1 /\*  
 2 Kevin Baron  
 3 11/17/12  
 4 1/27/13  
 5 Crayola 64  
 6 \*/  
 7   
 8 import java.awt.\*; //for Graphics and DrawingPanel  
 9 import java.util.\*; //for Scanner  
 10 import java.io.\*; //for File and FileNotFoundException  
 11   
 12 public class Crayola64\_Commented {  
 13   
 14 public static final boolean WHITEOUTLINE = false; //whether or not there is an outline around the blank color used for space.  
 15 public static final int FONTSIZE = 5;  
 16 public static final int COLUMNWIDTH = 150; //how many characters wide the screen will be.  
 17 public static final int YMARGIN = 20;  
 18 public static final int XMARGIN = 20;  
 19   
 20 //process a message.  
 21 public static void main(String[] args) throws FileNotFoundException {  
 22 processor(getMessage());  
 23 }//eomain  
 24   
 25 //give an intro and ask for a message. Return the message.  
 26 public static String getMessage() {  
 27 System.out.println("This program takes a line you type and converts it to Crayola-64.");  
 28 System.out.println("What is your message?");  
 29 return new Scanner(System.in).nextLine();  
 30 }//eogetMessage  
 31   
 32 //draw the panel and determine where to put each character on the panel.  
 33 public static void processor(String message) throws FileNotFoundException {  
 34 DrawingPanel panel = new DrawingPanel(panelX(message), panelY(message));  
 35 Graphics g = panel.getGraphics();  
 36   
 37 int x = XMARGIN;  
 38 int y = YMARGIN;  
 39   
 40 for (int i = 0; i < message.length(); i++) {  
 41 char ch = message.charAt(i);  
 42   
 43 //handle capitals and <{[(/ characters differently by preceding with a '`'.  
 44 if (ch == 40 || ch == 60 || ch == 47 || ch == 91 || ch == 123 || (ch > 64 && ch < 91)) {  
 45 converter(g, '`', x, y);  
 46 if (ch > 64 && ch < 91) {  
 47 ch = (char) (ch + 32);//now treat the capital letter like lowercase.  
 48 }//eoif  
 49 //advance the x-marker and go to the next line if necessary.  
 50 x += FONTSIZE;  
 51 if (x >= XMARGIN + COLUMNWIDTH \* FONTSIZE) {  
 52 x = XMARGIN;  
 53 y += FONTSIZE;  
 54 }//eoif  
 55 }//eoif  
 56   
 57 //convert the raw character.  
 58 converter(g, ch, x, y);  
 59 x += FONTSIZE;  
 60 if (x >= XMARGIN + COLUMNWIDTH \* FONTSIZE) {  
 61 x = XMARGIN;  
 62 y += FONTSIZE;  
 63 }//eoif  
 64   
 65 //handle \}])> by adding a '`' to the end.  
 66 if (ch == 41 || ch == 62 || ch == 92 || ch == 93 || ch == 125) {  
 67 converter(g, '`', x, y);  
 68 x += FONTSIZE;  
 69 if (x >= XMARGIN + COLUMNWIDTH \* FONTSIZE) {  
 70 x = XMARGIN;  
 71 y += FONTSIZE;  
 72 }//eoif  
 73 }//eoif  
 74 }//eofor  
 75 }//eoprocessor  
 76   
 77 //scan rgb.txt to determine what RGB values are used for each character. If the character is not found, the default is white.  
 78 public static void converter(Graphics g, char ch, int x, int y) throws FileNotFoundException {  
 79 Scanner input = new Scanner(new File("rgb.txt"));  
 80 int red = 255;  
 81 int green = 255;  
 82 int blue = 255;  
 83 while (input.hasNextLine()) {  
 84 Scanner info = new Scanner(input.nextLine());  
 85 if (info.nextInt() == ch) {  
 86 red = info.nextInt();  
 87 green = info.nextInt();  
 88 blue = info.nextInt();  
 89 }//eoif  
 90 }//eowhile  
 91 g.setColor(new Color(red, green, blue));  
 92 g.fillRect(x, y, FONTSIZE, FONTSIZE);  
 93 if (ch == 32 && WHITEOUTLINE == true) {  
 94 g.setColor(Color.BLACK);  
 95 g.drawRect(x, y, FONTSIZE - 1, FONTSIZE - 1);  
 96 }//eoif  
 97 }//eoconverter  
 98   
 99 //tells how wide the panel should be given the length of the message  
100 public static int panelX(String message) {  
101 if (COLUMNWIDTH < outputActualLength(message)) {  
102 return (2 \* XMARGIN + FONTSIZE \* COLUMNWIDTH);  
103 } else {  
104 return (2 \* XMARGIN + FONTSIZE \* outputActualLength(message));  
105 }//eoifelse  
106 }//eopanelX  
107   
108 //tells how tall the panel should be given the length of the message  
109 public static int panelY(String message) {  
110 return (2 \* YMARGIN + FONTSIZE \* ((outputActualLength(message) - 1) / COLUMNWIDTH + 1));  
111 }//eopanelY  
112   
113 //because some characters require two colors to represent them,  
114 //the length of the typed message might be different from the output length.  
115 public static int outputActualLength(String message) {  
116 int numberOfDoubled = 0;  
117 String list = "ABCDEFGHIJKLMNOPQRSTUVWXYZ()<>[]{}/\\";  
118 for (int i = 0; i < message.length(); i++) {  
119 if (list.indexOf(message.charAt(i)) != -1) {  
120 numberOfDoubled++;  
121 }//eoif  
122 }//eofor  
123 return message.length() + numberOfDoubled;  
124 }//eooutputActualLength  
125   
126 }//eoclass