CPSC313 - Assignment 4

Student: Evan Louie
Student#: 72210099
CSID: m6d7

Problem 1

Section A

14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT	CT	CT	CT	CT	CT	CT	CT	CT	CI	CI	CI	CO	CO	CO

Section B

Operation	Address	Cache Set	Tag	Hit?	Value
Read	0x0362	4	D	Miss-Conflict	Unkown
Read	0x7BFE	7	1EF	Miss-Cold	Unkown
Read	0x3FC9	1	FF	Miss-Cold	Unkown
Read	0x0361	4	D	Miss-Conflict	Unkown

- Reading 0x0362 will result in a Miss-Conflict due to the current value in cache not being valid. Thus it will have to be fetched from memory
- Reading 0x7BFE will result in a Miss-Cold due to their not currently being a 1EF tag in set 7
- Reading 0x3FC9 will result in a Miss-Cold due to their not currently being a FF tag in set 1
- Reading 0x0361 will result in a hit because after the first Miss-Conflict in the first read, tag D would have been loaded into set 4

	Binary	Tag	Index	Offset
0x0364	0000 0011 0110 0100	000001101	100	010
0x7BFE	0111 1011 1111 1110	111101111	111	110
0x3FC9	0011 1111 1100 1001	011111111	001	001
0x0361	0000 0011 0110 0001	000001101	100	001

Problem 2

Refer to cache.c

Problem 3

NOTE: Assuming cache is layed out contiguously in linear memory.

Section A

```
        Miss-Rate = 25%
        Initial read attempt will cold-miss and fetch information along the next 3 ints in the
        array (due to the 16-byte block size). Because the array is stored in row-major order,
        this will result in correct ordering.

    Section B
    ----------
        Miss-Rate = 100%
        Every read attempt will result in a miss because the array is being accesed in a
        column major order, which is against the way the way the cache will store the array
        (which is row-major).

    Section C
    ----------
        Miss-Rate = 50%
        The function reads sections of the array in a 2x2 square, thus it will always always
        require to fetch new information everytime it goes to the next line. Hence it will
        always miss, hit and repeat.

    Section D
    ----------
        Miss-Rate = 100%
        Just as in section B, the array is being accessed in the the incorrect order;
        increasing the array size will only lead to additional conflicts and required rewrites
        on top of the problems in section b when the cache is full.

    Section E
    ----------
        Miss-Rate = 100%
        Same as section D, increasing/decreasing the array size will not change the fact that
        the array will be read in the cache in the incorrect order. Hence it will stay 100%.

    Section F
    ----------
        Miss-Rate = 100%
        Using a 2-way set associative cache will not help as the array is still being read in
        the incorrect order.  The only way any n-way set associative cache would help in this
        situation was if the it was greater than or equal to a 48-way set associative cache
        because without n being that size; an array of 48x48 being read in always result in
        conflicts in the sets and require rewrites, never managing to take advantage of the
        fact that it stores the next numbers in the column before rewriting.


==============
Hours Spent
==============
    6.5 Hours
```