**Lecture 3**

<Java 프로그램 기본 구조>

1. 클래스 : 객체 지향 언어에서 프로그램을 개발하는 단위

2. 메서드 : 수행할 작업을 나열한 코드의 모임

3. 실행문 : 작업을 지시하는 변수 선언, 값 저장, 메서드 호출 등의 코드

<식별자>

: 보고 느낄 수 있는 모든 사물(객체)들을 각각 구별할 수 있는 것을 의미

<변수(Variable)>

: 프로그램은 메모리에 데이터를 보관하고, 각 메모리를 변수(Variable)로 구분

<데이터 기본 타입>

정수/실수/논리/문자형 변수의 사용 예

– 정수형: 3, -7, 0, ...

– 실수형: 3.14, -9.81, 0.0, ...

– 논리형: true, false

– 문자형: 'A, 'B', ...

< 연산자 (Operator) >

- 증감 연산자

: 증감 연산자는 변숫값을 1만큼 증가 또는 감소

– 값의 증가는 ++ 기호를, 감소는 -- 기호를 사용

– 변수의 앞에 있는 경우에는 전위 연산, 뒤에 있는 경우엔 후위 연산을 수행

- 비교 연산자

: 비교 연산자는 논리 타입을 제외한 기초 타입에만 사용할 수 있지만, ==와 !=는 모든 기초 타입에 사용

- 논리 연산자

: 논리 연산자는 피연산자의 조건을 결합해서 true와 false를 조사하며, 논리 타입에만 사용

<화면에 데이터 출력>

– System.out.println() : 내용을 출력한 후 행을 바꿈

– System.out.print() : 내용을 출력만 하고 행은 바꾸지 않음

– System.out.printf() : 포맷을 지정(형식 문자열)해서 출력

<형식 데이터 출력>

• %s - 해당 위치에 String 값을 삽입

• %d - 해당 위치에 int 값을 삽입

• %f - 해당 위치에 double 값을 삽입

• %b - 해당 위치에 boolean 값을 삽입

**Lecture 4**

<타입 변환>

자동 타입 변환: 작은 타입의 데이터는 같은 범주의 큰 데이터 타입으로 자동 변환됨

1. 문자열 🡪 숫자 변환

Int x = Integer.parseInt(“3”); //문자열을 정수로 변환하는 메소드

1. 숫자 🡪 문자열 변환

String a = “” + 1234; //문자열 1234 생성

String b = Integer.toString(123); //문자열 123 생성

<데이터 입력>

– 프로그램의 첫 행에 다음을 추가해 Scanner 클래스의 경로 이름을 컴파일러에 알림

**» import java.util.Scanner;**

– 키보드로 데이터를 입력 받기 위해 System.in 객체와 연결된 Scanner 객체를 생성

**» Scanner in = new Scanner(System.in);**

– Scanner 클래스가 제공하는 다양한 메서드를 이용해 키보드로 데이터를 입력 받음

**» int x = in.nextInt();** // 정수를 읽어 변수 x에 대입

<메소드(method)>

- 개념: 입력값에 따른 결과를 반환하는 일련의 코드 묶음

– 호출부: 메소드를 사용하는 영역, 입력 값을 전달하여 그 결과를 돌려받는 코드

– 정의부: 입력값을 받아 결과를 만드는 일련의 코드 묶음

- 매개변수의 종류

1) 형식 매개변수 (formal parameter)

: 선언에서 사용된 매개변수

2) 실 매개변수 (actual parameter)

: 호출에서 사용된 매개변수로 인자(argument)라고도 함

-메소드의 유형

1) 매개변수의 유무

🡪 매개변수 없음, 단일 매개변수, 다중 매개변수

1. 반환값의 유무

🡪 void 메소드: 메소드의 반환값은 존재하지 않는 메소드

(이 경우 반환 타입은 void가 되고, return 키워드는 생략 가능)

**Lecture 5**

<제어문>

- 프로그램의 흐름에 영향을 주고 때에 따라 제어가 가능하도록 하는 것

-제어문의 종류 🡪 조건문, 반복문, 분기문 등

<조건문>

-상황별 실행 흐름을 만드는 문법

-조건문의 종류 🡪 if문, if-else문, 다우 if문, 중첩 if문, switch 문, 삼항 연산자

<if문>

-if 문은 조건에 따른 수행 여부를 결정

Ex) 졸리면? 잠을 잔다 🡪 if (졸리면) { (1) 조건식

잠을 잔다. (2) 조건식이 참일 경우, 수행될 코드

}

<if-else>

-if-else 문은 두 조건식 중 하나를 선택할 때 사용, 조건식이 참인 경우 if 문의 코드가, 거짓인 경우 else 문의 코드가 수행

<다중 if문>

-추가 조건을 붙여야 할 때 사용

-조건문의 수행은 윗줄에서부터 차례대로 진행

-다중 if 문을 활용하면 여러 상황 중 하나를 선택할 수 있음

<중첩 if문>

-if 문에 다른 if문이 포함되는 것을 중첩 if문이라고 함

<switch문>

-switch문은 상황(case)에 따라 수행 코드를 나누는 문법

-0개 이상의 case 절과 0이나 1개의 default절로 구성

-switch 연산식의 주의 사항 🡪 가능한 모든 값에 대하여 일치하는 case 레이블이 없으면 오류가 발생

**Lecture 6**

<조건문(cont)>

- 삼항 연산자: 조건식에 따른 반환값을 지정하는 문법

: 조건 식이 참이면 A, 거짓이라면 B가 반환

(조건식) ? A : B;

<반복문>

1. While 문: 반복 횟수가 조건에 따라 변할 때 사용하기 좋은 반복문
2. For 문: 반복 횟수가 명확할 때 사용하기 편리한 반복문

<분기문>

1. Break 문: 반복문을 탈출하는 문법
2. Continue 문: 실행 흐름을 다음 반복으로 넘기는 문법

: 반복문의 모든 코드를 수행하지 않은 채 다음 반복으로 이동

<메소드 오버로딩 (Method Overloading)>

: 메소드 이름은 같지만 메소드 시그니처가 다른 메소드를 정의하는 것

-메소드 시그니처 (Method Signature)

: 메소드 이름, 매개변수의 개수, 매개변수의 타입과 순서를 의미