# 0303

2 차원 배열의 선언

1차원 배열 참조 변수들을 묶음으로 다루는 배열
  == 2 차원 배열;

- 1 차원 배열들의 주소를 묶음으로 참조하는 배열
- 수업 끝나고 이 부분 다시 한번 해석해보기.
- heap영역에 선언된 int[행][열]에서 행에 해당되는 부분은
- 열에 해당하는 값의 주소들을 가지고 있는
- ' 배열 참조 변수 이다 '이다.

2 차원 배열의 For문을 이용한 누적.

- row는 행을 의미한다.
- arr.length를 비교하면 2 차원 배열의 길이를 확인한다
- 안쪽 For문은  외부의 row가 행을 의미 한다는 걸 이용해 반복 시킨다

# OOP

06_OOP.pdf 2 MB

## Object Oriented Programmin

**객체 지향 프로그래밍**

프로그램 : 명령어들의 집합.
프로그래밍 : 명령어를 나열하는 것.

## Chap01 - 객체지향언어.

**객체( Object / 客體) : 독립적으로 각각 구분해서 인식할 수 있는 모든것.**
-물건 , 물체 ,요소,대상 ,목적
ex) 키보드 , 김치 ,옆집 아저씨 , 나(자신),미세먼지,모래 한알,판소리 무형 문화제

**지향 (Oriented) : 지정된 방향 : 어떠한 것을 목표함 .**
**객체 지향 : 독립적으로 각각 구분되어서 인식 할 수 있는 것을 목표한다.**

**( 컴퓨터에서의 )언어(Language) :**

## 객체 지향 언어 :

### 독립적으로 각각 구분되게 인식 할 수있는 것을

### 목표하는 코딩 방식

**********
**| 현실 세계는 사물이나 개념처럼 독립되고 구분되는 각각의 객체로 이루어져 있으며, |**

| 발생하는 모든 사건들은 객체간의 상호작용이다. |
| 이 개념을 컴퓨터로 옮겨 놓아 만들어낸 것이 객체지향 언어이다. |
**********

1.캡슐화(Encapsulation) : 코드들을 모아 놓는 역할.(외부의 침입을 보호)
2.상속성(Inheritance) : 부모 객체가 자식 객체에게 넘겨주는 것
3.다형성(Polymorphsism) : 다양한 형태를 가질 수 있다.(상황에 맞게 변할 수 있다)
+
4.추상화(Abstraction) : 구체적이지 않다.
속성 (명사): 객체를 구별 하기 위한 값(data) :  나 (키, 이름 , 나이,사는 곳)
기능 (동사): 객체가 할 수 있는 것들(동작/행동) : 나 (코딩한다/이동한다/밥을 먹는다 등)

'new'를 통해서 생성되는 것들은 객체를 만드는 것이다.

# 클래스(Class) : 객체의 특성(속성, 기능)에 대한 정의를 한 것 : 설계도

변수 : 메모리에 값을 저장하기 위한 공간.

## 변수

1개의 자료형
1개의 데이터

## 배열

1개의 자료형
여러개의 데이터

## 구조체

여러개의 자료형
여러개의 데이트

## 클래스

구조체 + 여러가지의 기능
클래스 = 추상화 +캡슐화 가 필수적이다.

# 추상화란

- 필요한 공통적인 부분을 추출하고
  구체적(자세한)내용을 제거하는
- 유연성을 얻을 수 있다

# New

Heap 영역에
People Class 안에 있는
6개의 변수와
2개의 메소드라는 정의가 할당된다.
할당만 되고 참조는 안됐다.
class는 사용자 정의 자료형이라고 불린다
클래스 다이어 그램;

# 캡슐화

속성과 기능들을 하나로 묶는 역할
정보 은닉
속성 == 값 ==data  <<< 여기에 직접 접근을 제한한다

getter / setter  ==  매우 중요

직접 접근은 원칙적으로 제한되기 때문에
클래스 안에 간접 접근을 위한 getter 와 setter을 이용한다.

