



java 14강 String, 유용한 클래스들

양 명 속

[now4ever7@gmail.com]



목차

- 유용한 클래스들
 - String 클래스
 - Calendar/ Date 클래스
 - Math 클래스
 - DecimalFormat
 - SimpleDateFormat



String 클래스

- String 클래스
 - 문자열을 위한 클래스
 - 문자열을 저장하고 이를 다루는데 필요한 메서드를 제공함

String substring(int begin)

– 시작위치(begin)부터 끝까지의 문자열을 얻어옴

String 클래스의 주요 메서드

메서드	설명
int indexOf(String str)	주어진 문자열을 첫 부분부터 검색하여 입력한 문자열을 제일 먼저 만나는 위치를 int로 반환. 없으면 -1 을 반환
int lastIndexOf(String str)	문자열을 마지막 부분부터 검색하여 입력한 문자열을 제일 먼저 만나는 위치를 int로 반환
char charAt(int index)	지정된 위치(index)에 있는 문자를 알려줌
boolean endsWith(String suffix)	지정된 문자열(suffix)로 끝나는지 검사함
boolean equals(Object obj)	매개변수로 받은 문자열과 String인스턴스의 문자열을 비교
int length()	문자열의 길이를 알려줌
String replace(char old, char nw)	문자열에서 특정 문자(old)를 원하는 문자(nw)로 바꾼 문자열을 반환
String[] split(String regex)	문자열을 지정된 분리자(regex)로 나누어 문자열 배열에 담아 반환
boolean startsWith(String prefix)	주어진 문자열로 시작하는지 검사
String substring(int begin, int end)	주어진 시작위치(begin)부터 끝 위치(end) 범위에 포함된 문자열을 얻는다. 시작위치의 문자는 범위에 포함, 끝 위치의 문자는 포함되지 않음 (begin <= x < end)
String trim()	앞뒤의 불필요한 공백 문자를 제거. 문자열 중간에 있는 공백은 제거되지 않음
String toLowerCase()	소문자로 변환한 문자열을 리턴
String toUpperCase()	대문자로 변환한 문자열을 리턴
Static String valueOf(int i)	지정된 값을 문자열로 변환하여 반환. 참조변수의 경우 toString()을 호출한 결과를 반환함

- `indexOf()` - 특정 문자가 있는 위치를 찾아내는 메서드
=> 그 위치부터 어떤 값을 추출해내거나, 그 값이 존재하는지를 확인할 때 사용
- `substring()` - String 값의 일부를 추출할 때 사용

```
ch = 0
앞 Java의 위치:6
뒤 Java의 위치:15
str.substring(6, 10):Java
.com 사이트 입니다.
www로 시작!
```

```
public class StringTest {
    public static void main(String[] args){
        //          0123456789012345678901
        String str = "Hello Java, Hi Java!";
        char ch = str.charAt(4);
        System.out.println("ch = " + ch);

        int idx = str.indexOf("Java");
        System.out.println("앞 Java의 위치:" + idx);
        System.out.println("뒤 Java의 위치:" +
                           str.lastIndexOf("Java"));
        String sub = str.substring(6, 10);
        System.out.println("str.substring(6, 10):" + sub);

        String str1 = "www.nate.com";
        if(str1.endsWith(".com")){
            System.out.println(".com 사이트 입니다.");
        }
        if(str1.startsWith("www")){
            System.out.println("www로 시작!");
        }
    }
}
```



예제

```
이메일 규칙이 올바릅니다.  
gmail, com  
Java Android!!  
s2 변수의 길이 :14  
1020  
1020
```

```
String email = "hong@gmail.com";  
if(email.indexOf("@") != -1 && email.indexOf(".") != -1  
    && email.indexOf("@") < email.indexOf(".")){  
    System.out.println("이메일 규칙이 올바릅니다.");  
}  
else{  
    System.out.println("이메일 규칙이 올바르지 않습니다.");  
}
```

```
String emailSub = email.substring(email.indexOf("@") + 1, email.indexOf(".")); //gmail  
String emailSub2 = email.substring(email.indexOf(".") + 1); //com  
System.out.println(emailSub + ", " + emailSub2);
```

```
String s = "    Java Android!!    ";  
String s2 = s.trim();  
System.out.println(s2);  
System.out.println("s2 변수의 길이 :"+s2.length());
```

```
int x = 10;  
int y = 20;  
System.out.println(String.valueOf(x) + y);  
System.out.println(Integer.toString(x) + y);
```

```
}//main
```

```
}
```



split()

