CSE 410 - Computer Systems
Programming Project 3 Questions

Assigned: Wednesday, October 24, 2001
Due: Wednesday, October 31, 2001 before class

Your name: ________________________________

1a. Consider the first two filter procedures you wrote to implement the flip and the mirror. You used a variety of labels to identify specific points in the code. Starting with the first label of each procedure (the same label that is stored in the filterJumpTable), write down each label you used, and then briefly state what condition causes the code to jump or branch to that label.

1b. Is there a significant difference in the complexity of the two procedures? If so, why?

2a. In class I talked a lot about making assumptions about branches being taken versus not being taken. Look at the procedure in the skeleton that implements the copy filter and find the one branch instruction in the procedure. If we use this procedure to copy the 5x4 pixel image in the writeup on page 3, how many times does the branch instruction get executed?

2b. How many of those times are "branch taken" and how many are "branch not taken"?

2c. For each branch instruction in your flip filter procedure, write down the instruction and an approximate formula for calculating how many times the branch is taken, based on the width and height of the image or other critical factors. I am not concerned about exact formulas, just a rough estimate of how many times each branch will be taken based on the image you are processing.

3. The BMP header format is an example of the non-aligned data that can often be a problem when a file format is fixed by an outside system that has fewer constraints on word alignment. Take a look at the procedure readBMPHeader in the skeleton. The word-aligned address of the header array is passed to it in register $a1. The procedure makes six references to words in the header using offsets and the contents of $a1.

   The first of the six is

   ```
   lhu    $t1,0($a1)      # the actual tag
   ```

   This instruction retrieves a half-word (16-bit) quantity from offset 0. It is half-word aligned because the offset is a multiple of the size of the quantity in bytes.

   Write down the other five instructions that make reference to the array using $a1. For each reference, write down the size of the quantity being retrieved or stored, and whether or not it is "aligned", considering its size.

4. Which of the extra filters did you implement? How well does it work or not work?

5. Did you implement any of the extra credit filters? If so, list each filter you implemented and briefly describe how well it works or doesn't work.