**3/19 - 9장 중첩 인터페이스 중첩클래스**

**14장 람다식**

**3/ 22 - 18장 입출력**

**9장 중첩 클래스**

**class A{--> Outer클래스=외부클래스**

**멤버변수**

**메소드(){지역변수}**

**생성자**

**class B{ -->inner 클래스=내부클래스=중첩클래스**

**멤버변수**

**메소드(){지역변수}**

**생성자 }**

**}**

**class A{ B b1 = new B(); }**

**class B{}**

**class C{ B b1 = new B(); }**

**class X**

**class A{**

**B b1 = new B();**

**class B{}**

**}**

**class C{ 오류 B b1 = new B();**

**A.B b1 = new A().new B();**

**}**

**class F { f1(){} f2(){} }**

**interface I1 { mi1() ; }**

**interface I2 { mi2() ; }**

**interface I3 { mi3() ; }**

**class D extends F implements I1, I2, I3{**

**mi1(){오버라이딩 의무화}**

**mi2(){오버라이딩 의무화 }**

**mi3(){오버라이딩 의무화 }**

**f1(){오버라이딩 선택적 }**

**f2(){오버라이딩 선택적 }**

**d1(){추가 메소드}**

**}**

**class D {**

**interface I1{ mi1();}**

**class I1sub implements I1{**

**mi1(){오버라이딩 의무화}**

**}**

**class I2sub implements I2{**

**mi2(){오버라이딩 의무화}**

**}**

**class I3sub implements I3{**

**mi3(){오버라이딩 의무화}**

**}**

**class FSub extends F{**

**f1(){오버라이딩 선택적 }**

**f2(){오버라이딩 선택적 }**

**}**

**d1(){추가 메소드}**

**}**

**static – 객체 생성 이전(프로그램 최초 시작) 메모리 할당 저장**

**객체 1개값 공유**

**클래스명.xxxx**

**static 중첩내부클래스--> static 메소드 가능**

**``중첩내부클래스--> static 메소드 불가능**

**class A{**

**class B{ void m() {구현 } }**

**interface I{ mi() ; }**

**class C implements I{ mi(){ 오버라이딩 구현} }**

**--> I 인터페이스 상속 하위클래스 C 이름 정의**

**C c1 = new C();**

**c1.mi();**

**C c2 = new C();**

**new I() { mi(){ 오버라이딩 구현} }.mi();**

**--> I 인터페이스 상속 하위클래스 익명 클래스 정의+객체생성**

**== I 인터페이스 타입 익명 객체 생성**

**==> 1번 객체 생성**

**pdf 파일 - 개인적 교안 –**

**ppt 파일 – 출판사 교수강사 배포 – 서약서**

**1>제네릭/ 중첩 인터페이스/ 람다**

**===> 자바 진화 문법 다양 / 간결**

**상위클래스나 인터페이스명 변수=**

**new 상위클래스나 인터페이스명( ) {  
 상위클래스나 인터페이스 상속 하위클래스 정의**

**=무명**

**추가멤버변수**

**추가메소드**

**상속 오버라이딩**

**};**

**- instance 멤버 내부클래스**

**- static 멤버 내부클래스**

**- 메소드내부- 지역 내부클래스**

**- 익명 내부클래스 – 상속 인터페이스 구현 / 객체 생성**

**- lambda 식**

**제네릭 람다 익명클래스**

**메소드 정의 – 클래스 틀 없을 수도 있다.**

**람다 – 매개변수 선언 실행 문장 블록 정의**

**무명의 메소드**

**class A{**

**int add(int x, int y){**

**return x +y;**

**}**

**}**

**(x, y)->{return x +y;};**

**(타입 변수명 , , , ) -> { 실행 문장 블록 }**

**(변수명 , , ,) -> { 실행 문장 블록 }**

**( ) -> { 실행 문장 블록 }**

**변수명 -> { 실행 문장 블록 }**

**변수명 -> 실행 문장;**

**(변수명, , , ) -> return x+y;**

**(변수명, , , ) -> x+y;**

**(매개변수 ) -> { 실행 문장 블록 정의 }**

**==> 무명 메소드, 무명 객체 정의 -->????**

**14.3 타겟 타입과 함수적 인터페이스**

**class A{ void ma() { } }**

**abstract class B { void ma() { } abstract void mb(); }**

**interface C { void ma() ; abstract void mb(); }**

**new C() {**

**public void ma() {......}**

**public void mb(){....}**

**}.ma();**

**new C() {**

**public void ma() {......}**

**public void mb(){....}**

**}.mb();**

**C c1 = new C() {**

**public void ma() {......}**

**public void mb(){....