

클래스, 객체

클래스객체기초

- 클래스(class) vs. 객체(object)

- new

- 필드변수 vs. 지역변수

- 객체 vs. 객체참조변수

클래스, 객체

클래스

- 대부분의 경우 객체 생성 및 객체에 대한 처리를 위한 코드
- 클래스는 자료형으로 사용될 수 있음

```
public class Date {  
    int year;  
    int month;  
    int day;  
}
```

필드(field)

```
public class Test {  
    public static void main(String[] args) {  
        Date date; ← Date 클래스의 객체 참조 변수 선언  
        date=new Date(); ← Date 객체 메모리 할당, 생성자 Date() 호출, 객체 메모리 참조 값 반환  
        date.year=2019; ← Date 객체의 year 필드에 2019 대입  
        date.month=3;  
        date.day=25;  
        System.out.println(date.year+"년"+date.month+"월"+date.day+"일");  
    }  
}
```

객체

- 클래스가 실체화된 것
- 클래스 관련 데이터 혹은 메소드의 모음

클래스 자료형

Date date = new Date() ;

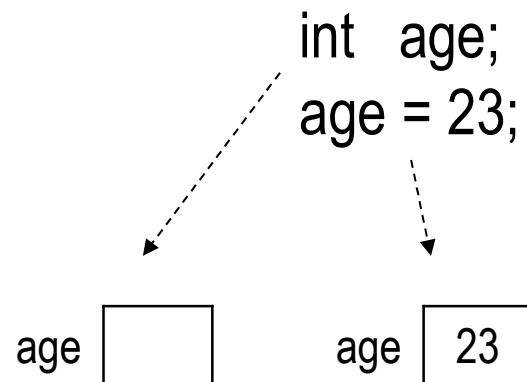
객체 참조 변수

- 클래스 Date의 객체 생성 (메모리 할당)
- 객체 생성자 Date() 호출
- 객체 메모리에 대한 참조 값 반환

기본 자료형 vs. 클래스 자료형

기본자료형

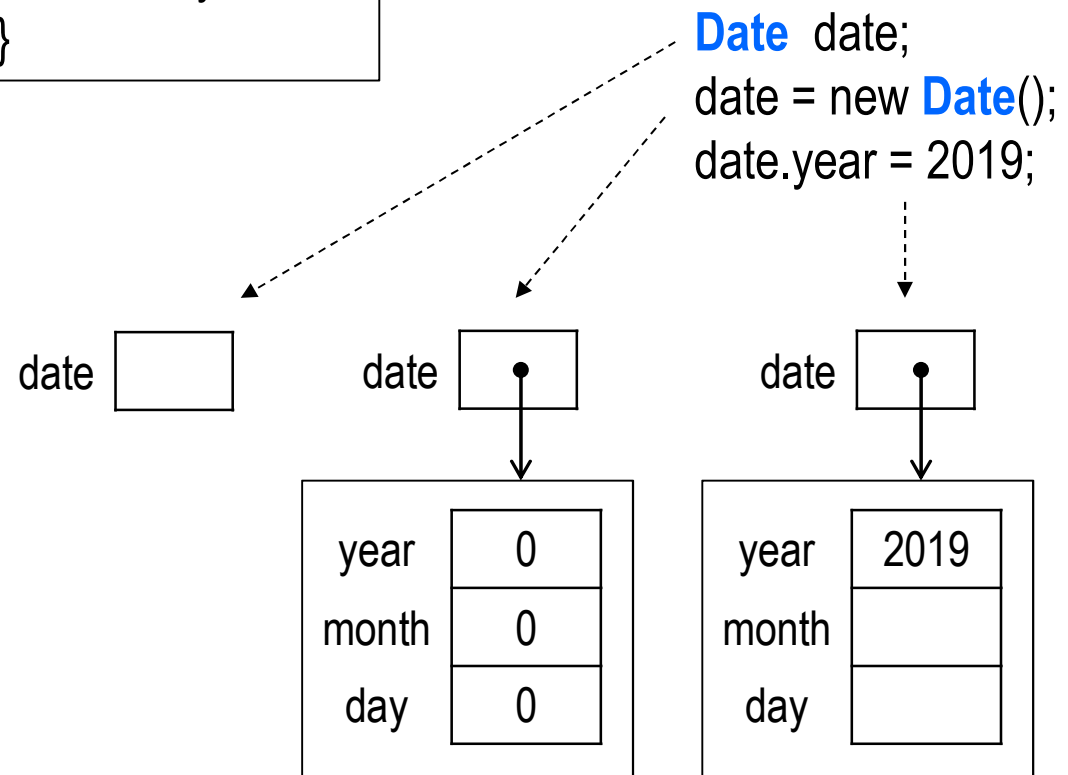
- 변수에 대응하는 메모리 공간에 데이터가 저장됨
- 변수 선언만으로 데이터를 저장할 메모리 공간 할당됨



```
public class Date {  
    int year;  
    int month;  
    int day;  
}
```

