

## do - while Repetition Structure

- ❖ Loop structure that guarantees the loop body is executed once.
- ❖ Condition is tested at bottom of loop
- ❖ Don't forget the semicolon for while(...);

```

graph TD
    Start(( )) --> Init[Initialize Counter]
    Init --> Do[do]
    Do --> Body[statement1;  
statement2;  
Increment Counter]
    Body --> Check{Check Counter}
    Check -- True --> Do
    Check -- False --> End(( ))
            
```

```

int nI = 0;
do
{
    System.out.print("Loop");
    System.out.println(nI);
    nI++;
}while(nI < 5);
System.out.println("\nEnd"+nI);
            
```

Copyright © 2006 R.M. Laurie 1

## for loop Flow Chart

- ❖ Initialize, assertion check, and increment/decrement in one statement

```

graph TD
    Start(( )) --> Init[Initialize Counter]
    Init --> Check{Check Counter}
    Check -- True --> Body[statement1;  
statement2;  
Increment Counter]
    Body --> Check
    Check -- False --> End(( ))
            
```

```

public class ForExample0
{
    public static void main(String args[])
    {
        int nI;
        for(nI = 0; nI < 5; nI++)
        {
            System.out.print("Loop");
            System.out.println(nI);
        }
        System.out.println("\nEnd"+nI);
    }
}
            
```

Loop0  
Loop1  
Loop2  
Loop3  
Loop4  
End5

Copyright © 2006 R.M. Laurie 2

## for Example

```

import java.text.*;
public class ForExample1
{
    public static void main(String args[])
    {
        int nI;
        String sOutput;
        DecimalFormat oNum =
            new DecimalFormat("0000");
        sOutput = "Number Square Cube\n"
            + "-----\n";
        for(nI=1; nI <= 10; nI++)
        {
            sOutput += " " + oNum.format(nI)
                + " " + oNum.format(nI*nI)
                + " " + oNum.format(nI*nI*nI)+"\n";
        }
        System.out.println(sOutput);
    }
}
            
```

Number	Square	Cube
0001	0001	0001
0002	0004	0008
0003	0009	0027
0004	0016	0064
0005	0025	0125
0006	0036	0216
0007	0049	0343
0008	0064	0512
0009	0081	0729
0010	0100	1000

## Nested for Loop Example

```

public class ForNested
{
    public static void main(String args[])
    {
        int nI, nJ;
        System.out.printf("Multipliers\n");
        for(nI = 1; nI <= 5; nI++)
        {
            System.out.printf("Row%2d:", nI);
            for( nJ = 8; nJ <= 32; nJ *= 2)
            {
                System.out.printf("%4d ", nI*nJ);
            }
            System.out.printf("%n");
        }
    }
}
            
```

Multipliers			
Row 1:	8	16	32
Row 2:	16	32	64
Row 3:	24	48	96
Row 4:	32	64	128
Row 5:	40	80	160

## Slide Set 11: Java Loop

```
import javax.swing.*;
public class ExceptionAvg
{
    public static void main(String[] args)
    {
        int nScore, nSum=0, nI=1;
        String sEntry;
        char cSelect;
        while(nI <= 50)
        {
            try
            {
                sEntry = JOptionPane.showInputDialog(null,
                "Enter Score "+ nI + " or \'q\' to Quit:");
                cSelect = sEntry.charAt(0);
                if(cSelect == 'q')break;
                nScore = Integer.parseInt(sEntry);
                nSum = nSum + nScore;
                JOptionPane.showMessageDialog(null, "Score "+nI+" = "+nScore);
                nI++;
            }
            catch(NumberFormatException n)
            {
                JOptionPane.showMessageDialog(null, "ERROR: Enter Number!");
            }
            catch(NullPointerException n)
            {
                break;
            }
        }
        JOptionPane.showMessageDialog(null, "FINAL AVERAGE = " + nSum/nI);
        System.exit(0);
    }
}
```

## Writing a String to a Text File

- ❖ **FileWriter** is a simple Java class used for writing to a file
- ❖ This will overwrite an existing file with the same name or create a new file

```
import java.io.*;
public class WriteFileStr
{
    public static void main(String[] args)
        throws IOException
    {
        FileWriter oFile = new FileWriter("c:/Out1.txt");
        oFile.write("Testing 1\n");
        oFile.close();
    }
}
```

Copyright © 2006 R.M. Laurie 6

## Appending a String to a Text File

- ❖ Appending to a file is done by putting **true** after file name when creating the file object

```
import java.io.*;
public class WriteFileStr
{
    public static void main(String[] args)
        throws IOException
    {
        FileWriter oFile = new FileWriter("C:/Out2.txt", true);
        oFile.write("Testing 2\n");
        oFile.close();
        System.out.println("Done");
    }
}
```

Copyright © 2006 R.M. Laurie 7

## User Input File Name and Data

- ❖ Appending to a file is done by putting **true** after file name when creating the file object

```
import javax.swing.*;
import java.io.*;
public class PromptWrite
{
    public static void main(String[] args) throws IOException
    {
        String sEntry, sFile;
        sFile = JOptionPane.showInputDialog(null, "Enter the File Name");
        FileWriter oFile = new FileWriter(sFile, true);
        sEntry = JOptionPane.showInputDialog(null, "Enter the Data");
        oFile.write("\n" + sEntry);
        oFile.write("\n4 x 7 = " + 4*7);
        JOptionPane.showMessageDialog(null, "Done");
        oFile.close();
        System.exit(0);
    }
}
```

Copyright © 2006 R.M. Laurie 8