Problem 51. (12 points):

This problem tests your understanding of cache conflict misses. Consider the following matrix transpose routine

```
typedef int array[2][2];

void transpose(array dst, array src) {
  int i, j;

  for (i = 0; i < 2; i++) {
    for (j = 0; j < 2; j++) {
      dst[j][i] = src[i][j];
    }
  }
}
```

running on a hypothetical machine with the following properties:

- `sizeof(int) == 4`.
- The `src` array starts at address 0 and the `dst` array starts at address 16 (decimal).
- There is a single L1 cache that is direct mapped and write-allocate, with a block size of 8 bytes.
- Accesses to the `src` and `dst` arrays are the only sources of read and write misses, respectively.

A. Suppose the cache has a total size of 16 data bytes (i.e., the block size times the number of sets is 16 bytes) and that the cache is initially empty. Then for each `row` and `col`, indicate whether each access to `src[row][col]` and `dst[row][col]` is a hit (h) or a miss (m). For example, reading `src[0][0]` is a miss and writing `dst[0][0]` is also a miss.

`dst` array		
	col 0	col 1
row 0	m	
row 1		

`src` array		
	col 0	col 1
row 0	m	
row 1		

B. Repeat part A for a cache with a total size of 32 data bytes.

`dst` array		
	col 0	col 1
row 0	m	
row 1		

`src` array		
	col 0	col 1
row 0	m	
row 1		