참조자료형(클래스자료형)

- 참조 변수에 대응하는 메모리 공간에 객체의 참조 값이 저장됨
- 변수 선언만으로 객체 공간 생성되지 않음. 별도 객체 생성 연산 수행 필요



생성자(constructor), 기본생성자(default constructor)

✚ 생성자(constructor)

- 클래스와 같은 이름의 메소드(함수)
- new 연산자를 통한 객체 생성 시 호출됨
- 클래스는 그 내부에 최소 하나 이상의 생성자 정의문 필요

✚ 기본생성자(default constructor)

- 생성자 중에서 함수 파라미터와 리턴 타입이 없는 생성자
- 클래스 내에 정의된 생성자 없으면 컴파일러에 의해 자동 생성됨
- 클래스 내에 정의된 생성자 있는 경우 기본생성자는 자동 생성되지 않음

```
public class Date {  
    int year;  
    int month;  
    int day;  
}
```

- 클래스 Date 내에 정의된 생성자 없으므로 Date()라는 기본생성자가 컴파일러에 의해 자동 생성됨

```
Date date = new Date();
```

- new 연산자에 의해 Date 객체 생성 시 클래스 Date의 **기본생성자 Date()** 호출

자동 생성된 기본생성자(default constructor) 예시

명령프롬프트에서 다음 명령으로 확인
javap Date

```
public class Date {  
    int  year;  
    int  month;  
    int  day;  
}
```

컴파일

```
public class Date {  
    int  year;  
    int  month;  
    int  day;  
    public Date();  
}
```

자동 생성된 기본생성자 Date()

클래스, 객체

```
public class Car {  
    String    id;  
    int       length;  
    double    weight;  
    char      fuelType;  
    boolean   exportOnly;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        Car car;           // Car 참조 변수 선언  
        car = new Car();    // Car 객체 생성  
        car.id="KSC-123";  
        car.length=3850;  
        car.weight=1500;  
        car.fuelType='D';  
        car.exportOnly=false;  
        System.out.println("고유번호="+car.id);  
        System.out.println("길이(mm)="+car.length);  
        System.out.println("중량(kg)="+car.weight);  
        System.out.println("연료유형="+car.fuelType);  
        System.out.println("수출용="+car.exportOnly);  
    }  
}
```

객체 실습 A

- 다음은 Test 클래스의 미완성 코드와 그 실행 결과를 보인 것이다. 아래 코드가 정상 동작하도록 Student 클래스와 Test 클래스를 완성하시오.
 - ✓ Student 클래스의 필드 id, numberOfFinishedSemesters, gender, gpa, foreignerYN의 각각 문자열, 정수, 문자, 실수, 불리언 자료형으로 정의하시오.

```
public class Test {  
    public static void main(String[] args) {  
        Student s;  
  
        System.out.println("학 번="+s.id);  
        System.out.println("이수학기수="+s.numberOfFinishedSemesters);  
        System.out.println("성 별="+s.gender);  
        System.out.println("평 점="+s.gpa);  
        System.out.println("외국인 여부="+s.foreignerYN);  
    }  
}
```

실행결과

```
학 번=KSU-123  
이수학기수=2  
성 별=여  
평 점=3.97  
외국인 여부=true
```


객체 실습 B

- 다음 Test 클래스는 id가 R2D2인 Robot 클래스의 객체를 생성한 후 그 객체의 id를 출력하는 코드이다. 이 Test 클래스의 오류를 수정하시오.

```
public class Robot {  
    String id;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        Robot robot;  
        robot.id="R2D2";  
        System.out.println(robot.id);  
    }  
}
```

객체 실습 C

- 자바 JDK에는 아래 세 클래스들이 이미 정의되어 있다. 이 세 클래스의 객체를 생성하여 각각 참조 변수 random, thread, JFrame에 저장하는 코드를 작성하시오.