}**

**};**

**c1.ma();**

**c1.mb();**

**@ FunctionalInterface - 컴파일러 메소드 1개 있는지 검사**

**interface I {**

**1 개 정의 약속 . FunctionalInterface**

**void m1(int x);**

**}**

**I i1 = (x)->{,,,,,.......};// I 타겟 타입 람다식**

**I 상속 하위클래스(무명,유명..) 정의 객체 생성**

**i1. m1(10);**

**688p – 711p**

**표준 api 함수적 인터페이스**

**interface I1{**

**void m1();---> 매개변수 없고 리턴값 없는 람다식**

**}**

**interface I2{**

**int m2(int i);---> 매개변수 int 1개, 리턴값 int 람다식**

**}**

**interface I3{**

**int m2(int i);---> 매개변수 int 1개, 리턴값 int 람다식**

**}**

**api**

**interface Consumer{**

**void accept(Object);--> 매개변수 모든 객체 1개, {..},리턴없다**

**}**

**interface Supplier{**

**Object get();--> 매개변수 없고 리턴값 있다**

**}**

**Comsumer c = o -> { o.toString()...........};**

**Supplier s = ( ) -> { o.toString()......return o;};**

**c.accept(new Employee());**

**18 입출력 / 네트워크**

**java.io.50여개클래스**

**영문자 ,숫자, 특수문자 몇개들– 모든 컴퓨터 동일 표현**

**한글 한문 =다국어 – 컴퓨터 표현 방식 다르다**

**영문자**

**A -> 65-->A**

**가-->1000 --->????**

**---> UNICODE**

**모든 문자 -- 2바이트 표현**

**char c = "가";**

|  |  |  |
| --- | --- | --- |
|  | **1바이트**  **이미지 동영상 음향** | **2바이트**  **문자기반** |
| **입력** | **InputStream**  **xxxxInputStream**  **read()**  **close()** | **Reader**  **xxxReader**  **read()**  **close()** |
| **출력** | **OutputStream**  **xxxOutputStream**  **write()**  **close()** | **Writer**  **xxxWrtier**  **write()**  **close()** |
| **File – 파일과 디렉토리 정보 제공 클래스**  **파일 입출력 기능 없다** | | |

**abstract class InputStream{**

**abstract void read() ; System.in.read();-> 하위클래스 오버라이딩**

**abstract void read(byte b[]) ;**

**abstract void read() ;**

**void close(){ }**

**}**

**class FileInputStream extends InputStream**

**class BufferedInputStream extends InputStream**

**class DataInputStream extends InputStream**

**==> 1바이트씩 입력 공통**

**read() -**

**write()**

**close**

**java.util.Scanner 클래스- 입출력 편리**

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

**키보드-->Scanner 타입 (정수 실수 String..)--->자바  
int <- s.nextInt();**

**double <-s.nextDouble()**

**char c->s.nextChar()**

**boolean b=s.nextBoolean() --> true**

**String s = s.next()-> 공백 이전 단어**

**String s = s.nextLine()-> 공백 포함 한 라인**

**3월 23일 시험 자동 채점/과락**

**과제(lc) 제출= 강사 채점**

자바프로그램

키보드입력

모니터출력

**콘솔입출력**

**모니터출력**

**System.out.println()**

**java.lang.System클래스 – 자바 실행 컴퓨터 정보**

**System.out 변수 – 컴퓨터 모니터 표준출력장치 1개**

**class System{**

**static PrintStream out;**

**System.out.println()**

**키보드입력**

**System.in.read();**

**java.lang.System클래스 – 자바 실행 컴퓨터 정보**

**System.in 변수 – 컴퓨터 키보드 표준입력장치 1개**

**class System{**

**static PrintStream out;**

**static InputStream in;**

**int result = System.in.read();--> 1바이트씩 입력**

**a-> System.in.read()->result -> 97**

**A------> 65**

**0 -----> 48**

**특수문자**

**한글, 정수,실수 x**

**콘솔=표준입출력**

**java.io.\***

**java.net.\***