(디지털 컨버전스) 스마트 콘텐츠와 웹 융합 응용 SW개발자 양성과정

훈련기간 : 2021.05.07 ~ 2021.12.08

KH정보교육원 | 전승리

```java
class CloneMemory {
    int[] arr;
    int num;

    public CloneMemory () {
        arr = new int[3];
        num = 3;

        for (int i = 0; i < 3; i++) {
            arr[i] = (int)(Math.random() * 6 + 1);
        }
    }
    public void reRandArr () {
        for (int i = 0; i < 3; i++) {
            arr[i] = (int)(Math.random() * 6 + 1);
        }
    }

    public void reNum () {
        num = 7;
    }

    public int[] getCloneArr () {
        return arr;
    }

    public int getCloneVariable () {
        return num;
    }

    public String toString () {
        return "arr[0] = " + arr[0] +
                ", arr[1] = " + arr[1] +
                ", arr[2] = " + arr[2];
    }
}
```

```java
public class MemoryCloneTest {
    public static void main(String[] args) {
        CloneMemory cm = new CloneMemory();

        System.out.println(cm);

        int[] save = cm.getCloneArr();

        System.out.printf("save[0] = %d, save[1] = %d, save[2] = %d\n",
                save[0], save[1], save[2]);

        cm.reRandArr();

        System.out.println("객체에 접근해 출력");
        System.out.println(cm);

        System.out.println("사전 저장 정보 출력");
        System.out.printf("save[0] = %d, save[1] = %d, save[2] = %d\n",
                save[0], save[1], save[2]);

        // 결론: 자바에서 객체에 대한 접근은 모두 메모리를 제어하는 방식이 된다.

        int num = cm.getCloneVariable();

        System.out.println("객체내 변수값 획득: " + num);

        cm.reNum();

        System.out.println("변경 후 사전 획득한 정보 재출력: " + num);

        // 결론: 앞서서도 확인했지만 값에 대해서는 복제가 이루어짐을 확인할 수 있다.
        System.out.println("변경 정보 파악: " + cm.getCloneVariable());
    }
}
```

메모리

System.out.println(cm) = 객체를 출력

int[] save = cm.getCloneArr(); = 게터로 초기화

```
arr[0] = 5, arr[1] = 2, arr[2] = 3
save[0] = 5, save[1] = 2, save[2] = 3
```

값을 바꾸고 출력해도 동일하다.

```
객체에 접근해 출력
arr[0] = 4, arr[1] = 2, arr[2] = 3
사전 저장 정보 출력
save[0] = 4, save[1] = 2, save[2] = 3
```

즉 객체에 대한 접근은 모두 메모리를 제어하는 방식

값

하지만 값의 경우에는 복제가 된 이후 객체의 변경 반영되지 않는다.

```
객체내 변수값 획득: 3
변경 후 사전 획득한 정보 재출력: 3
변경 정보 파악: 7
```

1. 6 , 3 = 12 사용자 VS 3 ,6 = 6 컴퓨터
두배는 적용됐는데 -3은 적용되지않았다.

```java
import java.util.ArrayList;

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

        ArrayList<String> lists = new ArrayList<String>();

        lists.add("빵");
        lists.add("버터");
        lists.add("우유");
        lists.add("계란");
        lists.add("쥬스");
        lists.add("베이컨");
        lists.add("파스타");
        lists.add("비프샐러드");
        lists.add("피자");

        for (String list : lists) {
            System.out.println("현재 항목은 = " + list);
        }

    }
}
```

출력 값

```
현재 항목은 = 빵
현재 항목은 = 버터
현재 항목은 = 우유
현재 항목은 = 계란
현재 항목은 = 쥬스
현재 항목은 = 베이컨
현재 항목은 = 파스타
현재 항목은 = 비프샐러드
현재 항목은 = 피자
```

ArrayList

Heap을 이용한 동적할당을 수행

사용법: ArrayList<내부에저장할데이터타입> 변수명 = new ArrayList<내부에저장할데이터타입>();

ArrayList의 배열방식 :
| 데이터1 | 다음링크 | ---> | 데이터2 | 다음링크 | ---> | 데이터3 | 다음링크 | ---> ....

일반 배열 방식 :
| 데이터1 | 데이터2 | 데이터3 | 데이터4 | 데이터5 | 데이터6 | 데이터7 | ...