클래스	import 문장
Random	import java.util.Random
JFrame	import javax.swing.JFrame
Thread	import java.lang.Thread

필드, 배열 원소, 지역변수 초기값

참 조: <https://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jls-4.12.5>

필드
(field)

```
public class Player {  
    String    id;           // null  
    int       height;      // 0  
    double    record;      // 0.0d  
    boolean   dopingTestPostive; // false  
    char      gender;      // '\u0000'  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        int    score;  
        //System.out.println(score); // 지역변수  
  
        Player player=new Player();  
        System.out.println("id="+player.id);  
        System.out.println("신 장="+player.height);  
        System.out.println("기 록="+player.record);  
        System.out.println("도 핑 테스트 결과="+player.dopingTestPostive);  
        System.out.println("성 별="+player.gender);  
        System.out.printf("%d", (int)player.gender);  
    }  
}
```

필드(field), static 필드, 배열 원소

- 필드, static 필드 및 배열 원소는 객체 생성 시 아래 디폴트 값으로 자동 초기화됨
 - ✓ int → 0
 - ✓ double → 0.0
 - ✓ boolean → false
 - ✓ char → '\u0000'
 - ✓ 참조 변수 → null

지역변수

- 지역변수는 자동 초기화되지 않음
- 비초기화 지역변수는 사용 시 오류 발생

- field:** 필드, 객체 변수, instance variable
- static field:** 클래스 변수, class variable

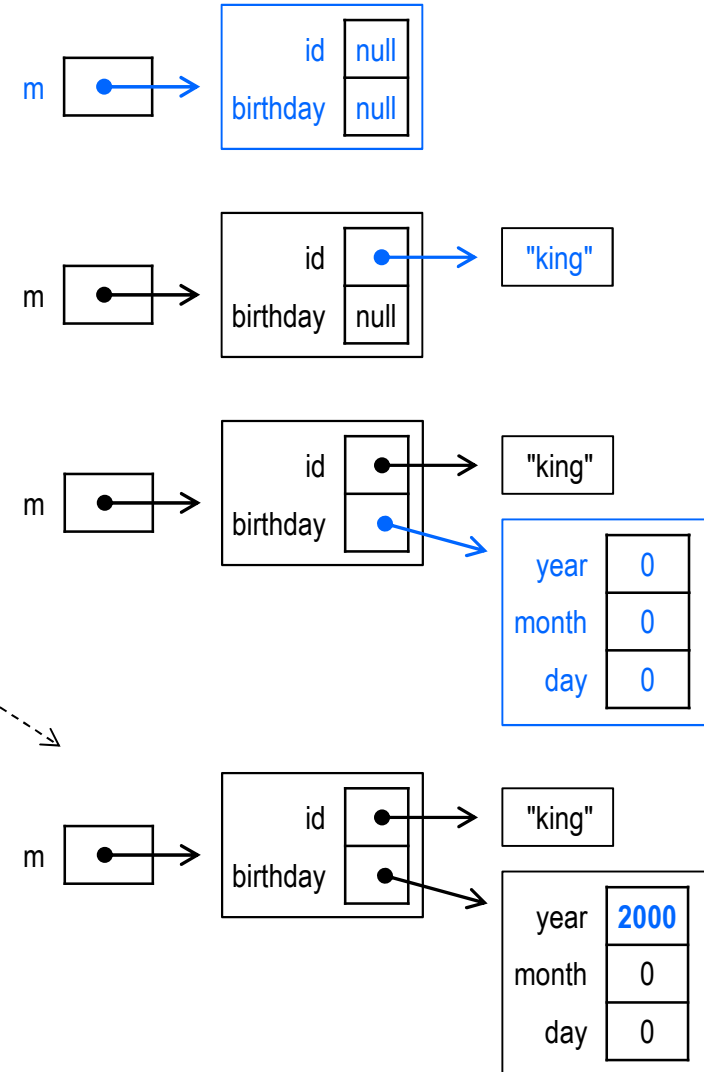
```
public class Test {  
    public static void main(String[] args) {  
        int n[]=new int[3]; // 배열 원소 초기화  
        System.out.println(n[0]); // 0  
    }  
}
```

