6일차
디지털컨버전스) 스마트 콘텐츠와 웹 융합 응용SW개발자 양성과정

강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com

학생 - namkyo Kim
siary11@naver.com

복습정리 + 숙제문제
질문은맨아래에 네모박스있습니다!

```java
public class ChallengeSolution {
    public static void main(String[] args) {
        // 현재 5~ 10 을 사용하고있음
        // 하지만 비트는 0 ~ 5를 표현해야하므로 5를 빼줘야함(빼는 바이어스값)
        final  int FIRST_BIAS = 5;
        // 두 번째는 7 ~ 10을 사용하고있음
        // 하지만 비트는 6 ~ 9를 사용해야하므로 1을 빼서 사용해야함 ( 마찬가지로 빼는 바이어스값)
        final int SECOND_BITS = 1;

        final int FIRST_RANGE = 6;
        final int FIRST_OFFSET = 5 ;

        final int SECOND_RANGE = 4;
        final int SECOND_OFFSET = 7 ;

        final int BIN = 1;
        int testBit = 0;
        int randNum;

        // int는 4바이트이므로 32비트에 해당하는 데이터를 저장할수있음
        // 우리가 testBit를 가지고 제어할 수 있는 비트의 개수는 32개
        // 숫자가 많은것이 아니므로 32개의 공간을 최대한 효율적으로 활용해야함
        // (32개라서 1개가 아까운상황)
```

```
23          // (32개라서 1개가 아까운상황)
24          // 이 상황에서 되도록 0번비트부터 순차적으로 활용하고 싶을것이고 중간에 비는 공간이 없길 원할
25
26          // 5 ~ 10,= 6개 A조
27          for (int i = 0; i < 6; i++) {
28              // 5~ 10
29              // 실제 출력값은 5 ~ 10 을 사용하되
30              // 비트 연산에서는 0 ~ 5를 사용하자는것!
31              randNum = (int)(Math.random() * FIRST_RANGE + FIRST_OFFSET);
32
33              // 2^5, 2^6, 2^7, 2^8, 2^9, 2^10
34              // 2^0 2^1 2^2 2^3 2^4 2^5
35              while ((testBit & (BIN << (randNum -FIRST_BIAS))) != 0) {
36                  randNum = (int)(Math.random() * FIRST_RANGE + FIRST_OFFSET);
37              }
38
39              System.out.printf(" 5 ~ 10randNum = %d\n", randNum);
40
41
42              testBit |= (BIN << randNum - FIRST_BIAS );
43          }
44
```

```
// 이번 추첨의 당첨자는 A조의 2번고객
if((testBit & (1 << 2)) != 0 ){
    System.out.println("당첨을 축하 드립니다." + (2 + FIRST_BIAS));
}
```

```java
import java.util.Scanner;

public class SwitchTest {
    public static void main(String[] args) {
        System.out.println("저희 상점에 방문해주셔서 감사합니다. 물건을 고르세요 ! ");

        // Boolean 이란 참, 거짓을 표현할 수 있는 데이터타입.
        Boolean isTrue = true;  // true를 지정했기 때문에 참

        Scanner sc = new Scanner(System.in);
        // 키보드입력 받기위해 스캐너클래스 임포트
        int num;

        while (isTrue){ // isTrue 는 위에서 true 로 지정했기때문에 무한루프
            System.out.print("숫자를 눌러 물건을 담으세요. : ");
            // 사용자 입력시엔 print !! 왜냐 ? 엔터가 개행되서 아랫줄로 이동됨 문제는 없지만
            num = sc.nextInt() ;     // 사용자입력값을 num변수에 담음

            // 입력된 키보드 값에 따라 적절한 처리를 하게 된다.
            // 키보드 값에 따라 처리하는 루틴은 case x에 해당한다.
            // 0번이 눌렸다면 case 0, 1번이라면 case 1과 같은 형식이다.

