## 변수

int: 정수형(숫자 - 음수, 양수)

long: 비슷함

float: 소수점

double: 조금 더 정밀한 소수점

## if문

if(조건문){

수행문;

}

## if-else문

if(조건문){수행문1;}

else{수행문2;}

```
int a = 5;
int b = 10;
if(a>b){
System.out.println("max =a");
}else{
System.out.println("max =b");
}
```

↓동일한 코드

```
int a = 5;
int b = 10;
int max;
max = (a>b) ? a:b;
System.out.println(max);
```

**if-else if-else문** if가 만족하지 않으면 else if 조건 확인, if, if, if 문 선호

if(조건1){

문장1;

}else if(조건2){

문장2;

}else if(조건2){

문장3;

}else {

문장4;

}

## switch문

switch(조건){

case 값1: 수행문1;

break;

case 값2: 수행문2;

break;

case 값3: 수행문3;

break;

case 값4: 수행문4;

break;

default: 수행문5;}

```
Scanner scan = new canner(System.in);

boolean isLoop = true;

// isLoop 가 true 인 동안 반복

while (isLoop) {

System.out.print("숫자를 입력하세요:
");

int inputNumber = scan.nextInt();

switch (inputNumber) {

  case 0: // case 0: 의 의미는 '입력된
숫자가 0 이면' 이라는 뜻을 가집니다.

  System.out.println("종료");

  isLoop = false; // 숫자 0 이 들어오는
경우 isLoop 를 false(거짓)으로 바꿈

  break;

  case 1:

  System.out.println("입금");

  break;

  case 2:

 System.out.println("출금");

  break;

  case 3:

  System.out.println("조회");

  break;

  default: // 0 ~ 3 사이의 숫자가 아닌
경우

  System.out.println("그런 명령은
  존재하지 않습니다!");

  break;} 생략가능???????
```

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

사람의 키보드 입력을 받고 싶으면 입력하세요.

* 변수 = scan.nextInt();

키보드 입력으로 int 타입을 수신한다면 nextInt()를 사용합니다. Double, Float 등 변경가능

출력문:

while 반복문이 참인 동안 루프는 돌아갑니다.

입력된 숫자가 0이면 "종료"라는 출력문이 뜨고 루프가 거짓으로 바뀌기 때문에 수행을 중단한 후 while문을 빠져나갑니다.

입력된 숫자가 1이면 "입급"

입력된 숫자가 2이면 "출금"

입력된 숫자가 3이면 "조회"

입력된 숫자가 0~3이 아니면 " 그런 명령은 존재하지 않습니다.

## while문

while (조건식;) {조건이 만족될 동안 반복할 코드}

```
int idx = 0;

final char ch = 'A';

while (idx < 10) {

System.out.println("idx: " + idx + ", 안녕: " + (char)(ch + idx));

idx++;}
```

출력문:

idx: 0, 안녕: A

..... idx: 9, 안녕: J

## for문

for (초기화식; 조건식; 증감식){수행문}

```
final int START = 3;

final int END = 10;

int sum = 0;

int count = 0;

for (int idx = START; idx<= END ; idx++){

sum = sum + idx;

System.out.println("count= " + (++count) + ", sum = " +sum);
```

출력문:

count= 1, sum = 3

.....count=8, sum = 52

*++count: count값이 1증가한 후 변수에 대입

```
final int START = 3;

final int END = 10;

int index = START;

for (; index < END; ) {

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

}

// for 문은 조건 파트가 참이 동안은 언제든지 반복된다는 것입니다.

// 초기화식 이나 증감식은 곁다리로 볼 수 있다.
```

출력문:

index = 3

.....index=9

```
for ( int i = START; i<= END; i++){

if(i%2 == 0){continue;}

//continue 는 skip 과 동일합니다.

(2 의 배수는 넘겨라)

System.out.println("i= " + i);
```

출력문:

i=3

i=5

i=7

i=9

## 랜덤숫자만들기

```
int randomNumber = (int)Math.random();

double randomNumber = Math.random();
```

```
int MAX=6;

int MIN=1;

int randomValue=0;

randomValue =

(int)(Math.random()*(MAX - MIN + 1))
+ MIN;
```

↓동일한 코드

```
int MAX=6;

int MIN=1;

int randomValue=0;

randomValue =

(int)(Math.random() * MAX) + 1;
```

1~6까지의 랜덤 숫자 생성

## 배열

데이터타입[ ] 변수이름 = new 자료형[갯수];

ex1) int[] studentIDs = new int[10];

int형 요소가 10개인 배열 선언

ex2) int[] studentIDs = ~~new int[]~~{101,102,103};
//생략 O, int형 요소가 3개인 배열 생성

ex3) int[] studentIDs; // 배열자료형 선언

studentIDs = **new int[]**{101,102,103}

// 생략 X, int형 요소가 3개인 배열 생성

num [3] =25;

num 배열 4번째 요소에 값25를 저장

age = num[3];

age 변수에 num 4번째 요소에 값을 저장

## 배열-Static에 할당

## 배열-Heap에 할당

// 결론: Stack은 중괄호{} 내에서 사용됨

// Heap은 new 하고 이후로 사용됨

```
Int[] numberArray = { 1, 2, 3, 4,
5 };

for (int i = 0; i <
numberArray.length; i++)
```

```
// 배열은 메모리 공간상에 순차적으로
배치됩니다.

// 주의할 부분이라면 배열의 시작이
0 부터라는 것에 주의를 해주세요.
```

numberArray[0] = 1

numberArray[1]은 숫자 2 를 표현하며

numberArray[2]는 숫자 3

numberArray[3]는 숫자 4

numberArray[4]는 숫자 5

## Class

class 이름{

속성을 선언}

## Class 생성하기

클래스를 사용하기 위해서는 클래스를 생성하여야함

class 이름 인스턴스이름 = new class 이름();

인스턴스.속성 = ex)갯수;

클래스와 인스턴스 선언 파일은 다른것이다.

클래스: 첫글자 영문 대문자

인스턴스: 첫글자 영문 소문자

## 생성자

인스턴스 생성 시 new 키워드와 함께 사용했던 생성자

person personLEE = new person();

객체가 생성될 때 매개변수를 입력받아서 생성하고 싶다.

public person(String name){

name = pname;}

생성자는 반환형이 없다.

생성자가 하나라도 있다면 default 생성자는 제공되지 않는다. 임의로 넣어줘야 에러가 나지 않는다.

## 매서드 작성

public 함수반환형(리턴타입) 이름 ( ){ 수행할 작업}

```
int add (int num1, int num2){

int result

result = num1 +num2;

return result;

}
```

3 -> [ ] -> 9 (리턴 타입 int)

버튼 누름 -> [ ] -> true (리턴 타입 boolean)

1 -> [ ] -> "예금" (리턴 타입 String)

회원 정보 -> [ ] (리턴 타입 void)

== 정확히는 리턴하지 않음을 의미함

## 클래스 내 매서드 구현하기

```
public String getStudentName() {

return studentName;

}

public void setstudentName(String
name) {

studentName = name;

}
```

```
class 참치선물세트{
int 일반;
int 야채;
int 고추;
    참치선물세트(int 일반, 야채, 고추){
this.일반 = 일반;
this.야채 = 야채;
this.고추 = 고추;
}
}
```

this: 우리의