- 특정 문자들을 경계로 하여 문자열을 여러 개의 토막으로 분리
- 공백, 마침표, 쉼표 등을 구분자로 하여 단어 단위로 문자열을 잘라낼 수 있음
- 인수로 구분자 문자 배열을 전달하면 잘라낸 토막 문자열들을 배열 형태로 리턴함



예제 - split()

```
public class StringTest2 {
    public static void main(String[] args){
        //      012345678901234567890123
        String str = "123456789";
        String s = str.replace('7', '칠');
        //String s = str.replace("7", "칠"); //가능

        System.out.println("str.replace('7', '칠'):" + s);

        String str2 = "smile.lucky.joy.happy";
        String[] word = str2.split("WW.");

        for(String w : word)
        {
            System.out.println(w);
        }
        System.out.println("-----");

        str2 = "smile,lucky,joy,happy";
        String[] word2 = str2.split(",", 3);
        for(int i=0;i<word2.length;i++)
        {
            System.out.println(word2[i]);
        }
    }
} //main
```

```
str.replace('7', '칠'):123456칠89
smile
lucky
joy
happy
-----
smile
lucky
joy,happy
```


실습1

```
파일명:test, 확장자:txt  
안녕하세요<br> 저는 홍길동입니다.  
폴더:c:\shop\upload  
파일명:test  
확장자:txt
```

- "test.txt" 의 파일명과 확장자를 분리하여 화면 출력(파일명이 변경되더라도 처리되도록, abc123.html)
 - substring(), lastIndexOf() 메서드 이용
- "안녕하세요WrWn 저는 홍길동입니다." 의 WrWn 을
 로 치환하여 화면 출력
 - replace()메서드 사용
- "c:\shop\upload\test.txt" 을 폴더, 파일명, 확장자를 구분하여 화면 출력
 - lastIndexOf(), substring() 메서드 이용
- 사용자로부터 url 주소를 입력 받아서 입력한 url 주소에 <http://www.mall.com>가 존재하면 입력 받은 주소를 화면출력하고, 존재하지 않으면 "url 주소가 적합하지 않다"고 화면 출력
 - indexOf()메서드 이용

```
URL 주소를 입력하세요  
http://naver.com  
url 주소가 적합하지 않습니다
```

```
URL 주소를 입력하세요  
http://www.mall.com/content.aspx  
해당 url 주소:http://www.mall.com/content.aspx
```



실습1

- 주민등록번호를 입력 받아서 "1990년 10월 11일, 남" 의 형태로 화면 출력
 - substring() 이용

주민등록번호를 입력하세요<하이픈없이 입력>
9901072223333
1999년 01월 07일 , 여

주민등록번호를 입력하세요<하이픈없이 입력>
0812303335555
2008년 12월 30일 , 남

- 성별 구하기
 - 주민번호 7번째 자리(index=6)가 1이나 3이면 '남', 2나 4 이면 '여'
- 출생년도 구하기
 - 주민번호 7번째 자리가 1 이나 2이면 1900년대 출생
 - 3 이나 4 이면 2000년대 출생

Calendar와 Date 클래스

■ Calendar 와 Date

- 날짜와 시간에 관련된 데이터를 쉽게 처리할 수 있도록 Calendar와 Date 클래스를 제공
- JDK 1.0 부터 Date 사용
- JDK 1.1 부터 보다 향상된 기능의 Calendar 가 추가

■ java.util.Date/ java.util.Calendar

```
Calendar cal = new Calendar(); //에러
```

※ Calendar 클래스의 객체 생성 방법

```
[1] Calendar cal = Calendar.getInstance();
```

getInstance()메서드는 Calendar 클래스를 구현한 클래스의 인스턴스를 반환함

```
[2] Calendar cal = new GregorianCalendar();
```

```
[3] GregorianCalendar cal = new GregorianCalendar(); //양력을 나타내는 클래스  
GregorianCalendar - Calendar 클래스의 자식 클래스
```

