* frame lines = size of container in cache

COMP5201 - Assignment 4
Sarabraj Singh - 29473858

Question 1

→ Assumptions
- ↳ each instruction is 4 bytes
- ↳ each loop has 8 instructions
- ↳ Direct-mapped cache w̄ 16-byte lines ∴ byte offset is 4 bits ($2^4 = 16$)
- ↳ determine cache misses in the first 16 iterations of the loop.
- ↳ each frame ∴ can hold 4 ops (16-byte line ÷ 4-bytes per op)
- ↳ we only care about ops that deal w̄ memory accesses. cache lookups only occur w̄ memory references.
- ↳ PC (program counter) increments after every op
- ↳ important → each iteration of the loop only loads 1 integer

⤳ instruction

Memory Address	Memory Content
3a5c50	Load
3a5c54	Op1
3a5c58	Op2
3a5c5c	Op3
3a5c60	Op4
3a5c64	Op5
3a5c68	Op6
3a5c6c	Op7

⤳ contiguous integers

	Memory Address	Memory Content
block 0	4b6d30	Int1
	4b6d34	Int2
	4b6d38	Int3
	4b6d3c	Int4
block 1	4b6d40	Int5
	4b6d44	Int6
	4b6d48	Int7
	4b6d4c	Int8
block 2	4b6d50	Int9
	4b6d54	Int10
	4b6d58	Int11
	4b6d5c	Int12
block 3	4b6d60	Int13
	4b6d64	Int14
	4b6d68	Int15
	4b6d6c	Int16

⤳ frame index = memory block # (mod) # of frames

Frame Index	Cache Content at Iteration-Load-Time
0	
1	
2	
3	1st iter – [3a5c50 to 3a5c5c]
4	5th iter – [4b6d40 to 4b6d4c]
5	1st iter – [3a5c50 to 3a5c5c] (9th, 10th, 11th, 12th iter – go between loading [3a5c50 to 3a5c5c] and [4b6d50 to 4b6d5c]
6	1st iter – [3a5c60 to 3a5c6c] (13th, 14th, 15th, 16th iter – go between loading [3a5c60 to 3a5c6c] and [4b6d60 to 4b6d6c]
7	
8	
9	
A	
B	
C	
D	
E	
F	

→ byte offset is 4 bits (or 1 hexit)

→ there are 16 or (2^4) frames in this D-cache ∴ the cache index is also 4 bits OR 1 hexit

→ therefore the tag index is 8 bits or 2 hexits

tag	index	offset
← 8 →	← 4 →	← 4 →

$\rightsquigarrow$ FI = frame index in the cache.

Iter:	Integer Wanted:	Expected Cache FI:	Action:	Misses:	Sum of Misses:	
1	4b6d30	3	Nothing in cache. Load [3a5c50->3a5c5c] into FI[5], [3a5c60->3a5c6c] into FI[6] and [4b6d30->4b6d3c] into FI[3]	3	3	a)
2	4b6d34	3	[4b6d34] exists at FI[3] in cache. No miss	0		
3	4b6d38	3	[4b6d38] exists at FI[3] in cache. No miss	0		
4	4b6d3c	3	[4b6d3c] exists at FI[3] in cache. No miss	0		
5	4b6d40	4	Expected [4b6d40] at FI[4] in cache. Not there. Miss and load [4b6d40->4b6d4c] into cache at FI[4]	1	1	b)
6	4b6d44	4	[4b6d44] exists at FI[4]. No miss	0		
7	4b6d48	4	[4b6d48] exists at FI[4]. No miss	0		
8	4b6d4c	4	[4b6d4c] exists at FI[4]. No miss	0		
9	4b6d50	5	Expected [4b6d50] at FI[5]. Load [4b6d50->4b6d5c] into FI[5]. Miss	1	7	c)
10	4b6d54	5	Expected [3a5c50->3a5c5c] at FI[5], load. Expected [4b6d54] at FI[5]. Load [4b6d50->4b6d5c] into FI[5]	2		
11	4b6d58	5	Expected [3a5c50->3a5c5c] at FI[5], load. Expected [4b6d58] at FI[5]. Load [4b6d50->4b6d5c] into FI[5]	2		
12	4b6d5c	5	Expected [3a5c50->3a5c5c] at FI[5], load. Expected [4b6d5c] at FI[5]. Load [4b6d50->4b6d5c] into FI[5]	2		
13	4b6d60	6	Expected [3a5c50->3a5c5c] at FI[5], load. Expected [4b6d60] at FI[6]. Load [4b6d60->4b6d56] into FI[6]. Miss	2	8	d)
14	4b6d64	6	Expected [3a5c60->3a5c6c] at FI[6], load. Expected [4b6d64] at FI[6]. Load [4b6d60->4b6d6c] into FI[6]	2		
15	4b6d68	6	Expected [3a5c60->3a5c6c] at FI[6], load. Expected [4b6d68] at FI[6]. Load [4b6d60->4b6d6c] into FI[6]	2		
16	4b6d6c	6	Expected [3a5c60->3a5c6c] at FI[6], load. Expected [4b6d6c] at FI[6]. Load [4b6d60->4b6d6c] into FI[6]	2		