**Huffman Coding with Images**

We have done Huffman coding and decoding with text. Doing Huffman coding with images seems to be conceptually very similar: instead of processing a line of text, we process a 2-D array of pixels or colors or whatever Java uses. We still need a map for a frequency table, a priority queue of nodes, a Huffman node, a tree, and the tree traversals to concatenate the Huffman code. We need scanners, files, and file writers.

The last time you processed images was probably PixLab. You likely never learned important details about images.

1. Is it better to use .gif, .png., .bmp, or .jpg?

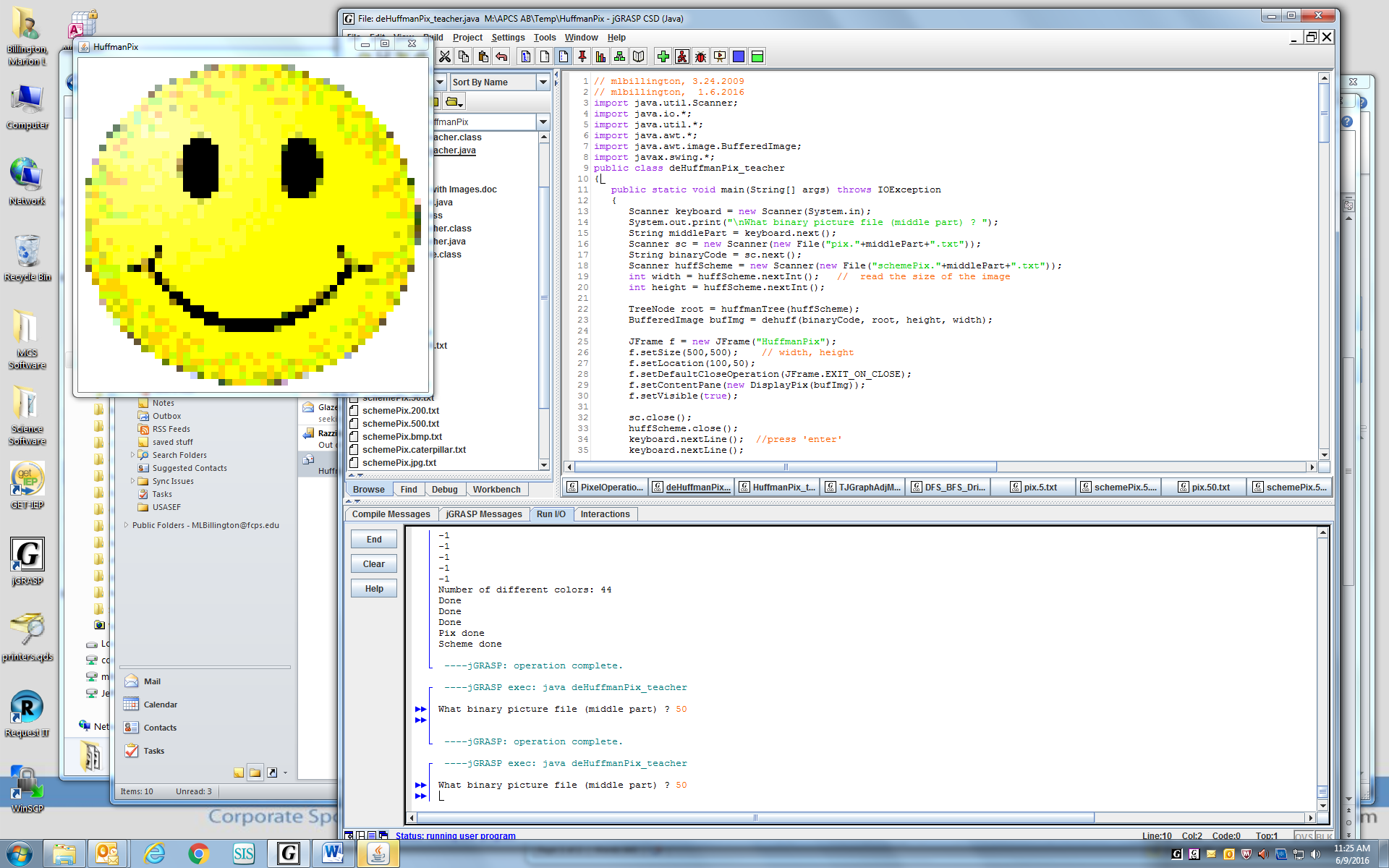
1. How exactly does BufferedImage store colors? What commands get and set the colors? Is the array row-major or column-major?
2. How can the computer recognize whether colors are the same or not?
3. Should the Huffman node store colors, pixels, Integers, Objects, or what?
4. How can we be sure that we are counting colors correctly? Can I work with a 5 x 5 image at first and see what’s happening? Does the debugger work with Buffered Images?
5. It sure would be nice to see the image, both before it is Huffman-coded and after it is decoded. What are the minimum commands to do that?