# PDF 복습

## 모르는 부분 아래에 적어두고 따로 공부

**프로그래밍 기초**

자바 특징 4가지

- 운영체제에 독립적
- 객체지향 프로그래밍
- 상대적으로 사용하기 쉽다 (명확한 코드 작성 가능 ) 다른 언어의 단점 보완 (포인터 / 메모리 관리)
- 자동 메모리 관리 garbage collection

jvm (java virtual machine)

- 자바를 실행하기 위한 가상 기계로 운영체제에 관계 없이 독립적으로 동작한다.
- java code = > 컴파일러가 번역해줌 = > 자바 byte code

jdk > jre >jvm

메모장으로 하기 힘드니깐 IDE로 개발한다.

main 메소드

public static void main(String[]args){}

고정된 형태로 JAVA Application을 실행하는데 무조건 필요한 메소드

**변수**

메모리에 값을 저장하기 위한 공간을 변수라 한다.

- 변수의 특징
  - 가독성이 증가
  - 재사용성 증가로 인한 코드 감소
  - 유지 보수 용이

변수 선언 (메모리에 공간을 할당하는 것)

- 자료형 변수명 ;

기본 자료형 8가지

- **boolean(1byte)**
- **char (2byte)**
- **byte(1byte) / short (2byte) / int (4byte) / long (8byte)**
- **float(4byte) / double(8byte)**

명명 규칙

- 대소문자 구분 되고 길이 제한 없다
- 예약어 사용하면 안된다 색 있는 애들
- 숫자 시작 안됨
- 특수 문자 보통 사용 안 함 (할 수는 있다)
- 카멜 표기법 (낙타 표기법)

값 대입

- 변수에 들어가는 값은 리터럴(iteral)이라 표현한다

리터럴 표기법

- F
- l
- " "
- ' '

상수
항상 같은 수
final

casting (형변환)
같은 자료형만 연산이 가능하기 때문에
형 변환을 통해 연산 시킨다

자동
컴파일러에 의해서 값의 범위가 큰 자료형과 작은 자료형이 연산시 자동으로
작은 자료형을 큰 자료형으로 컴파일러가 바꿔준다

강제
강제로 자료의 형태를 바꾸는데
1. 값의 범위가 큰 것을 작은 걸로 강제로 바꿀 때
2. 'A'라는 값이 유니코드 몇 번 인지 궁금할 때
3. 표기법을 바꾸고 싶을 때

**연산자**
산술 연산자
증감 연산자
비교 (true / false)
논리 (and / or) (&&=TTT /|| FFF)
논리부정(논리 값을 반대로 적용하는 것 ) (ex) boolea su = true; // !su = false
복합 대입 ++/-- 증감이 아닌 다른 수들 증감 시키고 싶을 때 사용 ex) i += 10; / i -= 2;

**제어문**
수행 흐름을 바꾸고 싶을 때
if = 조건이 True일 때 코드 수행
switch = ex) 달력 ,날자 , 점수 등 값이 정해져 있는 사이에서 고르는 것

반복문
For는 언제 끝날지 알수 있을 때
while은 언제 끝날지 모를 때 (Do ~while 은 한번은 무조건 실행)

분기문 (branch) : 반복문 도중에 반복을 브레이크 하거나 다시 한번 반복 시킬 때
break;
continue;

**배열**
같은 자료형의 변수를 묶음으로 다루는 것
번호는 0번 부터 부여된다.

얕은 복사
배열의 주소만

깊은 복사
배열의 값까지 '복제'

2차원 배열
1차원 배열을 묶음으로 사용하는 것.
int[][]arr = new int[3][2];
가변 배열은 열의 길이를 각각 다르게 설정하는 것.

**opp**
객체 지향 언어

각각의 객체들이 상호작용을 하는 것을 컴퓨터로 옮겨둔 것.

독립 되어 각각 인식될 수 있도록 하는 것.

객체는 속성과 기능이라는 것으로 구성되어 있다
객체를 집어 넣기 위해 클래스에 넣는다.

class를 new 연산자를 이용해 heap영역에 할당하면 객체가 만들어진다
class == 객체를 만들기 위한 설계도;

추상화
만들려고 하는 기능과 속성 중
필요한 공통점을 추출하고 불필요한 부분을 제거하는 것.
캡슐화
속성과 기능을 하나의 { } 로 묶은것을 캡슐화 라고 한다.

데이터의 직접 접근제한을 원칙으로 한다.
private 을 이용해 보호한다

직접 접근을 차단하고 간접 접근을 통해 이용 시킨다
간접 접근은 getter 그리고 setter으로 구현된다.