

# [디지털 컨버전스] 스마트 콘텐츠와 웹 융합 응용SW 개발자 양성과정

강사 : 이상훈

학생 : 임초롱

## Bit 연산자 : And(&), Or( / ), Shift(<<,>>)

링크 <https://github.com/limcholong/LectureContents/tree/main/java/CholongLim/Day5/src>

### ① And ( & ) :

EX ) Int num1 = 10, num2 = 138 ;

10 을 2진수로 바꾸었을 때,  $2^3 + 2^1 = 1010$

138 을 2진수로 바꾸었을 때,  $2^7 + 2^3 + 2^1 = 10001010$

```
10  ==> 1010
138 ==> 10001010  AND
-----
10  ==> 00001010
```

EX ) Int num1 = 4, num2 = 68;

4를 2진수로 바꾸었을 때,  $2^2 = 100$

68을 2진수로 바꾸었을 때,  $2^6 + 2^2 = 1000100$

```
4   ==> 100
68  ==> 1000100  AND
-----
4   ==> 100
```

AND 연산은 교집합의 개념이다.

### ② Or ( / ) :

EX ) Int num1 = 10, num2 = 138 ;

10 을 2진수로 바꾸었을 때,  $2^3 + 2^1 = 1010$

138 을 2진수로 바꾸었을 때,  $2^7 + 2^3 + 2^1 = 10001010$

```
10  ==> 1010
138 ==> 10001010  OR
-----
138 ==> 10001010
```

EX ) Int num1 = 4, num2 = 68;

4를 2진수로 바꾸었을 때,  $2^2 = 100$

68을 2진수로 바꾸었을 때,  $2^6 + 2^2 = 1000100$

```
4   ==> 100
68  ==> 1000100  OR
-----
68  ==> 1000100
```

OR 연산은 합집합의 개념이다.

## Bit 연산자 : And(&), Or( / ), Shift(<<,>>)

링크 <https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day5/src/BitShiftTest.java>

### ③ Shift ( << ) : $2^n$ 을 곱한다

EX ) Int num1 = 2, num2 = 5 ;

정수 1 << 정수 2 일 때, 정수 1 x  $2^{\text{정수 2}}$

즉, 2 << 5 일 때,  $2 \times 2^5 = 2 \times 32 = 64$

따라서 2 << 5 = 64가 출력된다.

숫자 2를 비트로 쓰면 0000 0010

왼쪽으로 5비트 이동 0100 0000 <<

---

$$2 \ll 5 = 2^6 = 64$$

EX ) Int num1 = 4, num2 = 3 ;

4 << 3 일 때,  $4 \times 2^3 = 4 \times 8 = 32$

따라서 4 << 3 = 32가 출력된다.

숫자 4를 비트로 쓰면 0000 0100

왼쪽으로 3비트 이동 0010 0000 <<

---

$$4 \ll 3 = 2^5 = 32$$

### ④ Shift ( >> ) : $2^n$ 으로 나누되 소수점을 버려야 한다.

EX ) Int num1 = 2, num2 = 5 ;

System.out.printf("%d >> %d = %d\n", num1, num2, num1 >> num2);

정수1 << 정수 2 일 때, 정수 2 /  $2^{\text{정수 1}}$

즉, 2 >> 5 일 때,  $5 / 2^2 = 5 / 4 = 1.25$

따라서 2 >> 5 = 1.25지만, 소수점을 버려야 함으로 1이 출력된다.

EX ) Int num1 = 35, num2 = 4;

System.out.printf("%d >> %d = %d\n", num1, num2, num1 >> num2);

35 >> 4 일 때,  $35 / 2^4 = 35 / 16 = 2.1875$

따라서 35 >> 4 = 2.1875 지만, 소수점을 버려야 함으로 2가 출력된다.

결론.

Shift 연산은  $2^n$ 을 곱하거나 나눈다.

Shift 연산은 정수형 끼리 가능하다.

