**5/19 목.**

복습)

변수를 선언

= → 대입 연산자.

// 사용자로부터 입력을 받기 위해 Scanner의 Instance 생성한다.

Scanner input = new Scanner(System.in);

→ 위치는 크게 영향 주지 않지만, 흐름상 위쪽에 위치하는 것이 더 좋다.

x = input.nextInt();

x → 변수 공간

Scanner 안에 있는 nextInt를 호출하기위해 . 을 찍어준다.

프로그램 안에서 변하지 않는 값 (숫자) 상수 → 리터럴 상수

변수를 사용하는 목적을 이해하자.

수학 → 미지수를 찾을 때 주로 사용;

**프로그램** → **‘변할 수 있는 수’** 라고 이해하는게 더 좋음.

변수 이름은 의미 있는 단어 사용하는 것이 좋음.

수식 작성시 **연산**이 먼저 일어나는 것들을 괄호( )로 묶음.

나머지 연산자 % 활용법이 매우 많음.

오류가 나지 않게끔, 중간중간에 값을 테스트 한다. → 전체 작성 후 테스트 시 오류 부분 찾기 힘듬.

-------------------------------------------------------------------------------------------------------

**코드를 통일하자.**

띄어쓰기 하나, **사소한 것**부터 시작.

ex) 위 코드에서 ( )를 코드뒤에 붙였으면, 다같이 붙이고, 연산자도 공백이 있다면 공백을 주고하자.

실수는 정수 표현 가능

정수는 실수 표현 불가능 → 정수 \* 실수는 실수를 따라간다. 하나만 실수여도 됨.

double r3 = 4 \* (r \* r \* r) / 3;

오른쪽의 연산 결과만 받아준다.

결국 왼쪽 double r3 과는 상관없이 오른쪽의 값이 중요.

→ 실수를 선언 해주어야 제대로 된 선언 결과

( ) 안의 수식이 먼저 계산됨으로 4 \* r^3에서 r^3은 실수이기 때문에 정수와 실수의 계산이 됨으로, 실수를 따라오게 되어 / 3 부분에서도 실수 결과값이 나온다.

double로 상환했지만, 그저 담기는 통으로 이해하면 됨.

**변수의 선언**

이름은 숫자로 시작 할 수 없다.

영소문자로 시작해야한다.

대문자 소문자 구별.

키워드(keyword)사용금지.

\_ , $ 문자 처음에 사용 가능은 하나 특별한 경우로 제한.

// 구분을 위한 공백은 줄바꿈 하여도 인식 가능하다.

System.out.println("몫은 '" + div + "'이고“

+ "나머지는 '" + left +"'입니다.");

**서식 (Format)**

System.out.printf(); 서식이 있는 문자를 출력하는 메소드.

public class Formatex {

public static void main(String[] args) {

System.out.printf("%dHello %dWorld%d\n", 1, 2, 3);

System.out.printf("Hello %dWorld", 2);

}

}

→ 1Hello 2World3

Hello 2World 의 값이 출력된다.

%d → 쉼표로 구별해서, %d의 자리에 ‘ , ’ 뒤에오는 정수를 넣어준다.

\n → 줄바꿈

**자료형(data type)**

컴퓨터 공간은 ‘무한하지 않음’

→ 데이터 파일들을 공간의 크기에 맞게 사용.

자바(java)는 강력한 규제(strong)타입. <--> weak타입으로 javascript 등이있다.

기초형(primitive type)과 참조형(reference type)

기초형은 값을 그대로 넣어줌.

참조형은 instance에 끈을 만들어 주는 것처럼 instance를 참조해서 사용할 수 있게함.

**기초형**(primitive type)→ 정수형, 실수형, 문자형, 논리형

정수형 : byte(가장 빠름), short, int, long(큰 수 표현)

실수형 : float(덜 정확), double(더 정확 – float의 두배정도의 정확도) - 근삿값을 사용. 정확한 수치 아님. → 오차가 생김. 아주 정확한 계산에서는 사용 X.

논리형 : boolean → 참 거짓

문자형 : char → character ‘문자열’과는 다름. 한 글자.

‘문자열’ → 여러 문자들의 합.

|  |  |  |
| --- | --- | --- |
| 종류 | 사용 방법 | 예 |
| 클래스명 | 각 단어의 첫글자는 대문자 | StaffMember, ItemProducer |
| 변수명, 메소드명 | 소문자로 시작되어 2번째 단어의 첫글자는 대문자 | width, payRate, acctNumber, getMonthDays(), fillRect(), |
| 상수 | 상수는 모든 글자를 대문자 | MAX\_NUMBER |

05\_19\_ 목

정수, 실수, 문자, 불련타입, 값 교환, 복합 대입 연산자, 증감연산자, 관계연산자. 논리연산자

int, float, char, boolean, Swap, AutoInc,

**대입 연산자** = 는

변수를 초기화 할 때, 변경된 값으로 넣을 때.

**증감 연산자**

주로 ‘++’, ‘--’ 많이 씀 → 1씩 증가, 감소

**복합 대입 연산자** → 대입연산자와 산술연산자를 합쳐 놓은 연산자.

**관계 연산자** – 물음 → 같은가? 다른가? p.89

결과는 true(참), false(거짓)으로 계산.

결과 → **boolean** 값인 true, false 로 출력.

== 같냐?

!= 다르냐?

> 크냐?

< 작냐?

>= 크거나 같냐?

<= 작거나 같냐?

// 0 ~ 100 사이??

int number = 33;

// 한번에 하나씩만 가능하다. 왼쪽이 먼저.

System.out.println(0 <= number <= 100);

-컴파일 에러