```

```
23
24          switch (num){   // 사용자 입력값으로 switch문 case 불르기위해 num 변수 넣음
25              // 문자 낱개로는 가능함( 홑따옴표)
26              // 현재는 숫자값이라서 문장 여러개의 문자열(쌍따옴표는 불가능함)
27              case 0:
28                  System.out.println("탈출합니다.");
29                  isTrue = false; // 0번을 입력 했을시 isTrue 가 거짓이 되고  while 무한루프 끝
30                  break;      // 말 그래도 브레이크.
31              case  1 :
32                  System.out.println("비누를 장바구니에 담았습니다. ");
33                  break;
34                  // break는 더 이상 밑으로 내려가지 않고
35                  // 이 시점에서 종료 할 수 있게 도와주는 역할을 한다.
36              case 2 :
37                  System.out.println("신발을 장바구니에 담았습니다.");
38                  break;
39              case 3 :
40                  System.out.println("에어팟을 장바구니에 담았습니다.");
41                  break;
42              default:
43                  // 이 default라는 녀석은 말 그대로 기본값에 해당합니다.
44                  // 우리가 예상치 못한 입력이 존재할 수 있음
45                  // 이 경우에 활용하는것이 default 임
```

```
1  public class ContinueTest {
2      public static void main(String[] args) {
3          for (int i = 0; i < 10; i++){
4              if(i % 2 == 0){
5                  // continue 를 만나면 아래쪽에 진행해야하는 코드가 남았더라도
6                  // 무조건 for loop의 최상단으로 이동하게 된다.
7                  // 그러므로 증감식이 진행된다.  ---> 설명을 너무 잘해주셔서 바로 이해
8                  continue;
9
10             }
11             System.out.println("i = "+ i);
12         }
13     }
14 }
15
```

```java
public class ChallengeShortcutExample {
    public static void main(String[] args) {
        // A가 500
        // B가 30 개
        // if (case A || case B)
        // 합격 케이스 500 + 불합격 500 + 추가검사500개
        // if (case B || case A)
        // 합격 케이스 30 + 불합격 970개 + 추가검사 970개

        // 1 ~ 1000까지의 숫자중 2의 배수는 A
        // 1 ~ 1000까지의 숫자중 33의 배수는 B
        int bigFrontCnt = 0, smallFrontCnt = 0;
        // 경우의 수가 두가지이기 때문에 변수를 두가지선언후 0으로 초기화
        // 작은놈먼저 아니면 큰놈먼저  이런식으로

        for (int i = 1; i <= 1000; i++){
            // if (i % 2 == 0 || i % 33 == 0){
            //      cnt++
            //}
            // 위 코드에서는 하나만 검사할때도 cnt가 증가
            // 둘 다 검사할때도 cnt가 증가함
            // 우리가 확인하고자 하는것은 검사가 총 몇 번 발생하는지다.
            // 그러므로 각 검사마다 값의 cnt(카운트)값을 증가시켜야함.
```

Run: Homework

```java
            // 그러니까 각 검사마다 카운트를 할 수 있도록
            // 카운트를 하는 코드와 검사 코드를 하나로 묶어놓은 것이다.
            // 카운트 하는 코드는 전위연산자로 배치하여 무조건0이 아니게 만들면 참이다.
            // AND 연산의 특성상 뒤의 조건을 확인해야하므로
            // 무조건 카운트는 증가하고 뒤의 조건을 확인하게 된다.

            // 전위연산자는 라인을 돌기전에 실행되기때문에 전위연산자 사용
            // 이거 전위연산자 후위연산자 항상 개념 생각하자 오늘 한번나와서 다시한번 복습하게되었다.
            if(((++bigFrontCnt != 0) && (i % 2 == 0)) ||
                    ((++bigFrontCnt != 0)  &(i % 33 == 0))){

            }
            if(((++smallFrontCnt != 0) && (i %33 == 0)) ||
                    ((++smallFrontCnt != 0) &&(i % 2 == 0))){

            }
        }
        System.out.println("큰 놈 앞에 있을때 : "+ bigFrontCnt);
        System.out.println("작은 놈이 앞에있을때 : "+ smallFrontCnt);
    }
}
```