# Switch ~ case 조건문 기본 개념 및 예제

링크 <https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day6/src/SwitchTest.java>

```
4 public static void main(String[] args) {
5     System.out.println("저희 상점에 방문해주셔서 감사합니다. 물건을 고르십시오 호갱님!");
6
7     ① Boolean isTrue = true;
8
9     Scanner scan = new Scanner(System.in);
10    int num;
11
12    ② while (isTrue) {
13        System.out.print("숫자를 눌러 물건을 담으세요: ");
14
15        num = scan.nextInt();
16
17        ③ switch (num) {
18            case 0:
19                System.out.println("탈출합니다.");
20                isTrue = false;
21                break;
22            case 1:
23                System.out.println("비누를 장바구니에 담았습니다.");
24                break;
25
26            ④ case 2:
27                System.out.println("신발을 장바구니에 담았습니다.");
28                break;
29
30            case 3:
31                System.out.println("에어팟을 장바구니에 담았습니다.");
32                break;
33
34            default:
35                System.out.println("그런건 없습니다!");
36                break;
37        }
38    }
```

## < 기본 개념 >

**Switch ~ case** : 조건문. Switch 와 case가 같이 사용되어야 한다.  
Switch 와 case의 데이터 타입이 일치해야 한다.  
정수 혹은 문자열로 사용 가능하다.

**Boolean** : 참, 거짓을 표현할 수 있는 데이터 타입이다.

**Break** : 더 이상 밑으로 내려가지 않고, 특정 시점에서 종료할 수 있도록 도와주는 역할을 한다.

**Default** : 기본값. 예상치 못한 입력값에 대해서 default 를 활용한다.

① Boolean istrue = true;

② while (true)일 때 내부내용이 반복될 것이다. istrue = true

③ switch (num) 은 num = scan.nextInt(); 로 키보드 입력을 받을 것이며, Int 이다. 따라서 case도 Int이어야 한다.

④ 1,2,3을 입력했을 때 각각의 상황이 일어나며, while을 반복한다.  
예상치 못한 값을 입력했을 때는 default 가 실행되며 while을 반복한다.  
0 입력 시, 탈출합니다가 출력되며. isTrue는 거짓이 되고, while의 반복은 끝난다.

# Continue 기본 개념 및 예제

링크 <https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day6/src/ContinueTest.java>

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

Run: BitTest

```
"C:\Program Files\Java\jdk-15.0.2\bin\
i = 1
i = 3
i = 5
i = 7
i = 9

Process finished with exit code 0
```

## < 기본 개념 >

**Continue** : 아래 진행되어야 하는 코드가 남아있더라도 최상단 for문으로 이동한다.

### ① for loop

for (int i = 0; i < 10; i++)

int i는 0에서 시작해서 증감하며 i < 10일때 반복된다.

### ② if ( i % 2 == 0 )

i가 2의 배수(짝수)일 때, continue가 실행된다.

a) i = 1일때, 1은 2의 배수가 아니므로 ③ 이 실행된다.

b) i = 2일때, 2는 2의 배수이므로 continue가 실행되어, for (int i = 0; i < 10; i++)으로 되돌아가고 i++ 로 i 값이 증감한다.

### ③ System.out.println("i = " + i);

i = 1 , i = 3 , i = 5 , i = 7 , i = 9 가 출력된다.

(짝수에 대해서 continue가 실행되어 홀수만 출력이 된다.)

# Array (배열) 의 기본 개념 및 예제

링크 <https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day6/src/ArrayTest.java>

## < 기본 개념 >

### Array :

동일한 데이터 타입의 변수가 여러 개 필요할 때 사용된다.  
배열을 배우기 전에는 `int num = 1`, `num 2 = 2`, `num3 = 3` 등  
각 변수를 선언하였지만,  
배열 사용 시 `int arr [] = { 1, 2, 3 }; ...` 으로 한 번에 가능하다.  
For 문이나 while 문 등의 반복문과의 혼합구성에 탁월하다.

배열 만드는 방법 : **stack**( 지역 변수 ) 에 할당하는 방법

데이터 타입 변수명 [] =  
`int arr []` 로 만든다.

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

-----  
arr | 1 | 2 | 3 | 4 | 5 |  
-----

인덱스(방) [0] [1] [2] [3] [4]

배열의 인덱스(방) 번호는 0번부터 시작한다.

`arr[0] = 1`, `arr[1] = 2`, `arr[2] = 3`, `arr[3] = 4`, `arr[4] = 5`

## < 기본 개념 >

### stack :

stack에 할당한다는 것은 지역변수로 처리함을 의미한다.  
매서드나 클래스에서 stack 할당 시,  
해당 매서드나 클래스 내부에서만 배열이 활성화 된다.

