Part D:

	row-major access (UArray2)	column-major access (UArray2)	blocked access (UArray2b)
90 degree rotation	5	3	1
180 degree rotation	3	5	1

Explanation for #1s:

The blocked access means that the array is indexed by the blocks that the data is stored in. The reason why this would be the best way to iterate through the array is because the miss rate would be once per block. The locality of the blocked structure allows for elements to in a block to be stored in the same blocks in memory. This means that when the program looks for the first element in the block, the whole block is put into the cache. There will not be a cache miss until the next element is an element not in this block. The degree of rotation does not matter because the data in a block will stay in a single block upon rotation (for the most part).

Explanations for #3s:

For a 180 rotation using row major access and 90 degree column major access, the iteration of the array follows the same way the new array will be stored which results in read to writes not being as bad as the final two options. This is because there only needs to be a stride for either read or write instead of both. In 180 degree row major access, the row values are stored in horizontal increasing memory locations. Therefore, the cache will miss every time that the next value of the array is no longer continuous or when the next row element is in the block to the right in memory (but there is no stride so miss rate for read will be lower). These values will still be stored next to each other in the array that is created (because rows map to rows), so having to read to write will not be a huge deal because the writing memory will be in the same direction (horizontal) as the memory that is read in. For 90 rotation column major, the 90 degree rotation results in row becoming columns and columns becoming rows. Therefore, reading the memory will be done with a stride and will miss when the stride leaves the current block, but writing memory is done in the order of reading. The reason that both of these will be worse than the blocked access is because the miss rate goes up because of the striding that is necessary in order to read columns (striding means there will be less hits for a singular block). Both of these also are not as efficient as blocked access because one row/column will go across multiple blocks, so there will be more misses per row.

Explanations for #5s:

90 degree row major and 180 degree column major have the worst hit rate. This is because there is a stride to read and write memory which results in more misses. They are also worse than blocking due to the fact that rows and columns are stored across multiple blocks. In the 90 degree row major, data in rows are stored in horizontal increasing memory locations. This means that not all of the data for a row is put into the cache with one block (resulting in increased miss rate for a row). Also, to write this row, rows turn into columns. This means when writing, there also needs to be a number of reads for one write. This means that the cache needs to read multiple memory blocks in to read the data, and needs to read multiple different blocks in to write the data, resulting in more cache misses. For 180 degree column major, it is very similar to above, however at first multiple blocks are read in to get a column, and then multiple other blocks are read in to get the row.