클래스, 객체

```
public class Member {  
    String id;  
    YMD birthday;  
}
```

```
public class YMD {  
    int year;  
    int month;  
    int day;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        Member m=new Member();  
        m.id = "king";  
        m.birthday=new YMD();  
        m.birthday.year=2000;  
        m.birthday.month=12;  
        m.birthday.day=31;  
        System.out.println("회원아이디=" + m.id);  
        System.out.println("회원출생년=" + m.birthday.year);  
        System.out.println("회원출생월=" + m.birthday.month);  
        System.out.println("회원출생일=" + m.birthday.day);  
    }  
}
```



객체 vs. 객체 참조 변수

```
public class YMD {  
    int    year;  
    int    month;  
    int    day;  
}
```

date2 = date1;

- date1에 저장된 객체참조값을 date2에 대입 저장

```
public class Test {  
    public static void main(String[] args) {  
        YMD date1=null;  
        date1=new YMD();  
        date1.year=2020;  
        date1.month=4;  
        date1.day=1;
```

```
        YMD date2=null;
```

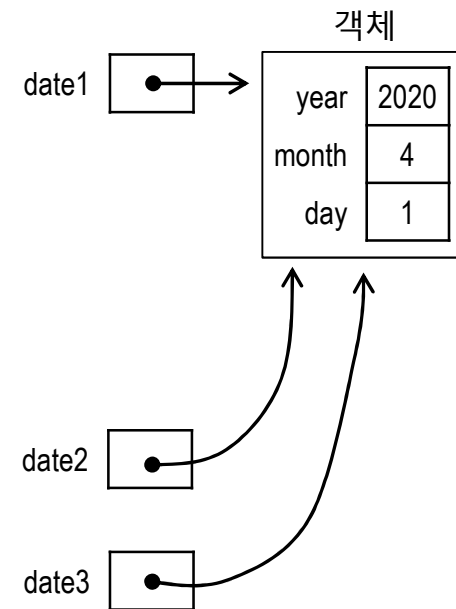
```
        date2=date1;
```

```
        YMD date3=date1;
```

```
        System.out.println(date1.year+"년"+date1.month+"월"+date1.day+"일");  
        System.out.println(date2.year+"년"+date2.month+"월"+date2.day+"일");  
        System.out.println(date3.year+"년"+date3.month+"월"+date3.day+"일");
```

```
    }  
}
```

객체참조변수들
date1, date2, date3



객체 vs. 객체 참조 변수

```
public class YMD {  
    int    year;  
    int    month;  
    int    day;  
}
```

```
public class Test {  
    public static void main(String[] args) {
```

```
        YMD date1=null;  
        date1=new YMD();  
        date1.year=2020;  
        date1.month=4;  
        date1.day=1;
```

```
        YMD date2=null;  
        date2=date1;
```

```
        YMD date3=date1;
```

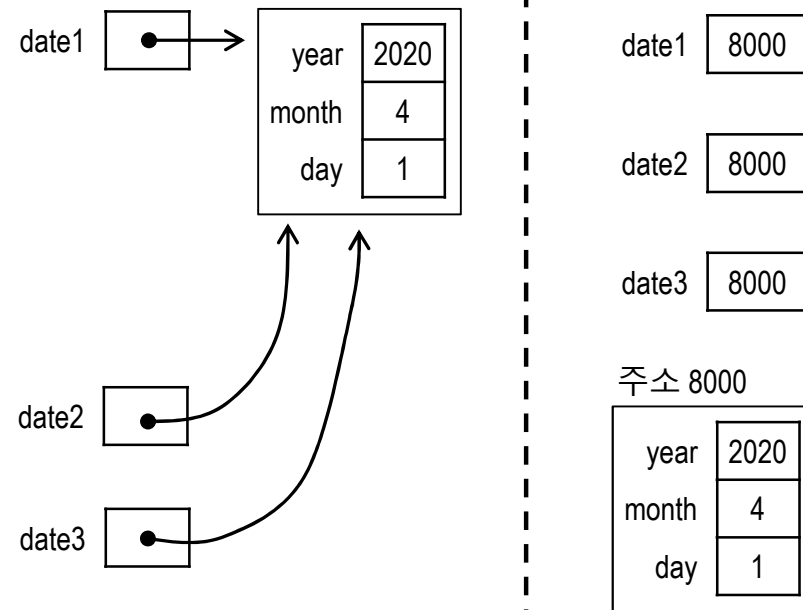
```
        System.out.println(date1.year+"년"+date1.month+"월"+date1.day+"일");  
        System.out.println(date2.year+"년"+date2.month+"월"+date2.day+"일");  
        System.out.println(date3.year+"년"+date3.month+"월"+date3.day+"일");
```

```
    }  
}
```