배열에서 String 사용

```
4 public static void main(String[] args) {  
5     String Alphabet [] = {"a", "b", "c", "d", "e"};  
6  
7     for (int i = 0; i < 5; i++) {  
8         System.out.println(Alphabet[i]);  
9     }  
10 }
```

Run: BitTest x

↑	a
↓	b
↺	c
↻	d
⏏	e

-----  
Alphabet | a | b | c | d | e |  
-----

인덱스(방) [0] [1] [2] [3] [4]

## 25번 : 복습 문제

링크 [https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day6/src/Day6\\_Quiz1.java](https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day6/src/Day6_Quiz1.java)

```
10 int sum = 0;
11
12 for (int i = 1; i <= 100; i++) {
13     if (i % 11 == 0 && i % 5 == 0 && i % 2 == 0) {
14         System.out.println("110의 배수 = " + i);
15     } else if (i % 11 == 0 && i % 5 == 0) {
16         System.out.println("55의 배수 = " + i);
17     } else if (i % 11 == 0 && i % 2 == 0) {
18         System.out.println("22의 배수 = " + i);
19     } else if (i % 5 == 0 && i % 2 == 0) {
20         System.out.println("10의 배수 = " + i);
21     } else if (i % 11 == 0) {
22         System.out.println("11의 배수 = " + i);
23         sum += i;
24     } else if (i % 5 == 0) {
25         System.out.println("5의 배수 = " + i);
26         sum -= i;
27     } else if (i % 2 == 0) {
28         System.out.println("2의 배수 = " + i);
29         // System.out.printf("2의 배수 = %d\n", i);
30         sum += i;
31     }
32 }
33
34 System.out.println("최종 결과 = " + sum);
35 }
36 }
```

### 25번 문제 내용 :

1 ~ 100 까지의 숫자 중 2의 배수는 모두 더한다.  
여기서 5의 배수는 모두 뺀다. 11의 배수는 더한다.  
중복이 발생할 경우엔 무시하며, 모든 값을 처리한 이후 결과값은?

If ( 110배수라면 출력,  
else if (55의 배수라면 출력)  
else if (22의 배수라면 출력)  
else if (10의 배수라면 출력)

else if (11의 배수일때, sum = sum + 11의 배수  
sum = 0 + 11의 배수(11의 배수는 더한다.)

else if ( 5의 배수일때, sum = sum - 5의 배수  
sum = ( 0 + 11의 배수) - 5의 배수

else if ( 2의 배수일때, sum = sum + 2의 배수  
sum = ((( 0 + 11의 배수) - 5의 배수) + 2의 배수

// 위 식에서 중복이 발생할 경우는 무시를 어떻게 프로그래밍 한 것인지  
이해하지 못하겠습니다.

## 27번 : 복습 문제 (Challenge)

링크 [https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day6/src/Day6\\_Quiz2.java](https://github.com/limcholong/LectureContents/blob/main/java/CholongLim/Day6/src/Day6_Quiz2.java)

```
5 // 27번 문제
6
7 // 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987 ...
8 // fn = f(n-2) + f(n-1)
9
10 Scanner scan = new Scanner(System.in);
11 int num;
12 int f1 = 1;
13 int f2 = 1;
14 int sum ;
15
16 for(int i = 0; i < 20 ; i++) {
17     System.out.print("피보나치 수열의 n번째 값은?(n을 입력하시오) : ");
18     num = scan.nextInt();
19
20     if ( num == 0 ) {
21         System.out.println("입력을 종료합니다.");
22         break;
23     } else if ( num == 1 ) {
24         System.out.printf("%d번째 피보나치 수열 값은 : %d\n", num, f1);
25     } else if ( num == 2 ) {
26         System.out.printf("%d번째 피보나치 수열 값은 : %d\n", num, f2);
27     } else {
28         System.out.println();
29         // num > 2 일때, 출력해야하는 값.
30     }
31 }
32
33 }
```

### 27번 문제 내용 :

아래와 같은 형태의 숫자 배치가 있다.

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89 ... (피보나치 수열)  
사용자가 15를 입력했을 때 15번째 값을 구하도록 해보자.  
( n을 입력하면 n번째 값을 구하는 프로그래밍 )

피보나치 수열 :

$$fn = f(n-2) + f(n-1)$$

위 내용을 System.out.println() 안에 포함될 값으로 문제에 적용하지 못하겠습니다.