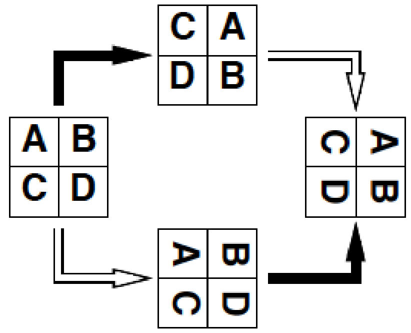
Assignment 1, problem 2:

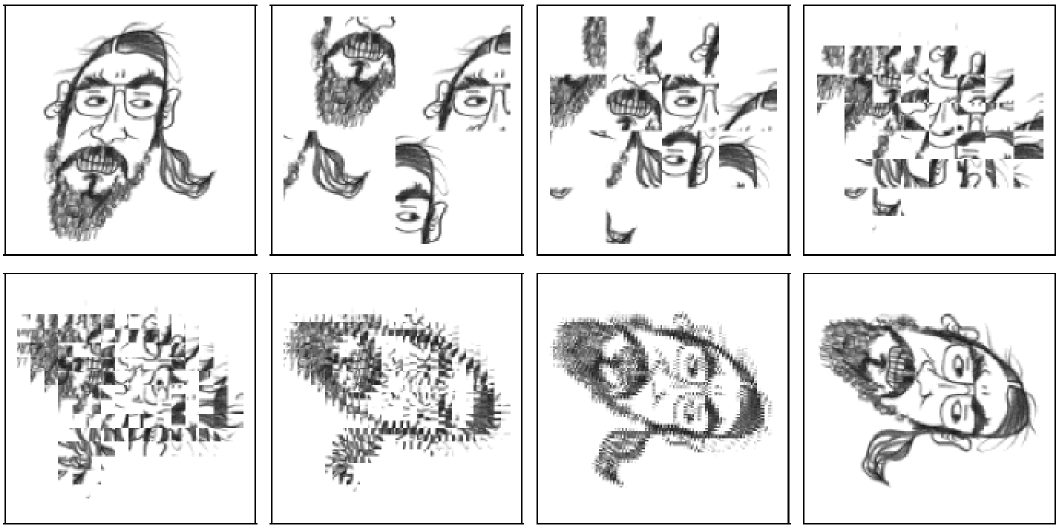
Test: complexity, (divide&conquer,).

Most graphics hardware includes support for a low-level operation called *blit*, or block transfer, which quickly copies a rectangular chunk of a pixel map (a two-dimensional array of pixel values) from one location to another. This is a two-dimensional version of the standard C library function memcpy() and Javas System.arraycopy(.).

Suppose we want to rotate an $n*n$ pixel map 90° clockwise. One way to do this, at least when n is a power of two, is to split the pixel map into four $n/2*n/2$ blocks, move each block to its proper position using a sequence of five blits, and then recursively rotate each block. Alternately, we could first recursively rotate the blocks and then blit them into place.



Two algorithms for rotating a pixel map.
Black arrows indicate blitting the blocks into place; white arrows indicate recursively rotating the blocks.



The first rotation algorithm (blit then recurse) in action.

a) How many blits does the algorithm perform when n is a power of two?

b) What is your algorithm's running time if a $k*k$ blit takes $O(k^2)$ time?

c) (This last one was on the exam to but you don't do it now: Describe how to modify the algorithm so that it works for arbitrary n, not just powers of two.)

(This problem was 6+2+6 = 14p on the exam.)