_,$ 이지만 $는 오라클사에서만, \_는 되도록 x

private 자료형 인스턴스변수( = 속성 = 필드)명;

public ExClass(){ }

public method(){

. . .

}

}

1. 패키지명
2. 클래스명
3. **데이터(인스턴스 변수=멤버변수, 필드) :** 이 데이터는생성자나 setter를 이용해서 초기화하지 않으면 객체는 null, 숫자는 0, boolean은 false로 초기화되어 들어간다
4. 생성자함수 : 클래스명과 똑같이 리턴타입이 없는 메소드를 생성자라 하며 처음 클래스형 객체를 만들때 호출된다. 모든 클래스는 반드시 하나 이상의 생성자가 있어야 한다. 만약 하나도 없으면 JVM이 디폴트 생성자를 만들어 준다(new 연산자로 호출되는 메서드)
5. **메소드**
6. **Getter & setter**
7. 생성자의 이해 : 생성자는 매개변수 있는 생성자와 매개변수 없는 생성자 등 여러 종류의 생성자를 가질 수 있다. 생성자가 없을 때는 디폴트 생성자가 컴파일러 단계에서 자동 생성한다. 한 개 이상의 생성자가 있으면 디폴트 생성자는 자동 생성되지 않는다.

**(ex1)**

**public** **class** Square {

**private** **int** side;

**public** Square() {System.***out***.println("매개변수가 없는 생성자 호출했음");}

**public** Square(**int** side) {

**this**.side = side;

System.***out***.println("매개변수가 있는 생성자 호출했음");

}

**public** **int** area() { **return** side\*side; }

**public** **int** getSide() { **return** side; }

**public** **void** setSide(**int** side) {**this**.side = side;}

}

**public** **class** SquareMain {

**public** **static** **void** main(String[] args) {

Square s1 = **new** Square();

System.***out***.println("s1의 한변의 길이는 "+s1.getSide());

Square s2 = **new** Square(10);

System.***out***.println("s2의 한변의 길이는 " + s2.getSide());

s1.setSide(10);

s2.setSide(20);

System.***out***.println("s1의 넓이는 " + s1.area());

System.***out***.println("s2의 넓이는 " + s2.area());

}

}

**(ex2) package** com.lec.ex2\_human;

**public** **class** Man {

**private** String name;

**private** **int** age;

**private** **int** height;

**private** **double** weight;

**public** Man() {

System.***out***.println("매개변수 없는 초기화 함수 호출");

}

// 파라미터값을 갖는 생성자 : 개발자가 생성자 함수를 만들면 컴파일러가 디폴트 생성자 안 만들어요

**public** Man(String name, **int** age, **int** height, **double** weight) {

System.***out***.println("데이터 4개 초기화 생성자 함수 호출");

**this**.name = name;

**this**.age = age;

**this**.height = height;

**this**.weight = weight;

}

**public** Man(String name, **int** height, **double** weight) {

System.***out***.println("키, 몸무게 초기화 생성자 함수");

**this**.name = name;

**this**.height = height;

**this**.weight = weight;

}

**public** Man(String name) {

System.***out***.println("이름 초기화 생성자 함수 호출");

**this**.name = name;

}

//파라미터가 달랑 하나인 생성자

**public** Man(**int** height){

System.***out***.println("키 초기화 생성자 함수");

**this**.height = height;

}

// 파라미터가 실수형인 생성자

**public** Man(**double** weight){

System.***out***.println("몸무게 초기화 생성자 함수");

**this**.weight = (**int**)weight;

}

**public** **double** calculateBMI() {

**double** result = weight / ( (height/100.0) \* (height/100.0));

**return** result;

}

// setter

**public** **void** setName(String name) { **this**.name = name;}

**public** **void** setAge(**int** age) {**this**.age = age;}

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

**public** **void** setWeight(**double** weight) {**this**.weight = weight;}

// getter

**public** String getName() {**return** name;}

**public** **int** getAge() {**return** age;}

**public** **int** getHeight() {**return** height;}

**public** **double** getWeight() {**return** weight;}

}

**public** **class** Woman {

**public** Woman(){

System.***out***.println("Woman 클래스의 생성자 함수 왔어요");

