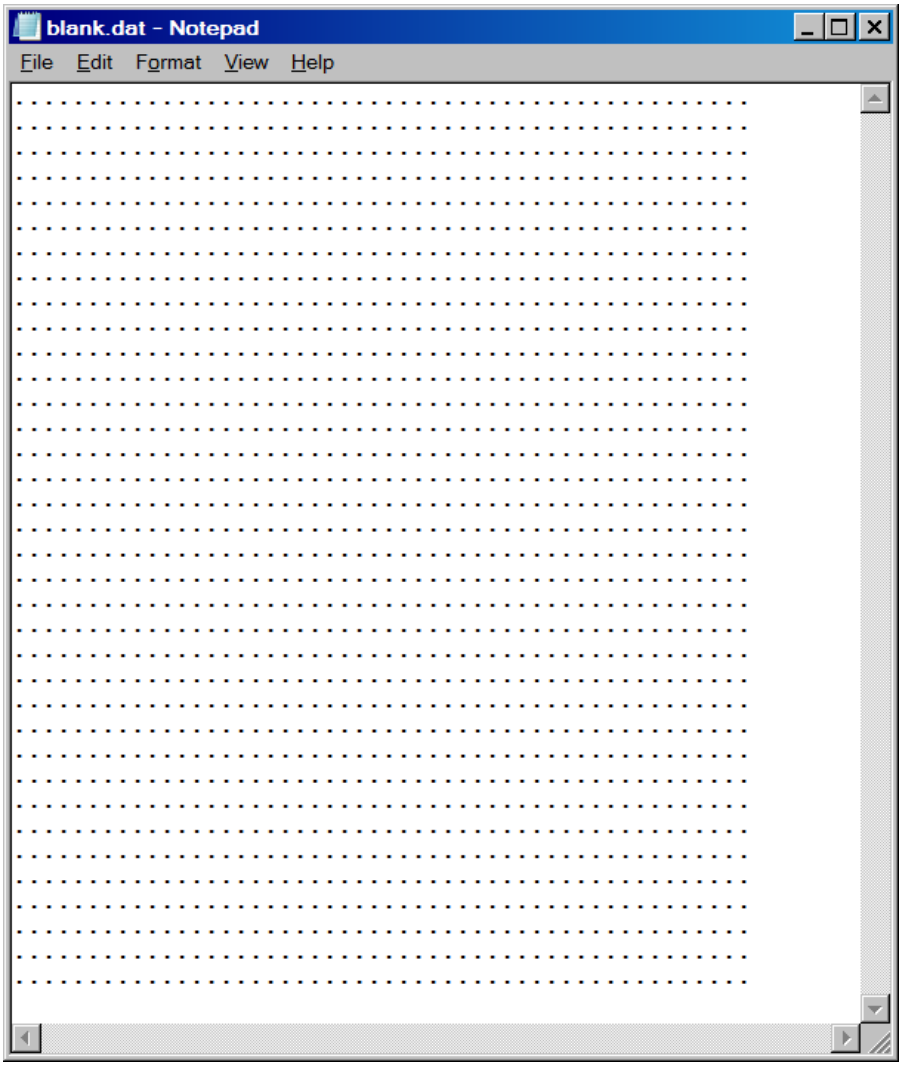
Advanced Graphics Programming	Unit 03 Lab
Graphics Background File Program	**90, 100 & 110 Point Versions**

Assignment Purpose:

The purpose of this assignment is to combine knowledge of text file editing, text file transfers with basic AWT Graphics class command to create simple backgrounds for video games.