- **Calendar** 는 추상 클래스이기 때문에 직접 객체를 생성할 수 없고, 메서드를 통해서 완전히 구현된 클래스의 인스턴스를 얻어야 함

```

import java.util.*;
class DateTest1 {
    public static void main(String[] args) {
        Date date = new Date(); //현재 날짜
        System.out.println("현재 날짜:"+date);
        System.out.println("현재 날짜:"
            +date.toLocaleString() );

        int year = date.getYear();
        int month = date.getMonth(); //0~11
        int day = date.getDate();
        int hours = date.getHours();
        int minutes = date.getMinutes();
        int seconds = date.getSeconds();
        System.out.println("year : " + (year+1900));
        System.out.println("month : " + (month+1)); // month+1 해야 함
        System.out.println("day : " + day);
        System.out.println("hours : " + hours);
        System.out.println("minutes : " + minutes);
        System.out.println("seconds : " + seconds);
        System.out.println("요일 : " + date.getDay()); //0:일요일

        //현재의 시간을 1970년 1월1일 0시 0분 0초를 기준으로 밀리초 단위로 환산
        long diff = date.getTime()/1000; //초
        System.out.println("1970-01-01부터 오늘까지 경과된 시간(초) : " + diff+"초");

        diff = diff/(24*60*60); //일
        System.out.println("1970-01-01부터 오늘까지 경과된 일 수 : " + diff +"일");
    }
}

```

```

현재 날짜:Tue Sep 02 21:40:03 KST 2014
현재 날짜:2014. 9. 2 오후 9:40:03
year : 2014
month : 9
day : 2
hours : 21
minutes : 40
seconds : 3
요일 : 2
1970-01-01부터 오늘까지 경과된 시간(초) : 1409661603 초
1970-01-01부터 오늘까지 경과된 일 수 : 16315 일

```

예제

```
import java.util.*;

public class CalendarTest {
    public static void main(String[] args) {
        Calendar ca = Calendar.getInstance();
        int x = ca.get(Calendar.DAY_OF_YEAR);
        System.out.println("오늘은 일년중 " + x + "번째 날입니다.");

        int year = ca.get(Calendar.YEAR);
        int month = ca.get(Calendar.MONTH);
        //int day = ca.get(Calendar.DAY_OF_MONTH);
        int day2 = ca.get(Calendar.DATE);
        int hour = ca.get(Calendar.HOUR_OF_DAY);
        int minute = ca.get(Calendar.MINUTE);
        int second = ca.get(Calendar.SECOND);
        int weekday = ca.get(Calendar.DAY_OF_WEEK);

        System.out.println("year : " + year);
        System.out.println("month(0~11) : " + month); //month+1 해야 함
        System.out.println("day : " + day2);
        System.out.println("hour(0~23) : " + hour);
        System.out.println("minute(0~59) : " + minute);
        System.out.println("second(0~59) : " + second);
        System.out.println("요일(1~7, 1:일요일) : " + weekday);

        System.out.println("ca : " + ca.getTime());
    }
}
```

```
오늘은 일년중 151번째 날입니다.
year : 2012
month(0~11) : 4
day : 30
hour(0~23) : 18
minute(0~59) : 5
second(0~59) : 32
요일(1~7, 1:일요일) : 4
ca : Wed May 30 18:05:32 KST 2012
```

```
public final Date getTime()
- Returns a Date object
representing this Calendar's
time value
```

예제2

```
date1 : 2012년 1월 1일, 일요일  
date2 : 2012년 5월 30일, 수요일  
date1과 date2의 차이<초> : 12960000초  
date1과 date2의 차이<일> : 150일
```

```
import java.util.*;  
class CalendarEx2{  
    public static void main(String[] args){  
        // 0 1 2 3 4 5 6 7  
        final String[] WEEK = {"", "일", "월", "화", "수", "목", "금", "토"};  
  
        Calendar date1 = Calendar.getInstance();  
        Calendar date2 = Calendar.getInstance();  
  
        // date1의 날짜를 2012-01-01로 설정한다.  
        date1.set(2012, 0, 1);  
  
        System.out.print("date1 :"+showDate(date1) + ",");  
        System.out.println(WEEK[date1.get(Calendar.DAY_OF_WEEK)] + "요일");  
        System.out.print("date2 :"+showDate(date2) + ",");  
        System.out.println(WEEK[date2.get(Calendar.DAY_OF_WEEK)] + "요일");  
  
        //두 날짜간의 차이 - getTimeInMillis() 를 이용해서 천분의 일초 단위로 변환해야 함  
        long difference =(date2.getTimeInMillis() - date1.getTimeInMillis())/1000; //초  
        System.out.println("date1과 date2의 차이(초) : "+ difference +"초");  
        System.out.println("date1과 date2의 차이(일) : "+ difference/(24*60*60) +"일");  
    }  
}
```

getTimeInMillis() - 현재의 시간을 1970년 1월1일 0시 0분 0초를 기준으로 밀리초 단위로 환산



예제2

```
public static String showDate(Calendar cal)
{
    String res=cal.get(Calendar.YEAR) + "년"
        + (cal.get(Calendar.MONTH)+1) + "월"
        +cal.get(Calendar.DATE) + "일";

    return res;
}
```