}

}

**package** com.lec.ex2\_man.kid;

**public** **class** Kid {

**private** String name;

**public** Kid(String name) {

**this**.name = name;

System.***out***.println("매개변수 있는 Kid 생성자");

}

}

**import** com.lec.ex2\_human.\*;

**import** com.lec.ex2\_human.kid.Kid;

**public** **class** humanMain {

**public** **static** **void** main(String[] args) {

Woman woman1 = **new** Woman();

Woman woman2 = **new** Woman();

// ------------ ------------

// ①레퍼런스변수선언 ②인스턴스생성

System.***out***.println("woman1과 woman2가 같은가? " + (woman1==woman2));

System.***out***.println("woman1과 woman2가 같은가? " + woman1.equals(woman2));

// Kid kid = new Kid(); 매개변수 있는 생성자 함수가 있을 경우 디폴트 생성자 생성 안 함

Kid kid = **new** Kid("홍아기");

Man hong = **new** Man("홍길동");

Man kim = **new** Man("김길동", 180, 70.5);

Man kim2;

kim2 = kim;

kim.setAge(185);

System.***out***.println("hong과 kim이 같은 객체인지 : " + hong.equals(kim));

System.***out***.println("kim과 kim2이 같은 객체인지 : " + kim2.equals(kim));

**double** bmi = kim.calculateBMI();

System.***out***.println(kim.getName() + " : "+ kim.getHeight() +"," +kim.getWeight());

System.***out***.println("kim의 bmi지수 : " + bmi);