ArrayList와 일반배열의 차이점

일반 배열

1.배열은 메모리가 연속적으로 배치된다
2.INDEX값이 정해져있다.

ArrayLisst 배열

1.불연속 배치다. - 일반배열에 비해 느림
2.자동할당된다.

랜덤 당첨자 문제

```java
import java.util.ArrayList;
import java.util.Arrays;

class Roulette {
    final int IMPOSTER = 3;
    ArrayList<String> nameLists;
    String[] tmpArr;
    int[] tmpIdx;
    int[] success;


    int nameLength;
    Boolean isRedundant;
```

```java
    public Roulette (String[] names) {
        nameLength = names.length;
        isRedundant = true;

        nameLists = new ArrayList<String>();
        tmpArr = new String[nameLength];
        tmpIdx = new int[nameLength];

        success = new int[IMPOSTER];

        int i = 0;

        for (String name : names) {
            tmpArr[i++] = name;
        }
    }
```

Roulette 생성자 :
배열을 입력받으면 생성자에서
이름 범위만큼 각 배열이 초기화된다

```java
public void shuffle () {
    int i = 0;

    isRedundant = true;

    do {
        tmpIdx[i] = (int)(Math.random() * nameLength);

        if (checkDuplicate(tmpIdx,i)) {
            continue;
        }

        i++;

        if (i == nameLength) {
            isRedundant = false;
        }
    } while (isRedundant);
}
```

Shuffle : 각 이름의 인덱스를 랜덤하게 변경하는 메서드

tmpIdx[i] = (int)(Math.random()*nameLength)
0부터 배열의 범위만큼 랜덤 정수를 생성

if (CheckDuplicate(tmpIdx.i)){continue}
중복체크 : 중복이 발생되지 않을때 까지 반복

if(i == nameLength) : isRedundat = false;
배열의 마지막 범위에서 반복문 종료

```java
public void checkSuccess (String[] inputArr) {
    int i = 0;

    isRedundant = true;

    do {
        success[i] = (int)(Math.random() * nameLength);

        if (checkDuplicate(success,i)) {
            continue;
        }
        System.out.printf(i + " 번째 당첨자 : %s \n",inputArr[tmpIdx[success[i]]]);
        i++;

        if (i == IMPOSTER) {
            isRedundant = false;
        }
    } while (isRedundant);
}
```

checkSuccess : 당첨자 뽑기 메서드

if (checkDuplicate) : 중복체크 메서드

입력된 배열에서 당첨자 출력

checkDuplicate 중복체크 메서드

```java
private Boolean checkDuplicate (int[] inputArr,int idx) {
    for (int i = 0; i < idx; i++) {
        if (inputArr[i] == inputArr[idx]) {
            return true;
        }
    }

    return false;
}
```

```java
public void printSuccessArr () {
    for (int i = 0; i < IMPOSTER; i++) {
        System.out.printf("success[%d] = %d\n", i, success[i]);
    }
}
```

printSuccessArr :
당첨자 3명을 출력하기 위한 메서드

```java
@Override
public String toString() {
    return "Roulette{" +
            "tmpIdx=" + Arrays.toString(tmpIdx) +
            '}';
}
```

toString:
배열이 잘 섞였는지 확인하기위한 메서드

```java
public class Prob48 {
    public static void main(String[] args) {
        String[] names = {
                "박세진", "김창욱", "김민규", "김중연", "문성호",
                "강병화", "최승현", "유종현", "한상우", "전승리",
                "이경환", "최준환", "김원석", "여인준", "이태양",
                "김윤영", "정도영", "황정아", "임초롱", "김남교",
                "이주형", "김도연", "최혜주", "김도혜", "고재권",
                "임익환", "안보미", "이상훈"
        };

        Roulette r = new Roulette(names);

        System.out.println(r);

        r.shuffle();


        System.out.println(r);
        r.checkSuccess(names);

        r.printSuccessArr();
    }
}
```

ArrayList로 출력하지 못했습니다

출력 값

```
Roulette{tmpIdx=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]}
Roulette{tmpIdx=[23, 2, 6, 10, 17, 13, 15, 26, 14, 18, 4, 24, 1, 5, 0, 16, 3, 19, 7, 11, 21, 20, 22, 9, 25, 27, 8, 12]}
0 번째 당첨자 : 박세진
1 번째 당첨자 : 김도혜
2 번째 당첨자 : 김창욱
success[0] = 14
success[1] = 0
success[2] = 12
```

번호대로 출력된다.