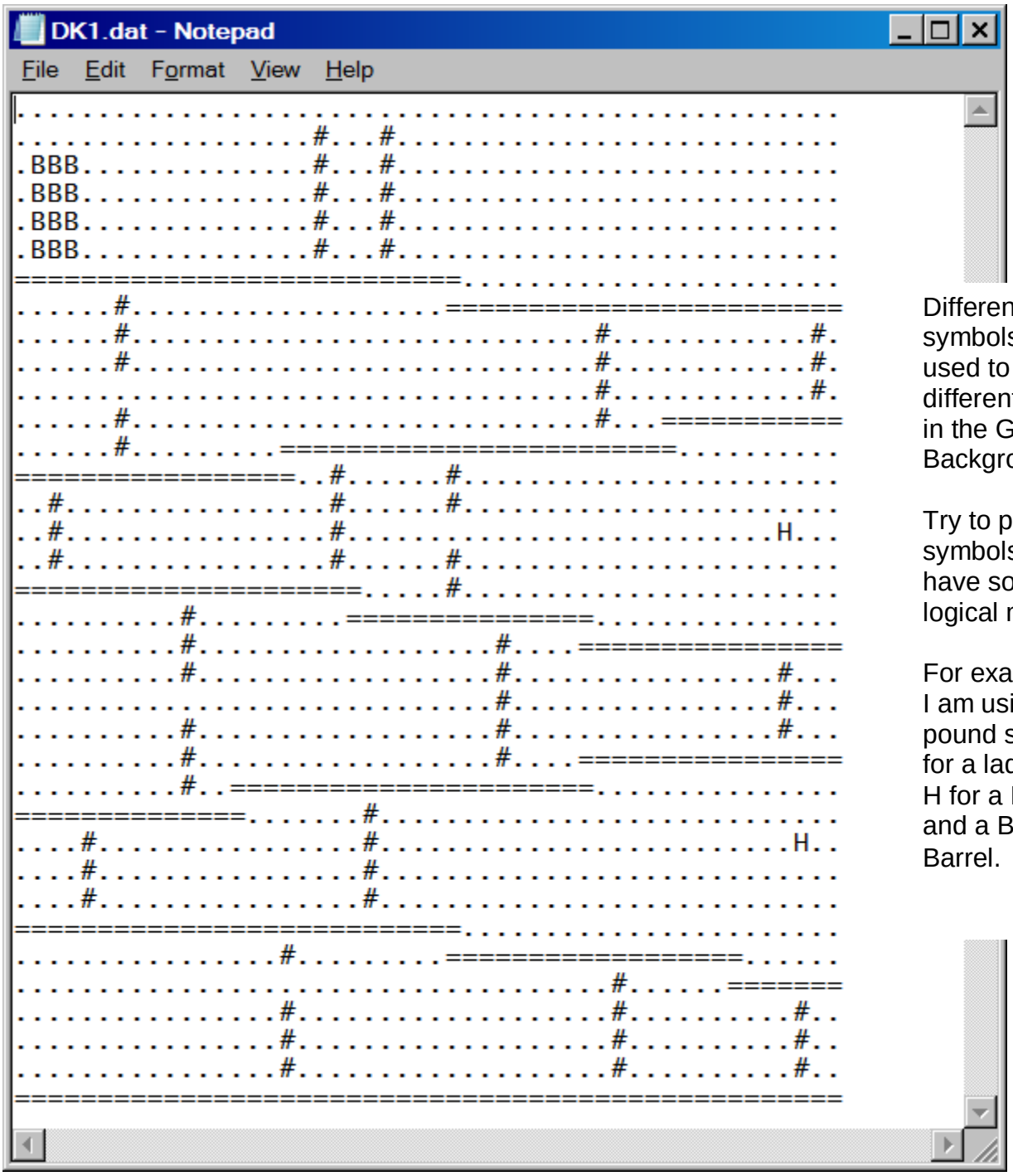
You need to write a program that will take a text file, and convert it into a graphics background. In addition to the **Unit03vst.java** file (shown later), you are also provided with several text files to test your program. The **blank.dat** file is shown below. The **DK1.dat** file is on the next page, which is used for the 90-point version of the program.



The **blank.dat** file contains 36 rows of 50 periods.

Use this file to design your Graphics Background.

An example is shown on the next page.



Different text symbols are used to represent different objects in the Graphics Background.

Try to pick symbols that have some logical meaning.

For example, I am using a pound sign (#) for a ladder, an H for a Hammer, and a B for a Barrel.

Using the provided student starting file, you need to write a program that will ask the user for a file name, read in that text file, and then display the Graphics Background like the example on the next page.