}

}

클래스 객체

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

설계도 제품

객체 ≒ 인스턴스

3. this키워드의 이해

this란 객체자신

~~~~~~~~~~~~~~~~~~~~~~~~~~~~

객체지향 프로그래밍의 특징

(1) 캡슐화와 데이터은닉

(2) 다형성 : 같은 모습이지만 다른 기능 cf.오버로딩

(3) 상속

**(ex3)** 은행계좌(Account) 클래스 설계

데이터(속성) : 계좌번호(accountNo:String),

예금주(ownerName:String),

잔액(balance:int)

기능(메소드) : 예금하다(void deposit(int)).

인출하다(int withdraw(int)),

잔액조회(int getBalance())

<Account.java>

/\* 은행계좌(Account) 클래스 설계

데이터(속성) : 계좌번호, 예금주, 잔액

기능(메소드) : 예금하다. 인출하다 \*/

**public** **class** Account {

**private** String accountNo; // 계좌번호

**private** String name; // 예금주

**private** **long** balance; // 잔액

**public** Account() {

System.***out***.print("계좌오픈 감사");

System.***out***.println("계좌번호, 이름이 입력 안 되있음");

}

**public** Account(String accountNo, String name, **int** balance){

**this**.accountNo = accountNo;

**this**.name = name;

**this**.balance = balance;

System.***out***.print("계좌 오픈 감사합니다 잔액: ");

System.***out***.print(balance+"원");

}

**public** Account(String accountNo, String name) {

**this**.accountNo = accountNo;

**this**.name = name;

balance = 0; // 안해도 됨

System.***out***.print("계좌 오픈 감사합니다 잔액: ");

System.***out***.print(balance+"원");

}

**public** **void** deposit(**int** money) {// 무조건 예금

balance += money;

}

**public** **void** withdraw(**int** money) {// 잔액이 있을 경우만 인출

**if**(balance>=money) {

balance -= money;

}**else** {

System.***out***.println("잔액이 부족합니다");

}

}

**public** String getAccountNo() {**return** accountNo;}

**public** **void** setAccountNo(String accountNo) {**this**.accountNo = accountNo;}

**public** String getName() {**return** name;}

**public** **void** setName(String name) {**this**.name = name;}

**public** **long** getBalance() {**return** balance;}

**public** **void** setBalance(**long** balance) {**this**.balance = balance;}

}

**public** **class** AccountMain {

**public** **static** **void** main(String[] args) {

Account hong = **new** Account("111-111","홍길동",2200000000L);

//Account hong1 = new Account(20000);

Account hong1 = **new** Account();

System.***out***.println("hong 계좌정보 : " + hong.getAccountNo() + ", " + hong.getName());

System.***out***.println("hong1 계좌정보 : " + hong1.getAccountNo() + ", " + hong1.getName());

System.***out***.println("hong 잔액 : " + hong.getBalance());

System.***out***.println("hong1 잔액 : " + hong1.getBalance());

hong1.setAccountNo("222-222");

hong1.setName("신길동");

hong.withdraw(1000);

hong1.deposit(1000);

hong1.withdraw(2000);

}

}

**(ex4)** 직육면체의 가로, 세로, 높이, 부피나 직사각형의 가로, 세로, 넓이를 속성으로 갖는 클래스를 구현하라. 부피나 넓이를 구하는 메소드 calVolume()도 구현한다.

**public** **class** BoxOrRect {

**private** **int** width;

**private** **int** height;

**private** **int** depth;

**private** **int** volume; // 부피나 넓이를 갖음

**public** BoxOrRect() { } // 디폴트 생성자

**public** BoxOrRect(**int** width, **int** height){ // 직사각형

**this**.width = width;

**this**.height = height;

depth = 0;

calVolume();

}

**public** BoxOrRect(**int** width, **int** height, **int** depth){ // 직육면체

**this**.width = width;

**this**.height = height;

**this**.depth = depth;

calVolume();

}

**private** **void** calVolume() {

**if**(depth==0)

volume = width\*height;

**else**

volume = width\*height\*depth;

// volume = (depth==0)? (width\*height) : (width\*height\*depth);

}

// getter만

**public** **int** getWidth() {**return** width;}

**public** **int** getHeight() {**return** height;}

**public** **int** getDepth() {**return** depth;}

**public** **int** getVolume() {**return** volume;}

}

**public** **class** MainClass {

**public** **static** **void** main(String[] args) {

BoxOrRect box = **new** BoxOrRect(5,6,10);

BoxOrRect rect = **new** BoxOrRect(10,5);

System.***out***.println("박스 부피 : "+box.getVolume());

System.***out***.println("rect 넓이 : "+rect.getVolume());

}//main

}

**(ex5)** name, age, gender를 데이터로 갖고, 데이터 정보를 출력하는 print() 메소더를 갖는 PersonInfo 클래스를 구현하고 main함수를 이용하여 test 구현하시오(print() 메소드 실행 결과 : 이름=홍길동, 나이=20, 성별=m, main 메소드 안에는 PersonInfo형 객체 인스턴스를 배열로 구현해 봅니다)

**public** **class** PersonInfo {

**private** String name;

**private** **int** age;

**private** **char** gender;

**public** PersonInfo() { }

**public** PersonInfo(String name, **int** age, **char** gender) {

**this**.name = name;

**this**.age = age;

**this**.gender = gender;

}

**public** **void** print() {//이름 = 홍길동 나이 = 20 성별 = m

System.***out***.println("이름 = "+name+"\t나이 = "+age+"\t성별"+gender);

}

}

**public** **class** PersonInfoTestMain {

**public** **static** **void** main(String[] args) {

PersonInfo[] person = {**new** PersonInfo("홍길동", 20, 'm'),

**new** PersonInfo("홍길순",19,'f')

};

**for**(PersonInfo p : person) {

p.print();

}

**for**(**int** idx=0 ; idx<person.length ; idx++) {

person[idx].print();

}

PersonInfo[] iperson = **new** PersonInfo[2];

iperson[0] = **new** PersonInfo("홍길동", 20, 'm');

iperson[1] = **new** PersonInfo("홍길순",19,'f');

**for**(PersonInfo p : iperson) {

p.print();

}

}

}

**(ex6) Member 클래스를 구현하고 테스트하시고**

데이터(속성) : id, pw, name, email, address, birth, gender

메소드 : “아이디 = aaa

이름 = 홍길동

이메일 = hong@company.com

주소 = 서울 강남구

생일 = 2000-01-01

성별 = 남 ”(객체 정보)을 return하는 infoString() 메소드

main()함수 내용 : Member member = new Member(“aaa”, “xxx”, “홍길동”, “hong@company.com”,

“서울 강남구”, “2000-01-01”, ‘M’);

System.out.println(member.infoString() );