

CS1007 lecture #7 notes

tue 24 sep 2002

- news
 - homework #2 due tue oct 1
 - homework #1 should be returned in recitation this week
 - short quiz #1 (5 points) in class on thu sep 26
 - * bring one page of notes
 - * no computers, calculators, phones, etc.
- for, while and do...while loops
- what are classes and methods?
- the `java.lang` package
- the `java.util.Random` class
- the `java.util.Date` class
- reading: *ch 4.1-4.6*

classes.

- *classes* are the block around which Java is organized
- classes are composed of
 - data elements:
 - * *variables* — i.e., their values can change during the execution of a program
 - * *constants* — i.e., their values CANNOT change during the execution of a program
 - like variables, they have a type, a name and a value
 - *methods*
 - * modules that perform actions on the data elements
 - like variables, they have a type, a name and a value
 - unlike variables, the type can be *void*, which means that they don't really have a value
 - * *constructors* — special types of methods used to set up an object before it is used for the first time
- groups of related classes are organized into *packages*

the `java.lang` package.

- the superclass for all Java classes, at the top of the hierarchy
 - `java.lang.Object`
- wrappers around primitive data types; classes that define numeric limits and contain conversion methods
 - `java.lang.Boolean`
 - `java.lang.Character`
 - `java.lang.Byte`, `java.lang.Short`, `java.lang.Integer`,
`java.lang.Long`, `java.lang.Float`, `java.lang.Double`
- string handling functions
 - `java.lang.String`
- math functions
 - `java.lang.Math`

java.lang.Integer class.

- a *constructor*:

```
public Integer( int value );
```

- some *constants*:

```
public static final int MIN_VALUE
```

```
public static final int MAX_VALUE
```

- some *methods*:

```
public int intValue();
```

```
public static String toString( int i );
```

```
public static Integer valueOf( String s );
```

java.lang.String class.

- some *constructors*:

```
public String();  
public String( String value );
```

- some *methods*:

```
public static String valueOf( int i );  
public int charAt( int index );  
public int compareTo( String anotherString );  
public int length();
```

java.lang.Math class.

- some *constants*:

```
public static final double E  
public static final double PI
```

- some *methods*:

```
public static int abs( int a );  
-----  
public static native double sin( double a );  
public static native double cos( double a );  
public static native double tan( double a );  
-----  
public static native double pow( double a, double b );  
public static native double sqrt( double a );  
-----  
public static double random();  
-----
```

looping.

- if you want to do something many times
- two types of loops:
 - counter controlled (last time)
 - condition controlled (today)
- three loop statements:
 - `for`
 - `while`
 - `do`

counter-controlled for loop.

```
public class ex7a {  
    public static void main ( String[] args ) {  
        int n = 10, count;  
        int card=(int)( Math.random()*52 );  
        for ( count=1; count<=n; count++ ) {  
            card=(int)( Math.random()*52 );  
            System.out.println( "count="+count+" card="+card );  
        } // end for  
        System.out.println( "DONE!! count="+count+" card="+card );  
        System.exit( 0 );  
    } // end of main  
} // end of class ex7a
```


counter-controlled while loop.

```
public class ex7b {  
    public static void main ( String[] args ) {  
        int n = 10, count = 1;  
        int card=(int)( Math.random()*52 );  
        while ( count < n ) {  
            System.out.println( "count="+count+" card="+card );  
            card=(int)( Math.random()*52 );  
            count++;  
        } // end while  
        System.out.println( "DONE!! count="+count+" card="+card );  
        System.exit( 0 );  
    } // end of main  
} // end of class ex7b
```

counter-controlled do loop.

```
public class ex7c {  
    public static void main ( String[] args ) {  
        int n = 10, count = 1;  
        int card=(int)( Math.random()*52 );  
        do {  
            System.out.println( "count="+count+" card="+card );  
            card=(int)( Math.random()*52 );  
            count++;  
        } while ( count<n );  
        System.out.println( "DONE!! count="+count+" card="+card );  
        System.exit( 0 );  
    } // end of main  
} // end of class ex7c
```