date2 = date1;

- date1에 저장된
 객체참조값(예: 8000)을
 date2에 대입 저장

객체참조변수에는 객체에 대한
참조값(주소값)이 저장됨



객체 실습 D

- 날짜 클래스 DateInfo를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): year(연도,int), month(월,int), day(일,int)
- 직원 클래스 Employee를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): id(직원아이디,String), joinDate(직원입사일,DateInfo)
- 다음은 클래스 Employee의 객체를 생성한 후 그 정보를 출력하는 코드와 그 실행 결과를 보인 것이다.
.아래 코드의 빈 곳을 완성하시오

```
Public class Test {  
    public static void main(String[] args) {  
        Employee e;  
  
        System.out.println("직원 아이디: "+e.id);  
        System.out.println("직원 입사일: "+e.joinDate.year+"년"+e.joinDate.month+"월"+e.joinDate.day+"일");  
    }  
}
```

실행결과

직원 아이디: EMP-123
직원 입사일: 2015년3월2일

객체 실습 E

- 이름 클래스 Name을 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): firstName(성 제외 이름,String), lastName(성,String)
- 사람 클래스 Person을 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): name(이름,Name)
- 다음은 클래스 Person의 객체를 생성한 후 그 정보를 출력하는 코드와 그 실행 결과를 보인 것이다.
아래 코드의 빈 곳을 완성하시오

```
public class Test {  
    public static void main(String[] args) {  
        Person    p;  
          
        System.out.println("이름: "+p.name.firstName+" "+p.name.lastName);  
    }  
}
```

실행결과

객체 실습 F

- 점수 클래스 Score를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): langScore(국어점수,double), mathScore(수학점수,double), engScore(영어점수,double)
- 학생 클래스 Student를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): id(학번,String), score(점수,Score)
- 다음은 클래스 Student의 객체를 생성한 후 그 정보를 출력하는 코드와 그 실행 결과를 보인 것이다.
아래 코드의 빈 곳을 완성하시오

```
public class Test {  
    public static void main(String[] args) {  
        Student  s;  
  
        System.out.println("학 번: "+s.id);  
        System.out.println("국어점수: "+s.score.langScore);  
        System.out.println("영어점수: "+s.score.engScore);  
        System.out.println("수학점수: "+s.score.mathScore);  
    }  
}
```

실행결과

```
학번: S-001  
국어점수: 100.0  
영어점수: 80.0  
수학점수: 90.0
```

객체 실습 G

- 날짜 클래스 DateInfo를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): year(연도,int), month(월,int), day(일,int)
- 장소 클래스 LocationInfo를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): country(국가,String), city(도시,String)
- 테스트결과정보 클래스 TestInfo를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): score(점수,double), date(테스트날짜,DateInfo), location(테스트장소,LocationInfo)
- 지원자 클래스 Applicant를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): seqNo(지원자순번,int), testInfo(테스트결과정보,TestInfo)
- 다음은 클래스 Applicant의 객체를 생성한 후 그 정보를 출력하는 코드와 그 실행 결과를 보인 것이다. 아래 코드의 빈 곳을 완성하시오

```
public class Test {  
    public static void main(String[] args) {  
        Applicant  a;  
  
  
        System.out.println("지 원자 순 번: "+a.seqNo);  
        System.out.println("테 스트 점 수: "+a.testInfo.score);  
        System.out.println("테 스트 장 소: "+a.testInfo.location.country+" "+a.testInfo.location.city);  
        System.out.println("테 스트 날 짜: "+a.testInfo.date.year+"년"+a.testInfo.date.month+"월"+a.testInfo.date.day+"일");  
    }  
}
```

