2021.05.18 Java

〈생성자, Constructor〉

```java
class PersonTest {
    String name;
    int age;
    // 생성자의 특징
    // 1. return type이 없다.
    // 2. Class 이름과 method의 이름이 같다.
    // 3. new를 할 때 호출된다.
    // ---- 생성자 ---- //
    PersonTest() { System.out.println("이것은 생성자"); }
    // ---- 생성자 ---- //

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public int getAge() { return age; }
    public void setAge(int age) { this.age = age; }
}

public class _1st_ConstructorTest {
    public static void main(String[] args) {

        PersonTest pt = new PersonTest();
        pt.setAge(10);
        pt.setName("초딩");

        System.out.printf("나는 %s 이고 %d 살이다! 덤벼라!\n", pt.getName(), pt.getAge());
    }
}
```

```
"C:\Program Files\Java\jdk-16\bin\
이것은 생성자
나는 초딩 이고 10 살이다! 덤벼라!

Process finished with exit code 0
```

PersonTest pt = new PersonTest();

》 PersonTest datatype의 pt라고 하는 PersonTest Class의 복제본을 만들겠다.

ex))

FileWriter f1 = new FileWriter("data.txt");

》 FileWriter datatype의 f1이라고 하는 "data.txt에 파일을 저장하겠다"는 상태를 가지고 있는 FileWriter Class의 복제본을 만들겠다.

t_ConstructorTest.java × _2nd_WhyConstructorTest.java ×

```java
class ConsTest {
    /*Data 저장 영역 */
    int age;
    String name;
    /*Data 저장 영역 */

    //--생성자는 아래와 같이 여러 형식으로 구성할 수 있다.
    // 이와 같이 이름이 같고 입력이 다른 케이스로 method를 사용하는 방식을
    // 함수 오버로딩이라고 한다.

    // ConsTest로 이름은 같은데 입력 인자가 다름.
    // 입력 인자의 개수를 가지고 판단하는 것이 아니라 입력인자의 DataType을 보고 판단.

    /*기능 설정 저장 영역 */
    ConsTest() { System.out.println("안녕 나는 ConsTest()"); } // 앞에 datatype void를 적으면 무슨 차이?
    ConsTest(int a){
        System.out.println("안녕 나는 ConsTest(int a)");
        age = a;
    }
    ConsTest(float f) { System.out.println("안녕 나는 ConsTest(float f)"); }
    ConsTest(int a, String n){
        System.out.println("안녕 나는 ConsTest(int a, String n)");
        name = n;
        age = a;
    }

    public int getAge() { return age; }          //public도 datatype? public이 같는 의미나 기능은??
    public String getName() { return name; }
    /*기능 설정 저장 영역 */
```

〈함수 오버로딩〉

class에 method를 정의할 때.
같은 이름이지만
서로 다른 매개변수의 형식을 가지고
있는 method 여러개를
정의 하는 방법.
java 입장에서 method의 이름이 같다고
하더라도. 매개변수의 개수나 type이
다르다면 다른 method로 인식한다.