A run of HuffmanPix.java, reading smiley.gif and setting WIDTH=50; HEIGHT=50;

produces pix.50.txt 10101010101010101010101010101010101010101010101010101010101010101010101010 . . . etc.

and schemePix.50.txt

50 50  
-13413120 00000000  
-3364352 00000001  
-10957 0000001  
-10070733 0000010  
-10906 0000011  
-3353344 00001  
-11008 0001  
-6706535 00100000  
-10855 00100001  
-3342388 001000100  
etc.



Then a run of deHuffmanPix.java, typing in the "middle part" (50), shows this image:

1. **Is it better to use .gif, .png., .bmp, or .jpg?**

.gifs use fewer colors. I recommend using smiley.gif at first. It turns out that .bmp files don’t work, but I don’t know why.

1. **How exactly does BufferedImage store colors? What commands get and set the colors? Is the array row-major or column-major?**

|  |  |
| --- | --- |
| -1 | white |
| -154 | light yellow |
| -205 | yellow |
| -11008 | orange |
| -3342592 | light green |
| -16777216 | black |

BufferedImage stores a 2-D array of colors represented by “packed ints.” You don’t need to unpack them to Colors, but you can if you want. By printing out some packed ints and looking at the 5x5 image, I discovered the values in the chart at the right.

You get a single packed int from the 2-D array with int rgb = img.getRGB(c,r); You set a single packed int into the BufferedImage with img.setRGB(c,r, rgb);

The array is column-major! Loop over each column, then down each row. c goes with WIDTH and r goes with HEIGHT. (This caused me hours of puzzlement, until I figured it out.)

1. **How can the computer recognize whether colors are the same or not?**

Since the packed ints are ints which auto-convert to Integers, the computer has no trouble with statements like: if( map.containsKey( rgb ) )

1. **Should the Huffman node store colors, pixels, Integers, Objects, or what?**

I made my Huffman node store Objects. I am sure that Integers would work, too.

1. **How can we be sure that we are counting colors correctly? Can I work with a 5 x 5**

**image at first and see what’s happening? Does the debugger work with a BufferedImage?**

It is easy to count the colors after making the map of frequencies, using System.out.println("Count = "+ map.size() );

I recommend using at first a 5x5 image in a 500x500 frame.

The debugger works fine with images. Just drag out the BufferedImage. I would not have been able to do deHuffman without observing how the decoded 0’s and 1’s put the packed ints into the 5x5 BufferedImage. Columns and rows!

1. **It sure would be nice to see the image, both before it is Huffman-coded and after it is decoded. What is the minimum set of commands for that?**

The shell is done for you. Note that the Frame size (500, 500) is different from the image size WIDTH = 5; HEIGHT = 5;

Extra: If you did the column-row processing correctly, everything should work with a rectangular image. Try caterpillar.gif using Frame (500, 300) and WIDTH = 50; HEIGHT = 30;