실행결과

지원자 순번: 89
테스트 점수: 87.5
테스트 장소: 미국 뉴욕
테스트 날짜: 2020년3월25일

기본자료형 배열 vs. 참조자료형 배열

```
public class YMD {  
    int    year;  
    int    month;  
    int    day;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        int scores[]=new int[2];  
        scores[0]=100;  
        scores[1]=90;
```

```
        YMD dates[]=new YMD[2];  
        dates[0]=new YMD();  
        dates[0].year=2020;  
        dates[0].month=3;  
        dates[0].day=25;
```

```
        dates[1]=new YMD();  
        dates[1].year=2020;  
        dates[1].month=4;  
        dates[1].day=1;
```

```
        System.out.println(scores[0]+" "+scores[1]);  
        System.out.println(dates[0].year+" "+dates[0].month+" "+dates[0].day);  
        System.out.println(dates[1].year+" "+dates[1].month+" "+dates[1].day);
```

```
    }  
}
```

```
int scores[] = new int[2];
```

- 크기 2의 기본자료형 int 배열 선언
- scores 배열에는 int 값 저장됨

```
YMD dates[] = new YMD[2];
```

- 크기 2의 참조자료형 YMD 배열 선언
- YMD 배열에는 YMD 객체 참조 값 저장됨

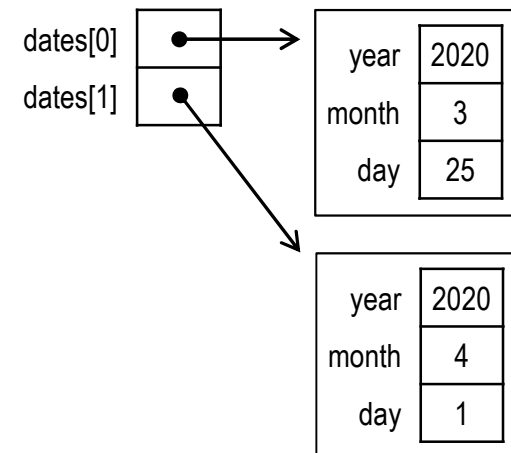
```
scores[0] = 100 ;
```

- 기본자료형 배열에는 배열 원소에 기본자료형 값이 저장됨

```
dates[0] = new YMD() ;
```

- 참조자료형 배열에는 배열 원소에 객체 참조 값이 저장됨

scores[0]	100
scores[1]	90



참조자료형 배열: 반복문 접근

```
public class YMD {  
    int  year;  
    int  month;  
    int  day;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        YMD dates[]=new YMD[2];  
        dates[0]=new YMD();  
        dates[0].year=2020;  
        dates[0].month=3;  
        dates[0].day=25;  
        dates[1]=new YMD();  
        dates[1].year=2020;  
        dates[1].month=4;  
        dates[1].day=1;  
  
        for (int i = 0; i < dates.length; i++) {  
            System.out.println(dates[i].year+"년"+dates[i].month+"월"+dates[i].day+"일");  
        }  
        for (int i = 0; i < dates.length; i++) {  
            YMD v = dates[i];  
            System.out.println(v.year+"년"+v.month+"월"+v.day+"일");  
        }  
        for (YMD ymd : dates) {  
            System.out.println(ymd.year+"년"+ymd.month+"월"+ymd.day+"일");  
        }  
    }  
}
```

실행결과

```
2020년3월25일  
2020년4월1일  
2020년3월25일  
2020년4월1일  
2020년3월25일  
2020년4월1일
```

클래스 정의: 배열 필드

```
public class Test {  
    public static void main(String[] args) {  
        Player p1;  
        p1=new Player();  
        p1.id="P-123";  
        double r1[]={12.1, 12.05, 11.7, 10.9};  
        p1.records=r1;  
  
        Player p2=new Player();  
        p2.id="P-456";  
        double r2[]={13.9, 14.1, 12.8};  
        p2.records=r2;  
  
        Player v[]={p1, p2};  
  
        for (int i = 0; i < v.length; i++) {  
            System.out.print("선수 아이디: "+v[i].id+" , 100m기록(초): ");  
            for (int j = 0; j < v[i].records.length; j++) {  
                System.out.print(v[i].records[j]+" ");  
            }  
            System.out.println();  
        }  
    }  
}
```

```
public class Player {  
    String id;  
    double records[];  
}
```

