ORF 201

COMPUTER METHODS FOR PROBLEM SOLVING

Lecture 5

Functions aka Methods

The Mortgage Calculator Revisited

Move date-related code out of **`main`**:

```
public static void main(String[] args)
{
    /*****************************************
    * Declare all variables (Note: bad style)
    *****************************************/

    int i, n;
    double r, amt, p, xi;
    Date d = new Date();

    /*****************************************
    * Prompt user for data
    *****************************************/

    n = promptForInt("Enter number of years: ");
    n *= 12;        // Convert to months

    amt = promptForDouble("Enter loan amount: ");

    r = promptForDouble("Enter ann. pct. rate (0.xx): ");
    r /= 12;        // Convert to months

    d.readDate();

    /*****************************************
    * Compute and display payment
    *****************************************/

    p = r*amt/(1-Math.pow(1+r,-n));
```

Mortgage Calculator- Continued

```
        System.out.print("Monthly payment: ");
        System.out.print(Format.floating(9,2,p));
        System.out.println("");

        /*************************************
        * Print out a heading for table
        *************************************/

        System.out.print("Date Balance     ");
        System.out.println("Interest  Principal");

        /*************************************
        * Loop over the months to update balance
        *************************************/

        xi = amt;
        for (i=0; i<n; i++) {
            d.incrMonth();
            System.out.print(d.monthString[d.month]);
            System.out.print(" "+d.day);
            System.out.print(" "+d.year);
            System.out.print(Format.floating(10,2,xi));
            System.out.print("    ");
            System.out.print(Format.floating(7,2, r*xi));
            System.out.print("    ");
            System.out.println(Format.floating(7,2, p-r*xi));
            xi -= p - r*xi;
        }
        System.out.println(Format.floating(10,2,xi));
}
```

Where'd Everything Go?

Created several *methods* to hide details.

promptForInt() and **promptForDouble()** are called *methods*

promptForInt():

The keyword **static** is required on all methods that are invoked by **main**. More on this later.

```
static int promptForInt(String s)
{
    System.out.print(s);
    return Console.in.readInt();
}
```

The **int** (or **double**) in front of the name indicates what kind of thing the method will return to the calling method.

promptFor Double():

The keyword **return** tells what gets returned to the calling method.

```
static double promptForDouble(String s)
{
    System.out.print(s);
    return Console.in.readDouble();
}
```

The stuff inside the parens is called the *argument list*. An argument list consists of zero, one, or more variables with their types indicated.

Question: What about **d.readDate()** and **d.incrMonth()?** Read on...

Methods Added to Class Date

readDate();

Code would be cleaner if it used **`promptForInt()`** and **`promptForDouble()`**, but these aren't available here. Fix this later.

```
public void readDate()
{
    System.out.println("Enter starting date…");
    do {
        System.out.print(" Enter Month (1-12): ");
        month = Console.in.readInt();
    } while (month<1 || month>12);

    do {
        System.out.print(
                " Enter Day(1-"    // a string
                + numDays[month]+ // an int
                "): "              // another string
        );
        day = Console.in.readInt();
    } while (day<1 || day>numDays[month]);

    System.out.print(" Enter Year: ");
    year = Console.in.readInt();
}
```

Data validation is part of data input.

Variables **`month`**, **`day`**, and **`year`** don't have a "**`d.`**" in front.

More Methods Added to Class Date

incrMonth();

```
public void incrMonth()
{
    month++;
    if (month>12) {
        month=1;
        year++;
    }
    if (day>numDays[month]){
        day = numDays[month];
    }
}
```

Class Date Needs Two Arrays of Constants

Syntax for initializing *static final arrays* is peculiar.

```
static final int[] numDays
             = {0, 31, 28, 31, 30, 31, 30,
                   31, 31, 30, 31, 30, 31};

static final String[] monthString
             = {"", "Jan", "Feb", "Mar", "Apr",
                    "May", "Jun", "Jul", "Aug",
                    "Sep", "Oct", "Nov", "Dec"};
```

Note that **`new`** is absent. Braces are in lieu of **`new`**.

Mortgage3.java

Final attempt. Move **`promptForInt`** and **`promptForDouble`** to **`Console.in`** in **`ccj`** utilities. Rename these methods **`readInt`** and **`readDouble`**.

In **`ccj/TextInputStream.java`**:

```
public int readInt(String s)
{
    Console.out.print(s);
    return readInt();
}

public double readDouble(String s)
{
    Console.out.print(s);
    return readDouble();
}
```

Two methods can have the same name but different argument list. They are different methods. Look at handout: Mortgage3.java