Q. Void ConsTest로 코딩할 때와
그냥 ConsTest로 코딩할 때의 차이점이
있는지..?

Q. "public int getAge"에서 public도
int 와 마찬가지로 datatype인지..?
그리고 public의 의미와 기능은 ..?

```java
}
public class _2nd_WhyConstructorTest {
    public static void main(String[] args) {
        ConsTest ct1 = new ConsTest();
        ConsTest ct2 = new ConsTest( a: 10);
        ConsTest ct3 = new ConsTest( a: 10, n: "hi");
        ConsTest ct4 = new ConsTest( a: 40);
        ConsTest ct5 = new ConsTest( f: 3.3f);

        // 원래 여기서 사용하던 Setter는 어디...?
        // Setter가 없는데도 결과가 나오네..?
        // 즉, 생성자는 객체를 처음 생성할 때 초기값을 설정해주는 역할.
        // 결론 적으로 초기 생성에 한정해서 Setter의 역할을 대신함

        /*...*/

        System.out.printf("ct1 name = %s, age= %d\n", ct1.getName(), ct1.getAge());
        System.out.printf("ct2 name = %s, age= %d\n", ct2.getName(), ct2.getAge());
        System.out.printf("ct1 name = %s, age= %d\n", ct1.getName(), ct1.getAge());
        //ct1과 ct2는 독립적인 객체들.(나머지들도 마찬가지)
        System.out.printf("ct3 name = %s, age= %d\n", ct3.getName(), ct3.getAge());
        System.out.printf("ct4 name = %s, age= %d\n", ct4.getName(), ct4.getAge());

    }
}
```

```
"C:\Program Files\Java\jdk-16\bin\jav
안녕 나는 ConsTest()
안녕 나는 ConsTest(int a)
안녕 나는 ConsTest(int a, String n)
안녕 나는 ConsTest(int a)
안녕 나는 ConsTest(float f)
ct1 name = null, age= 0
ct2 name = null, age= 10
ct1 name = null, age= 0
ct3 name = hi, age= 10
ct4 name = null, age= 40
```

아래와 같이 일일이 다하는 것 보다

ct2.setName();

ct2.setAge();

ct2.setMajor();

vs

new Contest(name, age, major)

이게 더 효율적.

// Tip!

// 사실 Setter의 사용을 지양 해야 한다. link>> (Getter, Setter 지양하기)

// 하지만. 아직 우리는 초보자 단계

// 그러므로 Setter를 안 써서 구현을 못하는 것 보다는

// Setter를 사용해서 일단 구현할 수 있도록 하는 것이 중요

// 객체와 인스턴스의 뜻...? 알아볼 것

// 메모리 상에 올라가면 다 객체다..?

// 객체 = 인스턴스 ..?

// 우리가 사용하는 모든 데이터는 메모리(Pc상에서 DRAM)에 올라가야지만 사용할 수 있고 눈으로 볼 수 있다.

// 결국 객체(또는 인스턴스)라는 단어 자체는 "메모리에 데이터를 올렸습니다"의 추상화된 표현이라고 볼 수 있다.

```java
class Dice{

int computer;
int zu;

Dice(){
    computer = (int)(Math.random() * 6 + 1);
    zu = (int)(Math.random() * 6 + 1);
    }
    public int getDice1() { return computer; }
    public int getDice2() { return zu; }
}
public class _3rd_Quiz34 {
    public static void main(String[] args) {
        //Quiz34. 이전에 random과 제어문을 활용해서 주사위 게임을
        // 이 주사위 게임을 class 방식으로 만들어볼 것.
        // 컴퓨터 vs 사용자
        Dice dice = new Dice();

        System.out.println(dice.getDice1());
        System.out.println(dice.getDice2());

        if(dice.getDice1()>dice.getDice2()){
            System.out.println("computer win");
        } else if (dice.getDice1() == dice.getDice2()) {
            System.out.println("draw");
        }  else {
            System.out.println("zu win");
        }
    }
}
```

```
_3rd_Quiz34
"C:\Program Files\
4
1
computer win
```

〈Quiz 34〉

Class 사용해서 주사위 게임 만들기.

생성자 안에서 Math.random() 실행하도록 만들고

getter 통해서

본문 main class에 사용.

Q. 본문 main class에서 "Dice dice = new Dice();"의 의미가

'이 main class에서 Dice class를 사용하겠다'라는 의미로 해석되는지..?

println 부분을 생성자 안 쪽으로 옮김.

```java
class Dice{

int computer;
int zu;

Dice(){
    computer = (int)(Math.random() * 6 + 1);
    zu = (int)(Math.random() * 6 + 1);

    System.out.println(computer);
    System.out.println(zu);
    }
    public int getDice1() { return computer; }
    public int getDice2() { return zu; }
}
public class _3rd_Quiz34 {
    public static void main(String[] args) {
        //...
        Dice dice = new Dice();

        if(dice.getDice1()>dice.getDice2()){
            System.out.println("computer win");
        } else if (dice.getDice1() == dice.getDice2()) {
            System.out.println("draw");
        }  else {
            System.out.println("zu win");
        }
    }
}
```