condition-controlled for loop (1).

```
public class ex7d {  
    public static void main ( String[] args ) {  
        int card1=(int)( Math.random()*52 );  
        int card2=(int)( Math.random()*52 );  
        int count=1;  
        for ( ; card1 != card2; ) {  
            System.out.println( "count="+count+" card1="+card1+  
                                " card2="+card2 );  
            card1=(int)( Math.random()*52 );  
            card2=(int)( Math.random()*52 );  
            count++;  
        } // end for  
        System.out.println( "MATCH! count="+count+" card1="+card1+  
                            " card2="+card2 );  
        System.exit( 0 );  
    } // end of main  
} // end of class ex7d
```

condition-controlled for loop (2).

OR include all updates in the update section of the for loop

```
.  
.   
.   
for ( ; card1 != card2; card1=(int)(Math.random()*52),  
      card2=(int)(Math.random()*52),  
      count++ ) {  
    System.out.println( "count="+count+" card1="+card1+  
      " card2="+card2 );  
} // end for  
.   
.   
. 
```

condition-controlled `while` loop.

```
public class ex7e {  
    public static void main ( String[] args ) {  
        int card1=(int)(Math.random()*52);  
        int card2=(int)(Math.random()*52);  
        int count=1;  
        while ( card1 != card2 ) {  
            System.out.println( "count="+count+" card1="+card1+  
                                " card2="+card2 );  
            card1=(int)(Math.random()*52);  
            card2=(int)(Math.random()*52);  
            count++;  
        } // end while  
        System.out.println( "MATCH! count="+count+" card1="+card1+  
                            " card2="+card2 );  
        System.exit( 0 );  
    } // end of main  
} // end of class ex7e
```

condition-controlled do loop.

```
public class ex7f {  
    public static void main ( String[] args ) {  
        int card1=(int)( Math.random()*52 );  
        int card2=(int)( Math.random()*52 );  
        int count=1;  
        do {  
            System.out.println( "count="+count+" card1="+card1+  
                                " card2="+card2 );  
            card1=(int)( Math.random()*52 );  
            card2=(int)( Math.random()*52 );  
            count++;  
        } while ( card1 != card2 );  
        System.out.println( "MATCH! count="+count+" card1="+card1+  
                            " card2="+card2 );  
        System.exit( 0 );  
    } // end of main  
} // end of class ex7f
```

`java.util.Random` class (1).

- there is another way to generate random numbers besides using the `Math.random()` from the `java.lang.Math` class
- there are two methods defined in the `Random` class:

```
public Random();  
public Random( long seed );  
// constructor -- can be called with or without a seed
```

```
public void setSeed( long seed );  
// sets the seed for the random number generator
```

- this class implements a *pseudo random number generator*
- which is really a sequence of numbers
- the *seed* tells the random number generator where to start the sequence

java.util.Random class (2).

- more methods defined in the Random class, used to get the random numbers:

```
public float nextFloat();  
// returns a random number between 0.0 (inclusive) and  
// 1.0 (exclusive)
```

```
public int nextInt();  
// returns a random number that ranges over all possible  
// int values (positive and negative)
```


`java.util.Date` class (1).

- this class is handy for getting the current date
- or creating a `Date` object set to a certain date
- some methods defined in the `Date` class:

```
public Date();  
public Date( long date );  
// constructor -- called without an argument, uses the  
// current time; otherwise uses the time argument
```

```
public boolean after( Date arg );  
public boolean before( Date arg );  
public boolean equals( Object arg );  
public long getTime();  
public String toString();
```

- computer time is measured in milliseconds since midnight, January 1, 1970 GMT

java.util.Date class (2).

- a `Date` object is handy to use as a seed for a random number generator
- for example:

```
import java.util.*;
public class ex7g {
    public static void main( String[] args ) {
        Date    now = new Date();
        Random rnd = new Random( now.getTime() );
        System.out.println( "here's the first random number: "+
                           rnd.nextInt() );
    } // end of main()
} // end of class ex7g
```