실행결과

```
선수 아이디: P-123, 100m기록(초): 12.1 12.05 11.7 10.9  
선수 아이디: P-456, 100m기록(초): 13.9 14.1 12.8
```

클래스 정의: 배열 필드

```
public class Test {  
    public static void main(String[] args) {  
        TripInfo t1=new TripInfo();  
        t1.startDate="20201123";  
        t1.endDate="20201130";  
        String p1[]= {"강릉","안동","경주","부산"};  
        t1.placeToVisit=p1;
```

```
  
        TripInfo t2=new TripInfo();  
        t2.startDate="20201201";  
        t2.endDate="20201208";  
        String p2[]= {"런던","파리","로마"};  
        t2.placeToVisit=p2;
```

```
  
        TripInfo v[]={t1, t2};  
        for (int i = 0; i < v.length; i++) {  
            System.out.print("여행 일정:"+v[i].startDate+" ~ "+v[i].endDate+" 방문장소:");  
            for (int j = 0; j < v[i].placeToVisit.length; j++) {  
                if(j>0) System.out.print("-");  
                System.out.print(v[i].placeToVisit[j]);  
            }  
            System.out.println();  
        }  
    }  
}
```

```
public class TripInfo {  
    String startDate;  
    String endDate;  
    String placeToVisit[];  
}
```

실행결과

```
여행일정:20201123 ~ 20201130, 방문장소:강릉-안동-경주-부산  
여행일정:20201201 ~ 20201208, 방문장소:런던-파리-로마
```

객체 실습 H

- ✚ 성적 클래스 TestResult를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): midTermScore(중간점수,double), finalTermScore(기말점수,double)
- ✚ 다음 세 학생의 정보를 각각 TestResult 객체로 만드시오(객체참조변수명 t1, t2, t3 사용)
 - 학생 1 → 중간점수 = 80, 기말점수 = 100
 - 학생 2 → 중간점수 = 90, 기말점수 = 94
 - 학생 3 → 중간점수 = 70, 기말점수 = 80
- ✚ 위 세 객체의 참조값을 크기 3의 배열 v에 저장하시오
- ✚ for문을 사용하여 배열 v 내 각 객체의 정보를 아래와 같이 출력하는 코드를 작성하시오

중간점수: 80.0, 기말점수: 100.0
중간점수: 90.0, 기말점수: 94.0
중간점수: 70.0, 기말점수: 80.0

객체 실습 I

- 날짜 클래스 DateInfo를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): year(연도,int), month(월,int), day(일,int)
- 직원 클래스 Employee를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): id(직원아이디,String), joinDate(직원입사일,DateInfo)
- 다음 두 직원의 정보를 각각 Employee 객체로 만드시오(객체참조변수명 e1, e2 사용)
 - 직원 1 → 직원아이디 = EMP-123, 직원입사일 = 2015년3월2일
 - 직원 2 → 직원아이디 = EMP-456, 직원입사일 = 2016년3월4일
- 위 두 객체의 참조값을 크기 2의 배열 v에 저장하시오
- for문을 사용하여 배열 v 내 각 객체의 정보를 아래와 같이 출력하는 코드를 작성하시오

직원번호: EMP-123, 입사일: 2015년3월2일 직원번호: EMP-456, 입사일: 2016년3월4일
--

객체 실습 J

- ✚ 구매내역 클래스 BuyingHistory를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): date(구매일자,String), itemsBought(구매물품목록,**String 배열**)
- ✚ 고객 클래스 Customer를 다음과 같이 정의하시오
 - 필드변수명(설명,자료형): id(아이디,String), buyingHistory(구매내역, BuyingHistory)
- ✚ 다음 두 고객의 정보를 각각 Customer 객체로 만드시오(객체참조변수명 c1, c2 사용)
 - 고객 1 → 아이디 = lamCoder, 구매일자 = 20191231, 구매물품목록 = 키보드,모니터,마우스
 - 고객 2 → 아이디 = BusanCitizen, 구매일자 = 20191030, 구매물품목록 = 야구공,야구배트
- ✚ 위 두 객체의 참조값을 크기 2의 배열 v에 저장하시오
- ✚ for문을 사용하여 배열 v 내 각 객체의 정보를 아래와 같이 출력하는 코드를 작성하시오