Homework

```java
public class Quiz {
    public static void main(String[] args) {
        // 1 ~ 100 까지의 숫자중 2의 배수는 모두 곱한다.
        //여기서 5의 배수는 모두 뺀다.
        //11의 배수는 더한다.
        //중복이 발생할 경우엔 무시한다.
        //모든 값을 처리한 이후 결과값은 무엇인지 프로그래밍해보자!
        int sum = 0;

        for(int i = 1; i <= 100; i++){

            // 공배수 체크 하기위해 && 연산을 사용
            // 윗부분 잘 기억 하기 ------------
            if (i % 11 == 0 && i % 5 == 0 && i % 2 == 0){

            }else if (i % 11 == 0 && i % 5 == 0){

            }else if (i % 11 == 0 && i % 2 == 0){

            }else if (i % 5 == 0 && i % 2 == 0 ){

            }else if (i % 11 == 0){
                System.out.println("11의 배수 = " + i);
```

```java
            }else if (i % 11 == 0 && i % 2 == 0){

            }else if (i % 5 == 0 && i % 2 == 0 ){

            }else if (i % 11 == 0){
                System.out.println("11의 배수 = " + i);
                sum +=i;
            }else if (i % 5 == 0){
                System.out.println("5의 배수 = " + i);
                sum -=i;
            }else if (i % 2 == 0){
                System.out.println("2의 배수 = " + i);
                sum +=i;
            }
        }
        System.out.println("최종 결과 : " + sum);
    }
}
```

```java
public class ArrayTest {
    public static void main(String[] args) {
        // 배열은 왜 써야 할까?
        // 동일한 데이터 타입의 변수가 여러개 필요할때
        // 일일히 int a,b,c,d,e,f,g .....z 까지 해봐야 26개 밖에 안됨
        // 만약 회사에서 직원 1000명을 관리해야 한다 가정한다면
        // 이것을 일일히 변수로 선언한다면 죽을 것이다.
        // 당연히 배열을 만들어서 관리해야할 것 이다.
        int arr[] = {1,2,3,4,5};
        // int num1 = 1, int num2 = 2 , int num3 = 3;
        // 데이터가 많으면 많을수록 단일 변수 선언은 지옥을 체험하게 함
        // 그러니 우리는 심신의 안정을 위해 배열을 사용해야할 것이다.

        // 배열을 만드는방법
        // 1. stack에 할당하는 방법 ( 지역변수 )
        //  1-1. 일단은 배열의 데이터 타입 (int 같은 ) 을 적는다.
        //  1-2. 배열의 이름이 될 변수명을 적는다.
        //  1-3. 배열임을 알리기위해 []을 변수옆에 적어준다.
        //  1-4. 필요하다면 배열의 값들을 초기화한다.
        //          (이때 원소로 지정한 숫자에 따라 배열의 길이가 지정된다.)
        //  *** 가변으로 구성하고 싶다면 new를 사용해야하는데 이것은 다음주에 학습!!***

        // 아래와 같은 데이터를 살펴보자
```

Homework

```java
        // int arr[] = {1,2,3,4,5};
        // 위 데이터는 아래와 같은 형식으로 저장된다.

        // ---------------------------
        // arr | 1 | 2 | 3 | 4 | 5 |
        // ---------------------------
        //       [0] [1] [2] [3] [4]
        // 배열의 인덱스 ( 방 ) 번호는 0번부터 시작함에 주의하도록 하자
        // 그러나 방 번호가 순차적으로 증가하기 때문에
        // for 문이나 while 문등의 반복문과의 혼합구성에 있어 매우 탁월 하다.

        for (int i = 0; i < 5; i++){
            System.out.printf("arr[%d] = %d\n", i,arr[i]);

        }
    }
}

// stack(지역변수)에 할당한다는 것은 지역변수로 처리함을 의미합니다.
// 그렇기 때문에 나중에 매서드나 클래스를 학습한 이후 스택에 할당하면
// 해당 매서드 혹은 클래스 내부에서만 해당 배열이 활성화됩니다.
```

