5/17 화요일 정리

html : 정적 내용물 화면에 바로 보여서 인식 쉬움 (웹페이지)

java : 동적 내용물 but 바로 티나지 않음.

Program : 원하는 방향으로 동작 할 수 있게끔.

* **프로그래밍 언어 공부는 프로그램을 만들어야 제대로 할 수 있다** -

→ Scratch 사용해서 만들기 연습하기

언어 – 2진법. 전기 on – 1 off – 0

논리적 연산 → 모든 것 표현 가능

0,1 로 이루어진 – 기계어

기본적인(간단한) 명령어들로 이루어진 – 어셈블리어

**Low Level(저급 언어)** - 기계적 언어 컴퓨터에 가까운 언어

 번역과정필요. - 컴파일(compile)

**High Level(고급 언어)** - 인간 친화적 언어 (Java, C, C++ 등), 문자 체계 – 대부분 영문자

작성 – 메모장

번역 – 컴파일 도구  컴파일러(Compiler)

실행 – 실행 환경  Runtime Evironments

**JVM** (Java Virtual Machine) → 작성한 코드를 읽어서 각 환경에 맞게끔 실행시켜줌.

**JRE** (Java Runtime Environment) → 클래스 라이브러리 및 특정 Java 프로그램이 실행해야 하는 기타 리소스를 제공하는 소프트웨어 계층.

**JDK** (Java Development Kit) → Java 환경에서 돌아가는 프로그램을 개발하는 데 필요한 툴들을 모아놓은 소프트웨어 패키지.

**JDK(밑의 것들을 포함) > JRE > JVM**

**- cmd (명령 프롬포트)**

----자바는 CMD로 실행.

**GUI** (Graphic) → 그림으로 조작

**CLI** (Command Line Interface) → 명령어로 조작

**- JAVA의 실행 순서**

Java Developer → .java(source code) → Compiler → .class(bytecode) → JVM → low-level code → Computer

**컴파일 언어** → sc → 번역 → 실행 - 조금 더 빠른 실행, 정확한 문법 필요

인터프리터 언어 → sc → 그때 그때 바로실행 - 유연한 문법

자바최신 버전을 안쓰는 이유 : 새로운충돌을 방지

가장 많이 쓰는 버전. 8버전, 그다음 11버전

실무 때 사용 : 버전 마다 변경 사항 공부하면 됨.

* **인코딩**

어떤 정보를 정해진 규칙(Code)에 따라 변환하는 것(en-code-ing);

ASCII → 최초의 문자 코드, 미국에서 개발, 미국에서 쓰이는 영어가 쓰기 적합한 형태

- **UTF-8**은 가장 많이 사용되는 가변 길이 유니코드 인코딩

**- 주석**

/\*

범위 주석

\*/

// 한 줄 주석(comment) - 사람을 위해 메모 남겨놓는 것

// → 한 줄 주석. 컴파일러가 한 줄을 무시. //가 시작하는 곳부터 한줄.

/\*

This is a big hurdle to leap over it you have to be able to create the program text somewhere, compile it successfully, load it, run it, and find out where your output went. With these mechanical details mastered, everything else is comparatively easy.

\*/

**Java** 는 객체지향 언어.

class 이름은 영단어, 첫글자는 대문자! ex) class Hello

{ } 중괄호 : class의 범위 영역을 표시. 반드시 정확해야함. 짝이 맞지 않으면 에러

들여쓰기를 통해 안의 내용물 작성. { ~ } 중괄호 시작과 끝까지.

→ **Main Method** : Java로 구현된 프로그램이 시작될 때 가장 먼저 실행되는 메소드.

**method** - 동작을 실행하게 하는 명령어.

**call** – 호출 메소드의 이름을 불러 주는 것.

**코드를 쓸 때 html에서처럼 대소문자 구별 확실히 하기!**

System.out.println("Hello World") → 문장(statement) 문장의 끝은 ;

문자열 “Hello World”를 출력하라

System.out.println → 출력하는 명령어 ***(출력방법 1)***

Hello World → 출력

“ ”(쌍따옴표)로 감사져 있는 것 → 문자열(string)

띄어쓰기 금지. 두 번째 단어의 첫 글자를 대문자로 표기

javac –encoding UTF-8 Hello.java

입력 → UTF-8 언어로 인코딩 하라.