고객 아이디:lamCoder, 구매일자:20191231, 구매물품목록:키보드,모니터,마우스
고객 아이디:BusanCitizen, 구매일자:20191030, 구매물품목록:야구공,야구배트
- ✚ for문을 사용하여 배열 v 내 각 객체의 정보를 아래와 같이 출력하는 코드를 작성하시오

고객 아이디:lamCoder, 구매일자:20191231, 구매물품목록:키보드 등 (3개)
고객 아이디:BusanCitizen, 구매일자:20191030, 구매물품목록:야구공 등 (2개)

객체 vs. 객체 참조 변수

```
public class Student {  
    String name;  
    int score;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        Student s1=null;  
        s1=new Student();  
        s1.name="이영희";  
        s1.score=99;  
        Student s2=null;  
        s2=s1;
```

s1, s2의 값을 출력해
보시오.

```
        Student students[]=new Student[10]; // 비교  
        students[0]=s1;  
        students[1]=new Student();
```

students[0], students[1],
students[2]의 값을
출력해 보시오.

```
        Student sArray[]={s1, students[1], new Student()}; // 비교  
        int n[]={88,99};
```

sArray[0], sArray[1],
sArray[2]의 값을
출력해 보시오.

```
    }  
}
```

객체 vs. 객체 참조 변수

```
public class Student {  
    String name;  
    int score;  
}
```

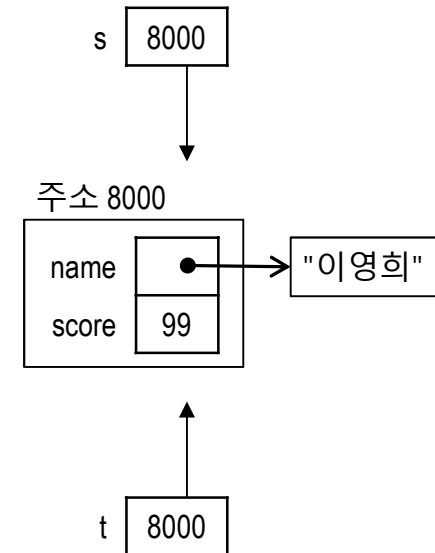
```
public class Test {  
    public static void main(String[] args) {  
        Student s=null;  
        s=new Student();  
        s.name="이영희";  
        s.score=99;  
    }  
}
```

updateStudentScore(s) 호출
직전, 직후에 s의 score를
출력해 보시오.

```
updateStudentScore( s );
```

t = s
• 객체 참조 값이 전달됨

```
private static void updateStudentScore(Student t) {  
    t.score=88;  
}
```



객체 실습 K

```
public class Student {  
    String name;  
    int score;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        Student s=null;  
        s=new Student();  
        s.name="이영희";  
        s.score=99;  
  
        char grade=getGrade(s);  
        System.out.println(grade);  
    }  
}
```

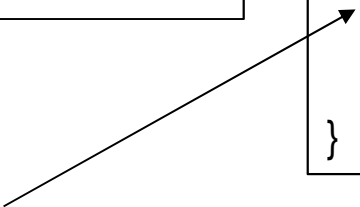
실습

- Student 객체 참조 값을
파라미터로 전달받아 score에
저장된 값을 등급(A,B,C,D,F)로
변환하여 반환하는 메소드
getGrade()를 완성하시오.

객체 실습 L

```
public class Student {  
    String name;  
    int score;  
}
```

```
public class Test {  
    public static void main(String[] args) {  
        Student s;  
        s=createStudent("이영희", 99);  
        System.out.println(s.name+","+s.score);  
    }  
}
```




실습

- 파라미터로 전달받은 학생의 이름과 성적 값에 대응하는 Student 객체를 생성한 후 반환하는 메소드 createStudent()를 완성하시오.

References

 <http://docs.oracle.com/javase/7/docs/api/>

 <https://docs.oracle.com/javase/tutorial/java/>

 김윤명. (2008). 뇌를 자극하는 Java 프로그래밍. 한빛미디어.

 남궁성. 자바의 정석. 도우출판.

 황기태, 김효수 (2015). 명품 Java Programming. 생능출판사.