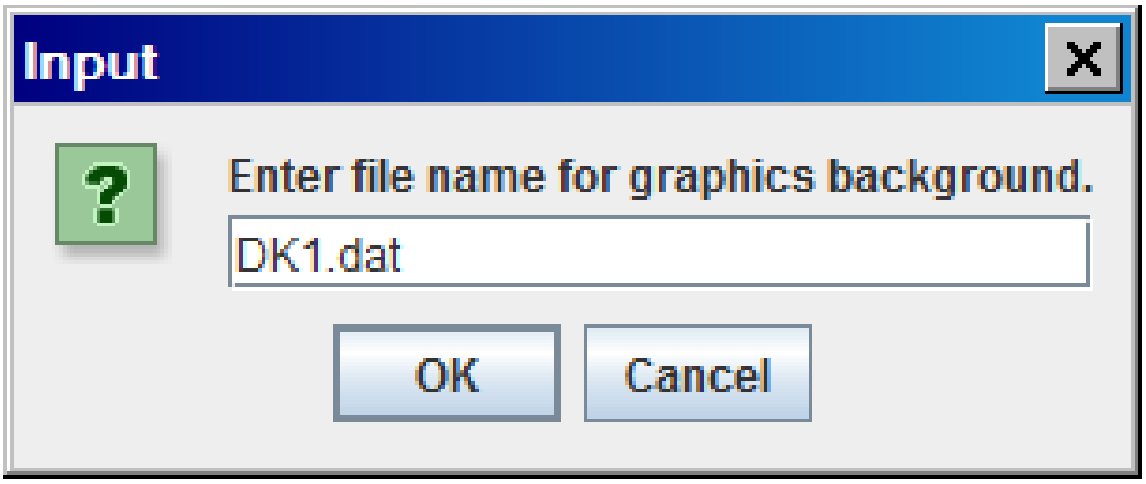
Unit03vST.java Provided Student File

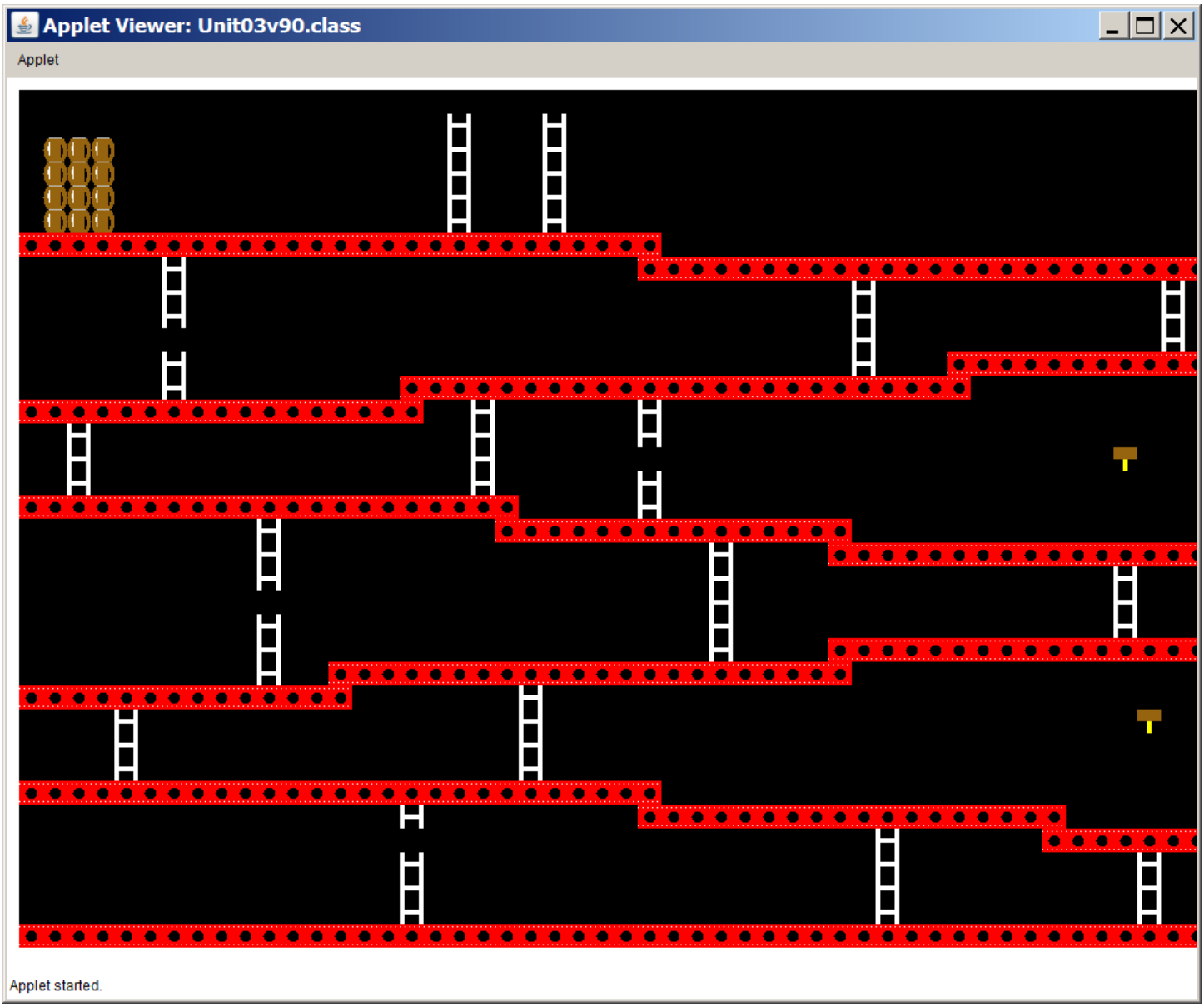
```
// Unit03vST.java
// 07-18-10 by Leon Schram
// This is the student starting program for the Unit03 lab assignment.
// When you execute this program, enter DK1.dat, DK2.dat or blank.dat


import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.*;
import java.awt.*;
import javax.swing.JOptionPane;


public class Unit03vST extends Frame implements WindowListener
{

   private int numRows = 36;  //  35 Rows are displayed.  The top row is hidden behind the title
bar.
   private int numCols = 50;
   private String background[];


   public Unit03vST()
   {
      this.addWindowListener(this);

      String fileName = JOptionPane.showInputDialog("Enter file name for graphics background.");
      background = new String[numRows];
      try
      {
         BufferedReader inStream = new BufferedReader(new FileReader("resources/" + fileName));
         String line;
         int row = 0;
         line = inStream.readLine();
         while(line != null)
         {
            background[row] = line;
            row++;
            line = inStream.readLine();
         }
      }
      catch (IOException e)
      {
         System.out.println("There were problems with the code as stated below\n");
         System.out.println(e.getMessage());
      }
      System.out.println();
      repaint();
   }


   public void paint(Graphics g)
   {

   }
```

```
   public int convert(int q)
   {
      return 0;
   }


   public void drawSpace (Graphics g, int y, int x)
   {

   }


   public void drawGirder (Graphics g, int y, int x)
   {

   }


   public void drawLadder (Graphics g, int y, int x)
   {

   }


   public void drawHammer(Graphics g, int y, int x)
   {

   }


   public void drawBarrel (Graphics g, int y, int x)
   {

   }
   public static void main(String [] args) {
      Unit03vST f = new Unit03vST ();
      f.setSize(1000,800);
      f.setVisible(true);
      f.setLayout(new FlowLayout());