Q. getter 안 쓰고 dice.computer / dice.zu로 쓰면
왜 오류가 나는지..?

```java
if(dice.computer()>dice.zu()){
    System.out.println("computer win");
} else if (dice.computer() == dice.zu()) {
    System.out.println("draw");
}  else {
    System.out.println("zu win");
}
```

생성자 안에서
computer와 zu 값을
초기화 해서 쓸 수
있다고 생각했는데
아닌가..?

```java
class TestDice {
    int comDice;
    int userDice;

    TestDice() {
        comDice = getRandomDice();
        userDice = getRandomDice();
    }
    int getRandomDice() { return (int)(Math.random() * 6 + 1); }

    Boolean userWin() {
        System.out.println(comDice + "vs" + userDice);
        if (comDice > userDice) {
            return false;
        } else if (comDice < userDice) {
            return true;
        } else {
            System.out.println("무승부.");
            return false;
        }
    }
}
public class _3rd_Quiz34_Solution {
    public static void main(String[] args) {

        TestDice td = new TestDice();

        if (td.userWin()) {
            System.out.println("사용자 승.");
        } else {
            System.out.println("컴퓨터 승.");
        }
    }
}
```

```
"C:\Program Files\J
1vs1
무승부.
컴퓨터 승.

Process finished wi
```

〈Quiz 34_강사님 풀이〉

int type(return type)의 getRandomDice라는
method를 따로 만들어서 생성자 내의 중복 제거.

Boolean type(return type)의 userWin이라는
method를 따로 만들고 각각의 조건에 true/false를 return 값으로 설정.

무승부의 경우에도 return값이 false로 지정되어
있어서 '컴퓨터 승'으로 출력됨.
이는 〈Quiz 35〉에서 풀이.

Quiz35.java

```java
class DiceTest{

int comDice;
int zuDice;

DiceTest(){
    comDice = throwDice();
    zuDice = throwDice();
    }
    int throwDice() { return (int)(Math.random() * 6 + 1); }

    int whoWin(){
        if(comDice > zuDice){
            return 0;
        } else if(comDice == zuDice){
            return 1;
        } else {
            return 2;
        }
    }
    void announce(){
        switch(whoWin()){
            case 0:
                System.out.println("Com 승");
                System.out.printf("Com:%d\n zu:%d\n", comDice, zuDice);
                break;
            case 1:
                System.out.println("무승부");
                System.out.printf("Com:%d\n zu:%d\n", comDice, zuDice);
                break;
            case 2:
                System.out.println("zu 승");
                System.out.printf("Com:%d\n zu:%d\n", comDice, zuDice);
                break;
        }
    }
}
```

〈Quiz 35〉

Class 사용해서 주사위 게임 만들기 - 무승부 포함.

'whoWin'이라는 method를 생성

》 각 상황에 따라 return값 0. 1. 2.를 주도록 한다

'announce'라는 switch문을 실행시키는 method를 생성하고

switch문에서는 'whoWin' method의 return 값을 인자로 사용

《 리턴이 없기 때문에 void datatype (Setter도 리턴이 없어서 void datatype)

```java
public class _3rd_Quiz35 {
    public static void main(String[] args) {
        //Quiz34. 이전에 random과 제어문을 활용해서 주사위 게임을 만들었던 적이 있다.
        // 이 주사위 게임을 class 방식으로 만들어볼 것.
        // 컴퓨터 vs 사용자
        // 무승부도 표현하도록 만들어볼 것.
        DiceTest dt = new DiceTest();

        dt.announce();
    }
}
```

ram Files\

〈Quiz 35_심화〉

Scanner 사용해서 사용자가 주사위게임을 계속 할지 정하게 한다.