GregorianCalendar()
GregorianCalendar(int year, int month, int dayOfMonth)

예제3

```
import java.util.*;
class CalendarEx4 {
    public static void main(String[] args) {
        //Calendar date = Calendar.getInstance();
        //date.set(2012, 4, 30);          // 2012년 5월 30일
        Calendar date = new GregorianCalendar(2012, 4, 30);
        System.out.println(showDate(date));
        System.out.println("----- 7일 후 -----");
        date.add(Calendar.DATE, 7);
        System.out.println(showDate(date));

        System.out.println("----- 6달 전 -----");
        date.add(Calendar.MONTH, -6);
        System.out.println(showDate(date));

        System.out.println("----- 15년 전 -----");
        date.add(Calendar.YEAR, -15);
        System.out.println(showDate(date));
    }
    public static String showDate(Calendar date) {
        return date.get(Calendar.YEAR)+"년 "
            + (date.get(Calendar.MONTH)+1) + "월 "
            + date.get(Calendar.DATE) + "일";
    }
}
```

public abstract void add(int field, int amount)

```
2012년 5월 30일
----- 7일 후 -----
2012년 6월 6일
----- 6달 전 -----
2011년 12월 6일
----- 15년 전 -----
1996년 12월 6일
```




Calendar와 Date간의 변환

1. Calendar를 Date로 변환

```
Calendar cal = Calendar.getInstance();  
Date date = new Date(cal.getTimeInMillis()); //Date(long date)  
또는  
Date date = cal.getTime();
```

2. Date를 Calendar로 변환

```
Date d = new Date();  
Calendar cal = Calendar.getInstance();  
cal.setTime(d);
```

```
public Date(long date)
```

date - the milliseconds since January 1, 1970, 00:00:00 GMT.

```
public final void setTime(Date date)
```



실습

생일(1990-07-15) 이후 경과일수: 8815일
수료일까지 남은 일수: 112일
7/28 부터 100일 후 : 2014년 11월 5일

- 생년월일부터 오늘까지 몇 일이 경과했는지 구하기
 - 두 날짜간의 차이 - getTimeInMillis() 를 이용
- 수료일(2020-02-13)까지 며칠 남았는지 구하기
 - 두 날짜간의 차이 - getTimeInMillis() 를 이용
- 오늘부터 100일 후의 날짜 구하기
 - 기준일로부터 며칠 후, 며칠 전 구하기 - add() 이용



형식화 클래스

■ DecimalFormat

- 형식화 클래스 중에서 숫자를 형식화하는데 사용됨
- 숫자 데이터를 정수, 부동소수점, 금액 등의 다양한 형식으로 표현 가능
- 반대로 일정한 형식의 텍스트 데이터를 숫자로 쉽게 변환하는 것도 가능



예제

```
public final String format(double number)
숫자 => 형식화된 문자열
```

```
import java.text.*;
class PrintTest{
    public static void main(String[] args) {
        DecimalFormat df = new DecimalFormat("#,###.##");
        String s2 = df.format(1234567.89512);
        System.out.println("Wn" +s2); // 결과 1,234,567.9

        DecimalFormat df2 = new DecimalFormat("#,###.00");
        s2 = df2.format(1234567.89512);
        System.out.println("Wn" +s2); // 결과 1,234,567.90

        DecimalFormat df3 = new DecimalFormat("#,##0");
        s2 = df3.format(123456.723456);
        System.out.println("Wn" +s2); // 결과 123,457

    }
}
```



형식화 클래스

```
public final String format(Date date)  
날짜 => 형식화된 문자열
```

■ SimpleDateFormat

- 날짜 데이터를 원하는 형태로 다양하게 출력할 수 있다
- Date 인스턴스만 format 메서드에 사용될 수 있다

```
Date date = new Date();  
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  
  
String result = sdf.format(date);
```

