**Assignment 9**

**- 이름 : 윤지원**

1. 삼각형을 나타내는 Triangle 클래스를 작성하시오. 삼각형의 속성으로는 실숫값의 밑변과 높이를, 동작으로는 넓이 구하기(findArea), 넓이 비교(isSameArea), 접근자, 생성자를 포함한다. 또한, 작성한 클래스를 다음 코드를 사용하여 테스트하시오.

|  |
| --- |
| public class TriangleTest {  public static void main(String[] args) {  Triangle t1 = new Triangle(10.0, 5.0);  Triangle t2 = new Triangle(5.0, 10.0);  Triangle t3 = new Triangle(6.0, 8.0);    System.out.println(t1.findArea());  System.out.println(t1.isSameArea(t2));  System.out.println(t1.isSameArea(t3));  }  } |

*class* Triangle {

*private* double base;

*private* double height;

*public* *Triangle*(double base, double height) {

this.*base* = base;

this.*height* = height;

}

*public* double *getBase*() {

*return* base;

}

*public* double *getHeight*() {

*return* height;

}

*public* double *findArea*() {

*return* (base \* height) / 2;

}

*public* boolean *isSameArea*(Triangle triangle) {

*if* (triangle.*findArea*() == *findArea*()) {

*return* true;

}

*else* {

*return* false;

}

}

}

*public* *class* Assignment9\_1 {

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

Triangle t1 = *new* *Triangle*(10.0, 5.0);

Triangle t2 = *new* *Triangle*(5.0, 10.0);

Triangle t3 = *new* *Triangle*(6.0, 8.0);

System.*out*.*println*(t1.*findArea*());

System.*out*.*println*(t1.*isSameArea*(t2));

System.*out*.*println*(t1.*isSameArea*(t3));

}

}

텍스트이(가) 표시된 사진

자동 생성된 설명

2. 생산된 모든 자동차와 빨간색 자동차의 개수를 출력하는 Car 클래스를 작성하시오. 그리고 다음 코드를 사용하여 테스트하시오.

|  |
| --- |
| public class CarTest {  public static void main(String[] args) {  Car c1 = new Car(“red”);  Car c2 = new Car(“blue”);  Car c3 = new Car(“red”);    System.out.printf(“자동차 수: %d, 빨간색 자동차 수: %d,  Car.getNumberOfCar(), Car.getNumberOfRedCar());  }  } |

*class* Car {

*static* int carNum = 0;

*static* int redCarNum = 0;

String color;

*public* *Car*(String color) {

this.*color* = color;

*if* (color == "red") {

redCarNum++;

}

carNum++;

}

*public* *static* int *getNumberOfCar*() {

*return* carNum;

}

*public* *static* int *getNumberOfRedCar*() {

*return* redCarNum;

}

}

*public* *class* Assignment9\_2 {

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

Car c1 = *new* *Car*("red");

Car c2 = *new* *Car*("blue");

Car c3 = *new* *Car*("red");

System.*out*.*printf*("자동차 수: %d, 빨간색 자동차 수: %d",

Car.*getNumberOfCar*(), Car.*getNumberOfRedCar*());

}

}



3. 다음을 만족하는 클래스 Account를 작성하시오.

· 다음의 2 개의 필드를 선언

private String owner;

private long balance;

· 위 모든 필드에 대한 getter와 setter의 구현

· 위 모든 필드를 사용하는 가능한 모든 생성자의 구현

· 메소드 deposit()의 헤드는 다음과 같으며 인자인 금액을 저축하는 메소드

public long deposit(long amount)

· 메소드 withdraw()의 헤드는 다음과 같으며 인자인 금액을 인출하는 메소드

public long withdraw(long amount)

· Account 클래스의 main() 메소드에서 Account 객체를 생성하여 적당한 저축과 인출을 수행한 후 잔금을 출력

4. 위에서 구현된 메소드 withdraw()를 다음 조건에 맞게 다시 작성하시오.

· 인출 상한 금액은 잔액까지로 하며, 이 경우 이러한 상황을 출력

· 클래스 AccountTest의 main() 메소드에서 인출 상한 이상의 금액을 인출하려는 메소드를 호출하여 출력

import *java*.*util*.*Scanner*;

*public* *class* AccountTest {

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

Account ac = *new* *Account*();

ac.*inputOwner*();

ac.*inputBalance*();

ac.*inputDeposit*();

ac.*inputWithdraw*();

ac.*finalBalance*();

}

}

*class* Account {

*private* String owner;

*private* long balance;

Scanner in = *new* *Scanner*(System.*in*);

*public* String *getOwner*() {

*return* owner;

}

*public* void *setOwner*(String owner) {

this.*owner* = owner;

}

*public* void *setBalance*(long balance) {

this.*balance* = balance;

}

*public* void *inputOwner*() {

System.*out*.*print*("계좌주의 성함을 입력해주십시오: ");

owner = in.*next*();

*setOwner*(owner);

}

*public* void *inputBalance*() {

System.*out*.*printf*("%s님 계좌의 잔액을 입력해주십시오: ", owner);

balance = in.*nextLong*();

*setBalance*(balance);

}

*public* void *inputDeposit*() {

System.*out*.*print*("입금할 금액을 입력해주십시오: ");

long amount = in.*nextLong*();

*deposit*(amount);

}

*public* void *inputWithdraw*() {

System.*out*.*print*("찾으실 금액을 입력해주십시오: ");

long amount = in.*nextLong*();

*withdraw*(amount);

}

*public* long *deposit*(long amount) {

*if* (0 < amount) {

balance += amount;

System.*out*.*printf*("입금한 금액: %d, 입금 후 잔액: %d", amount, balance);

System.*out*.*println*();

} *else* {

System.*out*.*print*("입금할 금액(0보다 큰 값)을 다시 입력하십시오.");

System.*out*.*println*();

}

*return* balance;

}

// *Assignment9\_3*

// *public long withdraw(long amount){*

// *balance -= amount;*

// *System.out.printf("찾으신 금액: %d, 출금 후 잔액: %d", amount, balance);*

// *}*

// *Assignment9\_4*

*public* long *withdraw*(long amount) {

*if* (amount <= balance) {

balance -= amount;

System.*out*.*printf*("찾으신 금액: %d, 출금 후 잔액: %d", amount, balance);

System.*out*.*println*();

} *else* *if*(amount > 0) {

System.*out*.*printf*("인출 상한 금액은 %d입니다. 출금할 금액을 다시 입력하십시오.", balance);

System.*out*.*println*();

} *else* {

System.*out*.*print*("출금할 금액(0보다 큰 값)을 다시 입력하십시오.");

System.*out*.*println*();

}

*return* amount;

}

*public* void *finalBalance*() {

System.*out*.*printf*("%s님 계좌의 잔금은 %d원 입니다.", owner, balance);

}

}