   }

   public void windowClosing(WindowEvent e)
   {
      dispose();
      System.exit(0);
   }
   public void windowOpened(WindowEvent e)
   { }
   public void windowIconified(WindowEvent e)
   { }
   public void windowClosed(WindowEvent e)
   { }
   public void windowDeiconified(WindowEvent e)
   { }
   public void windowActivated(WindowEvent e)
   { }
   public void windowDeactivated(WindowEvent e)
   { }
}
```

90-Point Version

For the 90-point version of the program, you need to enter the **DK1.data** file name. This will use the correct text file for the classic Donkey Kong game background. The background display is shown on the next page.





Each text symbol represents a 20 x 20 pixel square on the graphics screen. Your display needs to have 35 rows and 50 columns. NOTE: The file will have 36 rows. The first row (row 0) will not be visible since the title bar of the graphics window covers it up. You will also need to adjust the output 10 pixels down and 10 pixels over to make the image display properly. This means that the upper-left hand corner of the very first square displayed (row 0, column 0) will be at (10,10). However, since row 0 will not be visible, the upper-left hand corner of the very first visible square (row 1, column 0) will be at (10,30). The next square is at (30,30). The next is at (50,30) and so forth.

The period "." in the text file indicates empty space. The ***drawSpace*** method simply requires a 20 x 20 pixel square to be drawn in the desired background color. You need to complete the appropriate methods to display the other characters in the background.

100-Point Version

For the 100-point version you also need a ***drawUnknown*** method which will draw something a question mark to indicate the user has a character in the file that is not one from your list. The **DK2.dat** text file intentionally has several unknown characters, which must be used to test the 100 point version of the program. Note the pink question marks in the output below.