- SimpleDateFormat의 parse()는 문자열을 날짜로 변환해 줌

```
public Date parse(String source) throws ParseException
```

예제

```
2012-05-30
12-05-30 수요일
2012-05-30 18:53:04.390
2012-05-30 06:53:04 오후
```

```
import java.util.*;
import java.text.*;
```

H Hour in day (0-23)
h Hour in am/pm (1-12)

```
class DateFormatEx2{
    public static void main(String[] args) {
        // Calendar와 Date간의 변환
        Calendar cal = Calendar.getInstance();
        cal.set(2012, 4, 30); // 2012년 5월 30일 - Month는 0~11의 범위
```

```
        Date date = cal.getTime();
```

```
        SimpleDateFormat sdf1, sdf2, sdf3, sdf4;
        sdf1 = new SimpleDateFormat("yyyy-MM-dd");
        sdf2 = new SimpleDateFormat("yy-MM-dd E요일");
        sdf3 = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss.SSS");
        sdf4 = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss a");
```

```
        String str = sdf1.format(date) // format(Date d)
        System.out.println(str);
        System.out.println(sdf2.format(date));
        System.out.println(sdf3.format(date));
        System.out.println(sdf4.format(date));
```

```
    }
}
```

날짜를 입력하세요<2013-09-30>
2014-03-03
두 날짜 사이의 간격 : 80

예제-parse()

```
import java.util.*;
import java.text.*;
class DateFormatTest5{
```

사용자가 입력한 날짜와 오늘날짜 사이의 간격 구하기
=> 사용자가 입력한 문자열형식의 날짜를 **Date**로 변환해야 함

```
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("오늘 이전 날짜를 입력하세요(2013-09-30)");
        String str = sc.nextLine();
        DateFormat df = new SimpleDateFormat("yyyy-MM-dd");
        try {
            Date date = df.parse(str);
            Date today=new Date();
            long diff = (today.getTime()-date.getTime())/1000;
            diff = diff/(24*60*60);
            System.out.println("두 날짜 사이의 간격 : "+diff);

        } catch(ParseException e) {
            e.printStackTrace();
        }
    }
}
```

```
public Date parse(String source)
    throws ParseException
```

예제

사용자에게 날짜(2013-09-30)를 입력 받아서
2013년 09월 30일 형태로 출력하기

```
import java.util.*;
import java.text.*;
class Ex04_DateFormat{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("날짜를 입력하세요(2013-09-30)");
        String str = sc.nextLine();

        DateFormat df = new SimpleDateFormat("yyyy-MM-dd");
        DateFormat df2 = new SimpleDateFormat("yyyy년 MM월 dd일");

        try {
            Date date = df.parse(str);

            System.out.println(df2.format(date));
        } catch(ParseException e) {
            e.printStackTrace();
        }
    }
}
```

날짜를 입력하세요<2013-09-30>
2013-10-08
2013년 10월 08일



실습

- 사용자로부터 정수를 입력 받아서 천단위 구분기호가 표시되도록 출력하기

- 사용자에게 날짜를 입력 받아서 "2013년 09월 30일 월" 형태로 출력하시오.
 - SimpleDateFormat, parse()메서드 이용

```
날짜를 입력하세요<2013/09/30>  
2013/10/08  
2013년 10월 08일 화
```

- 오늘 날짜를 아래 형태대로 출력하시오
 - SimpleDateFormat 이용

```
오늘날짜 : 2012-05-30 수요일 09:33:16 오후
```



Math

- 수학 함수들을 제공하는 클래스
- 객체를 만들기 위한 클래스라기보다는 수학함수들을 하나의 범주에 묶어 놓기 위한 것
- 모든 계산 메서드는 static 이므로 `Math.abs()` 식으로 클래스명으로 바로 호출
- 두 개의 상수
 - `Math.E` - 자연로그의 밑, 2.71828182
 - `Math.PI` - 원주율, 3.14159265



Math 메서드 목록

메서드	설 명
abs()	절대 값을 계산함
ceil()	올림. 크거나 같은 최소의 정수, 수직선상의 바로 오른쪽 정수를 구함
floor()	내림. 작거나 같은 최대의 정수, 수직선상의 바로 왼쪽 정수를 구함
max()	두 수 중 큰 수를 선택
min()	두 수 중 작은 수
round()	반올림. 소수점 첫째 자리에서 반올림한 정수값을 반환함
random()	0.0 ~ 1.0 범위의 임의의 double 값을 반환함 $0.0 \leq x < 1.0$



