Lecture 3

Example

The purpose of this example is to provide the reader some experience with using Java's boolean variables.

In this program the user will enter a 5 digit positive integer. The program will check to see if the integer entered meets these conditions. If it doesn't it will tell the user and wait for the user to type in the correct kind of integer. Once this is done, the program will print out the digits of the integer in reverse order.

Here is a typical run: The user types in a 4 digit number and then clicks on OK; the program checks and then informs the user to type in a 5 digit one.

The user does and presses OK; the computer then prints out the digits of the integer in reversed order.

Here is the outline of what has to be done:

1) Have the program prompt the user to enter a 5 digit number
2) Check to see if this was done; if not, inform the user of the error
3) Once the input is right, print out the reversed digits

We begin with an outline of the program:
import javax.swing.*;
public class FiveDigits
{
    public static void main(String args[])
    {
      // end void main(String args[])
    } // end class FiveDigits

    Note that the very first line of the program

    import javax.swing.*;

    just contains the name of the package that contains the class,
    JOptionPane, that we are going to use in this program. There is no need
    to specify the class; the program will get it and its methods as
    needed.

    To continue, enter the code for capturing the integer n the user
    enters:

    import javax.swing.*;
public class FiveDigits
{
    public static void main(String args[])
    {
      String w;
      int n;
      w = JOptionPane.showInputDialog("Enter a 5 digit integer");
      n = Integer.parseInt(w);
      System.out.println("n = " + n);
      System.exit(0);
    } // end void main(String args[])
}// end class FiveDigits

The line System.out.println("n = " + n); is inserted to check
whether the code we have so far works. When this program is run it
produces the first pane:
Then, after the OK button is clicked it prints out "n = 3245" on the DOS command window.

So, the first part of our program is checks out.

The line `System.exit(0)` in the last line of the program above is there to provide a graceful exit from the program to the DOS window. If it were not there then, after typing in 3245 and clicking on the OK button the DOS window would hang. The only thing that can be done then is to press the Ctrl-Alt- sequence and close the DOS window with the "end task" command.

The next problem is that of checking whether or not the integer type in has 5 digits. This is done by introducing the boolean variable `hasFiveDigits`, initialized as false, and using it to see if the entered integer has five digits, i.e., falls in the range from 10000 to 99999. If it does not the program keeps cycling until the user enters a 5 digit integer.

We add the code for checking. (The previous lines are in blue. The line "System.out.println(...) " has served its purpose, so has been erased. The new code that checks the input is in red.)
import javax.swing.*;
public class FiveDigits
{
    public static void main(String args[])
    {
        String w;
        int n;
        boolean hasFiveDigits = false;
        w = JOptionPane.showInputDialog("Enter a 5 digit integer");
        n = Integer.parseInt(w);
        if (n >= 10000 && n < 100000) hasFiveDigits = true;
        while (!hasFiveDigits)
        {
            w = JOptionPane.showInputDialog("The number you entered did not have 5 digits. Enter one that does");
            n = Integer.parseInt(w);
            if (n >= 10000 && n < 100000) hasFiveDigits = true;
        }
        System.exit(0);
    } // end void main(String args[])
} // end class FiveDigits

Checking that this works, we find that it does:

We turn to the problem of determining the digits of a 5 digit integer. We shall use this feature of java:

If n and d are declared to be of type int then, in java, n/d is the integral part of the real number n/d.

For example, if n = 34 and d = 5 then java returns 6 for n/d; similarly, if n = 43 and d = 3 then n/d is, in a java program, 14.

So, suppose that the user inputs the 5 digit integer 35791. Let n1 = 35791. Then, in java,

\[ d1 = \frac{35791}{10000} = 3 \] = the first digit
Recall next a modulo b is the remainder that results when a is divided by b. In java a modulo b is written as a\%b. Thus

\( n_2 = n_1 \% 10000 = 5791 \) = the last four digits of 35791.

The process repeats itself:

\( d_2 = n_2 / 1000 = 5791 / 1000 = 5 \)

\( n_3 = n_2 \% 1000 = 791 \)

\( d_3 = n_3 / 100 = 7 \)

\( n_4 = n_3 \% 100 = 91 \)

\( d_4 = n_4 / 10 = 9 \)

\( n_5 = n_4 \% 10 = 1 \)

\( d_5 = n_5 / 1 = 1 \)

The final stage of our program is listed below. As before, previous code is in blue; new code is in red. When run it produces the panes displayed at the start of this lecture.
import javax.swing.*;

public class FiveDigits
{
    public static void main(String args[])
    {
        String w, output;
        int n, n1, n2, n3, n4, n5, d1, d2, d3, d4, d5;
        boolean hasFiveDigits = false;
        w = JOptionPane.showInputDialog("Enter a 5 digit integer");
        n = Integer.parseInt(w);
        if (n >= 10000 && n < 100000) hasFiveDigits = true;
        while (!hasFiveDigits)
        {
            w = JOptionPane.showInputDialog("The number you entered did not have 5 digits. Enter one that does");
            n = Integer.parseInt(w);
            if (n >= 10000 && n < 100000) hasFiveDigits = true;
        }

        n1 = n;    // n1 = 35791
        d1 = n/10000;    // d1 = 3
        n2 = n1 % 10000;    // n2 = 5791
        d2 = n2 / 1000;    // d2 = 5
        n3 = n2 % 1000;    // n3 = 791
        d3 = n3 / 100;    // d3 = 7
        n4 = n3 % 100;    // n4 = 91
        d4 = n4 / 10;    // d4 = 9
        n5 = n4 % 10;    // n5 = 1
        d5 = n5;    // d5 = 1
```java
output = "The number you entered was " + n + "\n";
output = output + "The reversed digits are " + "\n";
output = output + d5 + "\n" + d4 + "\n" + d3 + "\n" + d2 + "\n" + d1;
JOptionPane.showMessageDialog
(  
null,  
output,  
"reversed",  
JOptionPane.INFORMATION_MESSAGE
);
System.exit(0);
} // end void main(String args[])  
} // end class FiveDigit

Other Data Types

    The eight data types boolean, char, byte, short, int, long,  
    float, and double were mentioned at the start of this lecture. In the  
examples we have presented the emphasis has been on the boolean, int  
and double types. We shall use the char (character) type later when we  
turn to dealing with strings; a string is a sequence of characters.  

    There are two other primitive types of integers, the byte type  
and the short type. The byte is supported by 8 bits and the long by 16.  
The int, recall, is based on 32 bits.  

    There is one other decimal type, the type float. It is a real  
number supported by 32 bits instead of the 64 bits that underlie a  
double. These different types of integers and reals can be of use if  
space in memory or speed is an issue for your program.
Lab 2

Write an application that will prompt the user to enter a six digit (positive) integer, check that a six digit number has been entered, will not continue until an integer with exactly six digits has been entered, and then returns the sum and product of the digits.

Call the name of the text file for your program SixDigits.java

This would be a typical session:

<table>
<thead>
<tr>
<th>Image of input dialog showing incorrect entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User doesn't bother to read the directions</td>
<td></td>
</tr>
<tr>
<td>Program informs him of his error</td>
<td></td>
</tr>
<tr>
<td>He is not too bright</td>
<td></td>
</tr>
<tr>
<td>He is informed of his error</td>
<td></td>
</tr>
<tr>
<td>He finally catches on</td>
<td></td>
</tr>
<tr>
<td>The program calculates the sum and the product</td>
<td></td>
</tr>
</tbody>
</table>