Q. 한 줄로 Scanner class를 정의하지 않은 이유는..?

Q. 문자열- String + scan.nextLine / 문자 - ???

Q. 아래 Boolean의 초기화에 관한 질문..?

```java
import java.util.Scanner;

class DiceTest2{

int comDice;
int zuDice;

Scanner scan; // 한번에 Scanner scan = new Scanner(System.in); 으로 안 쓰고 나눠 쓴 이유는..?

DiceTest2(){
    comDice = throwDice();
    zuDice = throwDice();

     scan = new Scanner(System.in);
    }
    int throwDice() { return (int)(Math.random() * 6 + 1); }

    int whoWin(){
        if(comDice > zuDice){
            return 0;
        } else if(comDice == zuDice){
            return 1;
        } else {
            return 2;
        }
    }
    void announce(){
        switch(whoWin()){
            case 0:
                System.out.println("Com 승");
                System.out.printf("Com:%d\n zu:%d\n", comDice, zuDice);
                break;
            case 1:
                System.out.println("무승부");
                System.out.printf("Com:%d\n zu:%d\n", comDice, zuDice);
                break;
            case 2:
                System.out.println("zu 승");
                System.out.printf("Com:%d\n zu:%d\n", comDice, zuDice);
                break;
        }
    }

    Boolean reDice(){
        System.out.println("한 번더 해보겠습니까? Y(1)/N(0)");
         int choice = scan.nextInt();  // Y, N 같은 문자를 입력받고 싶을 때는 int 말고 뭘 써야 하는지...?

        Boolean isTrue= false; // 여기 이 코드가 왜 사용되어야 하는지..?
                               // false를 빼거나 case0 안 에서 Boolean isTrue = false;를 사용하면 되는거 아닌지..?
        switch(choice){
            case 0:
                isTrue = false;
                break;
            case 1:
                comDice = throwDice();
                zuDice = throwDice();
                isTrue = true;
                break;
        }
        return isTrue;
    }
}
```

```java
public class _3rd_Quiz35_2 {
    public static void main(String[] args) {

        DiceTest2 dt2 = new DiceTest2();

        do{
            dt2.announce();
        } while(dt2.reDice());
    }
}
```

```
iz35_2 ×
rogram Files\Java\jdk-16\b

한 번더 해보겠습니까? Y(1)/N(0)
1
zu 승
Com:4
 zu:6
한 번더 해보겠습니까? Y(1)/N(0)
1
zu 승
Com:4
 zu:6
한 번더 해보겠습니까? Y(1)/N(0)
1
Com 승
Com:5
 zu:3
한 번더 해보겠습니까? Y(1)/N(0)
0
```

Do-While

›› do를 실행하고 이후 조건에 따라

while을 실행 할지 말지 결정한다

OOP:

객체 지향 프로그래밍(Object-oriented programming)

장점 : 상속, 추상화, 다형성 등을 통해 코드의 재사용성을 높임. 캡슐화를 통해 보안처리가 가능함

그로인해 유지보수가 편하고 안정성이 좋고 확장이 용이함

단점: 많은 클래스로 인한 메모리 낭비가 높은 편

Instance:

<< 내 방을 class로 이용해 구분해보기 >>

class 종류 : 모든 교과서 , 모든 공책 , 모든 옷

교과서 instance : java교과서 , javascript교과서, html교과서

공책 instance : 공책1, 공책2

옷 instance : 상의1 ,하의 1, 상의2, 하의2

instance >> Q. 이 중에 틀린 개념이 있을까요??

1. 내가 맘대로 이름을 붙이는것...?
2. 메모리에서 생성된 객체를 불러오는 주소를 담은 변수 ..?
3. 특정 기능을 가진(일련의 method와 변수들을 가진) 것을 class

   class의 기능을 복제하여 다시 변수에 선언하여 재사용 하는 것을 instance..?
4. "클래스와 똑같은 기능,구성,성분"을 갖고 있는 분신이 '인스턴스(객체)'

   이 클래스와 인스턴스의 관계는 마치 '붕어빵틀'과 (거기서 찍어내진) '붕어빵'과 같다.

   하나의 붕어빵틀에서 (크기,모양,무늬 등이 똑같은) 붕어빵을 여러개 찍어낼 수 있듯이, 하나의 클래스에선 여러개의 인스턴스들을 생성해낼 수 있다.

   여기서 사용자가 이 인스턴스(객체)마다 다른 내용물을 채워넣어 다르게 이용할 수 있다는 것...?