예제

```
class MathTest
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        System.out.println("abs(-27.5) : "+ Math.abs(-27.5));
```

```
        System.out.println("abs(27.5) : "+ Math.abs(27.5));
```

```
        System.out.println("ceil(-27.3):"+ Math.ceil(-27.3));
```

```
        System.out.println("ceil(27.3):"+ Math.ceil(27.3));
```

```
        System.out.println("floor(-27.6):"+ Math.floor(-27.6));
```

```
        System.out.println("floor(27.6):"+ Math.floor(27.6));
```

```
        System.out.println("round(34.5374):"+ Math.round(34.5374)); //반올림(소수1째 자리에서
```

```
    )
```

```
        System.out.println("Math.E: "+ Math.E);
```

```
        System.out.println("Math.PI: "+ Math.PI );
```

```
    }
```

```
}
```

```
abs<-27.5> : 27.5
abs<27.5> : 27.5
ceil<-27.3>:-27.0
ceil<27.3>:28.0
floor<-27.6>:-28.0
floor<27.6>:27.0
round<34.5374, 2>:35
Math.E: 2.718281828459045
Math.PI: 3.141592653589793
```

예제

```
class RandomTest
{
    public static void main(String[] args)
    {
        int result;
        System.out.println("1~50 사이의 난수");
        for (int i=0;i<10 ;i++ )
        {
            result = (int)(Math.random()*50 + 1);
            System.out.print(result + "\t");
        }
    }
}
```



실습 - random() 메서드 이용

- 같은 번호가 두 번씩 들어간 배열 만들기
 - 짝수를 입력 받는다.
 - 입력 받은 짝수의 반을 구한다
 - 12를 입력 받았으면 $12/2 \Rightarrow 6$
 - 짝수의 반이 6 이라면 0~5 사이의 임의의 수를 얻는다 (random 메서드 이용)
 - 얻은 수가 배열에 2개 미만인지(0개나 1개) 확인하여 2개 미만이면 얻은 수를 배열에 대입하고, 2개 이상이면 다시 0~5 사이의 새로운 수를 얻는다.
 - 같은 수가 두 개씩 임의의 위치에 들어가도록

실습-random() 메서드 이용

```
배열의 개수를 입력하세요<짝수로>
12
0 1 0 2 4 1 2 3 4 3 5 5
Quit?(Y/N)
n
배열의 개수를 입력하세요<짝수로>
6
0 1 2 1 2 0
Quit?(Y/N)
n
배열의 개수를 입력하세요<짝수로>
4
0 0 1 1
Quit?(Y/N)
n
배열의 개수를 입력하세요<짝수로>
2
0 0
Quit?(Y/N)
y
```

실습

```
시간을 입력하세요<시, 분, 초>  
12  
30  
45  
----1시간 1분 1초 후-----  
13 : 31 : 46
```

■ Time 클래스의 멤버로

- 시간(hours)을 배열로 만들고, 시, 분, 초가 들어가도록

- 멤버변수 - hours[3] 배열
- 생성자에서 배열 초기화
- getter/setter
- 메서드 - incrementHour()

```
hours[0]:시  
hours[1]:분  
hours[2]:초
```

- 배열의 시, 분, 초를 모두 1증가시키는 메서드

- main()에서 사용자로 부터 시, 분, 초를 입력 받아 시간 배열에 넣고, 클래스 객체 생성 후 시분초를 증가시키는 메서드 호출 후 화면출력



과제



과제- 은행계좌 관리 프로그램

- 은행계좌 관리 프로그램
 - [1] 계좌 개설
 - [2] 입금
 - [3] 출금
 - [4] 계좌정보 전체 출력(전체 고객 잔액 조회)
- 프로그램을 간결하게 하기 위한 가정
 - ID(통장번호)는 중복되지 아니한다
 - 계좌 개설시 중복되는 계좌번호의 입력이 없을 것이라는 가정
 - 입금 및 출금액은 무조건 0보다 크다
 - 고객의 계좌에 대한 정보는 계좌번호, 고객이름, 고객의 잔액, 3가지만 저장 및 관리한다
- 저장의 형태는 ArrayList를 이용한다
- 출금시 출금하려는 금액이 잔액보다 크면 "잔액부족" 이라고 출력해주고, 출금처리는 하지 않는다



