

데이터 강제 형 변환

- 자료형을 원하는 자료형으로 '강제'로 변환 시키는 것.

큰 데이터 - > 작은 데이터 손실 확률 높음

강제 형 변환 방법-

[자료형을 변환 시키고 싶은 값 또는 변수 앞에 (자료형)을 작성]

(temp==임시라는 의미)

```
//ex) double temp = 3.14
//    int num = (int)temp;
//    출력 결과 num = 3;
double형 temp를 int형으로 강제 형 변환 시킨다
```

```
//int to byte 강제 형 변환
int iNum = 290;
byte bNum = (byte)iNum;

//(byte) 형 변환 명령이 없을 시 타입 미스 매치 제한이 일어남.
```


```
System.out.println("int to byte 강제 형 변환");
System.out.println("before:"+iNum);
System.out.println("after:"+bNum);
```

출력 결과

```
int to byte 강제 형 변환
before:290
after:34
```

- 변수 : 메모리에 값을 저장 할 수 있는 공간.

[변수와 메모리 구조]

- 
- 3가지 종류가 있다.

출력문 -

```
System.out.println(); -- 한 줄 출력 (줄 바꿈 효과가 일어난다.)
System.out.print(); -- 단순 출력.
```

printf 의 활용

%d 정수형 (데시머리?10진수라는 의미)

- %5d : 5 칸을 확보하고 오른쪽 정렬.
- %-5d : 5 칸을 확보하고 왼쪽 정렬.

%c (char) -문자

%f (float) - 실수 (실수들은 무조건 %f)
- %2f : 소수점 아래 2 자리 까지만 표시

%s 문자열

%b 논리형.

Escape



\t 블럭 건너뛰기

\n 출력 후 줄 넘김

\\ 역슬래쉬

\' 작은 따옴표 [안써도 되긴한다].

\\" 큰따옴표

\\u 유니코드

--System.out.println("\\u0041"); //유니코드 (16진수) 번호로 출력



스캐너 생성의 입력 받기

ex) int input = sc.nextInt();

연산자 operator



우선 순위.
'()' . " [] ' 1순위

2순위와 3순위 + - 의 차이
2순위 +- 음수 양수를 표현한다
3순위 계산식을 의미한다.

다른 클래스에서 참조하는방법.

public void main(){ }~선언 후

메인 메소드에서 초기화.

- 클래스를 분리하고 재사용 , 참조 하는 법에 대하여 배웠다.
- 혼자 사용할 수 없는 % 기호 등을 조심하자...


산술 연산자


증감 연산자 (if / for 원툴 연산자)

전위 연산
int ++num (다른 연산보다 먼저 실행됨.)

후위 연산
int num++ (다른 연산 후 연산함.)

int a = 3;
int b = 5;
int c = a++ + --b;
최종적으로 : a=4 b=4 c=7

비교 연산자.
> < <= >= == !=


논리 연산자.

&& == AND
|| == OR
True&&True == True
False||False == false
&& 둘 다 True 일 경우에만 True
true && true = true
true && false = false
false && false = false
|| 둘 중 하나만 True이여도 True
true || true = true
true || false = true
false|| false =false

오늘 배운 것.

- 연산자
- 강제 형변환
- 변수와 메모리 구조 (3구조)
- 출력문
- println 한줄출력
 - printf 포맷 출력
 - print 단순 출력
- 이스케이프 문자

- \t
- \n
- \\
- \'
- \"

- \u

Scannaer Class

```
import java.util.Scanner;
```

자바 기본 제공 jre에서 유틸 Scanner을 import하여 사용한다

```
Sacnner [변수명] = new Scanner(System.in);
```

시스템으로부터 데이터를 입력 받는다..

```
(데이터 타입)(변수명) = sc.[데이터 타입에 맞는 수식];
```

여러 종류의 연산자를 배웠다

- 괄호 연산자
 - (), []
- 단항 연산자
 - +, -, !, (자료형), ++, --, ~
- 산술 연산자
 - *, /
 - +, -
- 비교 연산자
 - > < >= <=
 - ==, !=
- 일반 논리 연산자
 - &&, ||
- 순수대입 연산자
 - = 오른쪽의 데이터를 왼쪽으로 대입한다.
- 산술대입
 - +=, -=, *=, /=, %=

위에서 부터 순서대로 우선 순위를 가진다.

같은 패키지 내에서

```
Public void main(){  
}
```

의 형태로 생성 후 다른 클래스 안에서 참조해 올 수 있다

Static을 추가시 동적이기 때문에 참조 할 수 없다.

- 혼자 사용할 수 없는 % 기호 등을 조심하자...