순서대로 나타나는 것 (위에서 아래로) → **절차수행.**

System.out.println("10"); // + 연산자(operator)

System.out.println("11");

System.out.println("12");

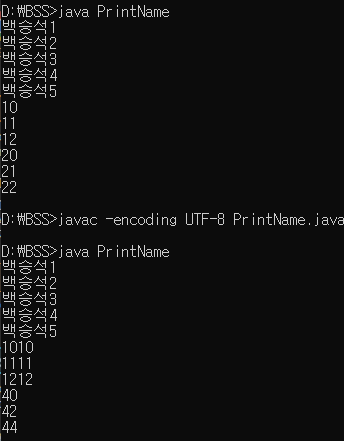
→ 문자열로 인식 → 1010, 1111, 1212 로 출력

System.out.println(20);

System.out.println(21);

System.out.println(22);

→ 정수값으로 인식 (소수점도 가능) → 41, 42, 44 로 출력



10 + " " + "10" → 10 10 으로 출력.... 공백을 넣음

줄 사이 공백 넣기

→ System.out.println(); 괄호안에 아무것도 넣지 않으면 공백 줄이 생김.

***- 출력 방법 2***

System.out.print(1); → 줄바꿈이 사라지고 연달아 출력됨.

System.out.print(2);

System.out.print(3);

- **println** → 한 줄씩 출력

- **print** → 바로 옆에 연달아 출력

System.out.println("1 2 3\n4 5 6\n7 8 9");

\n 은 1개 문자로 출력. 다음줄로 넘어가는 줄바꿈 문자. 대행문자.

/\* \(백슬래시) \*/

public class Add {

public static void main(String args[]) {

// int : 정수형 data type (자료형)

int x; int 부분은 변수의 형태

int y; x, y 부분은 변수의 이름 부분.

int sum; → 변수 선언(variable declare)

// **초기화 (initialize)** → 박스에 값을 가져오지 못하면 에러가 뜸.

x = 100; → // 대입연산자 (assignment)는 좌,우로 나뉘어짐

y = 200; → 오른쪽 값을 왼쪽 값에 입력해줌. (복사 하듯이.----그대로 들고오는 느낌보단 복사)

sum = x + y; → 더하기가 먼저 일어나, 300의 값을 sum 박스에 복사

System.out.println(sum); → 출력하는 메소드 호출. 변수 이름 자체를 부름.

return; //메소드의 끝을 의미 (생략 가능)

System.out.println(); → 도달 할 수 없는 문장. return;키워드가 있기 때문.

}

}

변수(variable) → 기억을 위한 공간.

변수 선언(variable declare) → 공간을 마련해달라 부탁.

변수의 이름은 알아보기 쉽고, **의미**를 담아서 짓기.

규칙 : 변수 이름은 **영소문자**로 시작한다. 대소문자를 구별

// 변수이름

// 영소문자로 시작 (대소문자 구별)

// 숫자도 올 수 있음

// abc12345

// 공백 x

// booktitle => bookTitle 대문자로 단어 구분.

// 특수문자 x 굳이 사용 => -, \_ (바, 언더바) 두가지 가능

// 의미가 있는 키워드는 사용 불가

중복되는 이름 X.

이름의 변경이 힘듬. → 고민 후 선정.

System.out.println(x – y); → 계산 값을 한번만 쓸 때 주로 씀.

int minus;

minus = x - y;

System.out.println(minus); → 계산 값을 여러번 쓸 때 주로 씀.

System.out.println(50 / 0);

0으로 나눌 수 없기 때문에 에러발생. → 밑의 결과값마저 나오지 않음. 프로그램의 흐름이 종료.

System.out.println("10 곱하기 5는 " + 10 \* 5); → 연산자의 순서. 곱하기먼저.

먼저 수행되는 것을 괄호로 묶어주어 이해쉽게 돕기.

System.out.println("22 나누기 4는 " + (22 / 4)); → 정수로만 출력.

System.out.println("22 나누기 4는 " + (22.0 / 4.0)); → 소수점을 포함해서 표기하면, 소수점까지 출력