과제 -은행계좌 관리 프로그램

- Account 클래스(부모 클래스)
 - 멤버변수
 - 계좌번호, 잔액, 고객이름
 - 생성자, getter/setter
 - 메서드
 - 입금하다
 - 입금액을 잔액에 더한다
 - 출금하다 - 출금하려는 금액이 잔액보다 크면 0을 리턴
 - 정상적인 경우에는 출금처리
 - 계좌정보를 출력하는 메서드
 - 계좌번호, 잔액, 고객이름 출력



과제- 은행계좌 관리 프로그램

- NormalAccount 클래스(Account의 자식)
- 보통예금 계좌
 - 멤버변수
 - 이자율 (%단위, 예:3% => 3)
 - 생성자
 - 메서드
 - 입금하다 오버라이딩
 - 원금과 이자율에 따른 이자도 잔액에 더한다
 - 예) 잔액이 200000원, 입금액이 10000원이면
 - 잔액에 입금액 10000원도 더하고,
 - 이자율 3%에 대한 이자 300원도 더해서
 - 잔액은 210,300원이 된다
 - 계좌정보를 출력하는 메서드 오버라이딩
 - 기본정보와 이자율도 출력



과제- 은행계좌 관리 프로그램

- HighCreditAccount 클래스(NormalAccount 의 자식)
- 신용신뢰계좌 - 입금시 바로 특별이자가 추가로 더해짐
 - 멤버변수
 - 특별 이자율 (%단위, 예:3% => 3)
 - 생성자
 - 메서드
 - 입금하다 오버라이딩
 - 원금과 이자뿐만 아니라 특별이자도 잔액에 더한다
 - 계좌정보를 출력하는 메서드 오버라이딩
 - 특별 이자율도 출력
- 특별이자는 신용등급에 따라 차등 지급된다
 - A 등급은 3%
 - B 등급은 2%
 - C 등급은 1%
 - => interface 이용(상수를 갖는 인터페이스)



과제- 은행계좌 관리 프로그램

- main()에서
 - [1] 입금시 사용자로부터 계좌번호와 입금액을 입력 받고, ArrayList에서 해당 계좌를 찾아서 해당계좌에 입금 처리
 - 해당 계좌가 없으면 "유효하지 않은 계좌번호 입니다." 에러 메시지 출력하기
 - [2] 출금시에도 입금시와 동일하게 처리하되 출금 메시지의 반환값이 0 이면 (출금액이 잔액보다 많은 경우) "잔액부족" 에러 메시지 출력

-----Menu-----
1. 계좌개설
2. 입
3. 출
4. 계좌정보 전체 출력
5. 프로그램 종료
선택: 1

[계좌종류선택]
1. 보통예금계좌
2. 신용신뢰계좌
선택: 1

[보통예금계좌 개설]
계좌ID: 100-200
이름: 홍길동
입금액: 100000
이자율: 3

-----Menu-----
1. 계좌개설
2. 입
3. 출
4. 계좌정보 전체 출력
5. 프로그램 종료
선택: 3

[출금]
계좌ID: 200-300
출금액: 21000
출금완료

-----Menu-----
1. 계좌개설
2. 입
3. 출
4. 계좌정보 전체 출력
5. 프로그램 종료
선택: 1

[계좌종류선택]
1. 보통예금계좌
2. 신용신뢰계좌
선택: 2

[신용신뢰계좌 개설]
계좌ID: 200-300
이름: 김길순
입금액: 200000
이자율: 3

신용등급<1toA, 2toB, 3toC>: 2

[출금]
계좌ID: 100-200
출금액: 800000
잔액부족

-----Menu-----
1. 계좌개설
2. 입
3. 출
4. 계좌정보 전체 출력
5. 프로그램 종료
선택: 2

[입금]
계좌ID: 100-200
입금액: 10000
입금완료

[입금]
계좌ID: 700-800
입금액: 50000
유효하지 않은 ID 입니다.

-----Menu-----
1. 계좌개설
2. 입
3. 출
4. 계좌정보 전체 출력
5. 프로그램 종료
선택: 4

=====전체 고객 계좌 정보=====
계좌ID: 100-200
이름: 홍길동
잔액: 110300
이자율: 3%

계좌ID: 200-300
이름: 김길순
잔액: 221000
이자율: 3%
특별이자율: 2%