Homework

```java
public class BitShiftQuestion {
    public static void main(String[] args) {
        int num = 1;

        System.out.printf("%d << %d = %d\n", num,2, num << 2);

        // 숫자 1을 비트로 쓰면 0000 0001
        // 여기서 만약 왼쪽으로 2비트 이동시키면?
        // 0000 0001 -> 0000 0100
        // 숫자1은 4가 됨.

        // 오늘 질문나와서 다시한번 복습하게됨
    }
}
```

```java
public class Homework {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("정수(n)를 입력하면 n 번째 값을 찾아드립니다 :");

        int user = sc.nextInt();
        // 사용자입력으로 n번째 값을 찾아야 하기때문에 스캐너 사용
        int num1 = 0, num2 = 1;
        int sum = 1;
        // 피보나치수열힌트를 주셨고 변수 2개라고하셨는데 저는 왜 3개인지 ..ㅠㅠ
        // 첫 번째 값과 두 번째 값이 같아서 num1 =0 num2 = 1 로 설정
        // sum 은 for문이 시작과 동시에 출력하기 위해 1로 설정
        // 그러고 이후 피보나치수열 규칙으로 연산
        // 열심히 풀어 봤지만 코드는 이게 최선이였습니다
        // 결과값은 잘 나오지만 뭔가 이쁘지가 않네요 코드가 ,,
        for (int i = 0; i < user; i++){
            System.out.println(sum);
        sum = num1 + num2;
        num1 = num2;
        num2 = sum;

        }
```

```
정수(n)를 입력하면 n 번째 값을 찾아드립니다 :10
1
1
2
3
5
8
13
21
34
55
\
Process finished with exit code 0
```

```java
public class test {
    public static void main(String[] args) {
        //-----------------------------------------------------------
        // 바뀌지 않는 값이기 때문에 final로 지정 해주었고 아래에서 FIRST_BIAS처럼  변수를 사용해
        // 코드의 가독성을 올리고 편집성을 올린듯 하다 .
        final int FIRST_BIAS = 5;
        final int SECOND_BIAS = 1;

        final int FIRST_RANGE = 6;
        final int FIRST_OFFSET = 5;

        final int SECOND_RANGE = 4;
        final int SECOND_OFFSET = 7;

        final int BIN = 1;
        //-----------------------------------------------------------
        int testBit = 0;
        int randNum;

        // 5 ~ 10 ---> 6개 A조
        for (int i = 0; i < FIRST_RANGE; i++) {
        // 5~10 까지의 숫자가 6개범위이기 때문에 for문도 범위에 맞춰 RANGE를 설정

            randNum = (int)(Math.random() * FIRST_RANGE + FIRST_OFFSET);
            // 랜덤값 Matr.random() 은 0.0 ~ 0.99999이기 때문에  6을 곱해주면
            // 0 부터 5까지의  6개의 숫자가 나오게 되고 FIRST_OFFSET = 5
            // 를 더해주면 범위는 5 ~ 10까지의 범위내에서 랜덤으로 번호를 뽑아내서
            //  randNum에 설정 --- > 그렇게 총 6개의 숫자를 뽑아냄

            while ((testBit & (BIN << (randNum - FIRST_BIAS))) != 0) {
                // while문 조건에 randNum - FIRST_BITS 를 하는 이유는
```

```
            // while문 조건에 randNum - FIRST_BITS 를 하는 이유는
            // 5 ~ 10 사이의 숫자가 나오겠지만
            // 2^5 2^6 2^7 2^8 2^9 2^10
            // 2^0 2^1 2^2 2^3 2^4 2^5 순서대로 담기위해서 -5
            // 만약 랜덤 숫자가 5 가 나왔더라면 testBit 와 앤드연산을 해서
            // 1이 되므로 2^0 비트에 담기게되는 원리?인데 여기서는 중복을 제거 하는 구문이고
            // 중복이나오면 아래 구문처럼 다시 랜덤숫자 뽑음
            // 배열의 인덱스방 같은 느낌으로 생각하는중
            randNum = (int)(Math.random() * FIRST_RANGE + FIRST_OFFSET);
            // 만약 while문이 만족된다면 위처럼 다시 랜덤 숫자 뽑아내도록 설정
        }

        System.out.printf("5 ~ 10 randNum = %d\n", randNum);

        testBit |= (BIN << (randNum - FIRST_BIAS));
        // 여기서는 중복되지 않은 결과를 testBit 에 OR 연산을 사용해
        // or연산은 합집합개념 testBit의 2^0에 위치시킨다
        // 이런개념 !같음

    }

    System.out.println("testBit = " + testBit);
    // 그렇게 testBit의 결과값은 다꺼내보면
    // 2^0 +2^1+ 2^2+ 2^3+ 2^4+ 2^5 으로
    // 결과값은 : 63 이 나올것이다 .

    // 7 ~ 10 ---> 4개 B조
    // -------------------------위에 정리하여서 아래도 동일하다고 판단--------------------
    for (int i = 0; i < SECOND_RANGE; i++) {


```

```
    // 7 ~ 10 ---> 4개 B조
    // -------------------------위에 정리하여서 아래도 동일하다고 판단------------------
    for (int i = 0; i < SECOND_RANGE; i++) {


        randNum = (int)(Math.random() * SECOND_RANGE + SECOND_OFFSET);


        while ((testBit & (BIN << (randNum - SECOND_BIAS))) != 0) {
            randNum = (int)(Math.random() * SECOND_RANGE + SECOND_OFFSET);
        }

        System.out.printf("7 ~ 10 randNum = %d\n", randNum);

        testBit |= (BIN << (randNum - SECOND_BIAS));
    }

    System.out.println("testBit = " + testBit);

    // 예로 추첨(랜덤) 시스템을 돌렸다고 가정함
    // 이번 추첨의 당첨자는 A조의 2번(랜덤으로 나온값) 고객님입니다 ~~!!~!
    if ((testBit & (1 << 2)) != 0) {
        System.out.println("당첨을 축하드립니다! 당신이 뽑은 번호는 A조 " + (2 + FIRST_BIAS));
    }
}
}
```