1.) a.) Spatial locality in instruction fetching b.) Temporal locality in data accesses
Submitted assembly .s files

2.) P&H (Textbook) 5.2.1, 5.2.2, 5.2.3, 5.2.6
5.2 Caches are important to providing a high-performance memory hierarchy to processors. Below is a list of 32-bit memory address references, given as word addresses.

3, 180, 43, 2, 191, 88, 190, 14, 181, 44, 186, 253

5.2.1 [10] <§5.3> For each of these references, identify the binary address, the tag, and the index given a direct-mapped cache with 16 one-word blocks. Also list if each reference is a hit or a miss, assuming the cache is initially empty.

Address	Index	Tag	Hit / Miss
0000 0000 0000 0000 0000 0000 0000 0011	0011	0000 0000 0000 0000 0000 0000 0000	M
0000 0000 0000 0000 0000 0000 1011 0100	0100	0000 0000 0000 0000 0000 0000 1011	M
0000 0000 0000 0000 0000 0000 0010 1011	1011	0000 0000 0000 0000 0000 0000 0010	M
0000 0000 0000 0000 0000 0000 0000 0010	0010	0000 0000 0000 0000 0000 0000 0000	M
0000 0000 0000 0000 0000 0000 1011 1111	1111	0000 0000 0000 0000 0000 0000 1011	M
0000 0000 0000 0000 0000 0000 0101 1000	1000	0000 0000 0000 0000 0000 0000 0101	M
0000 0000 0000 0000 0000 0000 1011 1110	1110	0000 0000 0000 0000 0000 0000 1010	M
0000 0000 0000 0000 0000 0000 0000 1110	1110	0000 0000 0000 0000 0000 0000 0000	M
0000 0000 0000 0000 0000 0000 1011 0101	0101	0000 0000 0000 0000 0000 0000 1011	M
0000 0000 0000 0000 0000 0000 0010 1100	1100	0000 0000 0000 0000 0000 0000 0010	M
0000 0000 0000 0000 0000 0000 1011 1010	1010	0000 0000 0000 0000 0000 0000 1011	M
0000 0000 0000 0000 0000 0000 1111 1101	1101	0000 0000 0000 0000 0000 0000 1111	M

5.2.2 [10] <§5.3> For each of these references, identify the binary address, the tag, and the index given a direct-mapped cache with two-word blocks and a total size of 8 blocks. Also list if each reference is a hit or a miss, assuming the cache is initially empty.

Address	Index	Offset	Tag	Hit / Miss
0000 0000 0000 0000 0000 0000 0000 0011	001	1	0000 0000 0000 0000 0000 0000 0000	M
0000 0000 0000 0000 0000 0000 1011 0100	010	0	0000 0000 0000 0000 0000 0000 1011	M
0000 0000 0000 0000 0000 0000 0010 1011	101	1	0000 0000 0000 0000 0000 0000 0010	M
0000 0000 0000 0000 0000 0000 0000 0010	001	0	0000 0000 0000 0000 0000 0000 0000	H – 3 loads 2
0000 0000 0000 0000 0000 0000 1011 1111	111	1	0000 0000 0000 0000 0000 0000 1011	M
0000 0000 0000 0000 0000 0000 0101 1000	100	0	0000 0000 0000 0000 0000 0000 0101	M
0000 0000 0000 0000 0000 0000 1011 1110	111	0	0000 0000 0000 0000 0000 0000 1010	H – 191 loads 190
0000 0000 0000 0000 0000 0000 0000 1110	111	0	0000 0000 0000 0000 0000 0000 0000	M
0000 0000 0000 0000 0000 0000 1011 0101	010	1	0000 0000 0000 0000 0000 0000 1011	H – 180 loads 181
0000 0000 0000 0000 0000 0000 0010 1100	110	0	0000 0000 0000 0000 0000 0000 0010	M
0000 0000 0000 0000 0000 0000 1011 1010	101	0	0000 0000 0000 0000 0000 0000 1011	M
0000 0000 0000 0000 0000 0000 1111 1101	110	1	0000 0000 0000 0000 0000 0000 1111	M

5.2.3 [20] <§§5.3, 5.4> Optimal design w/ 8 words of data: C1 has 1-word blocks, C2 has 2-word blocks, and C3 has 4-word blocks. Best miss rate? Stall time is 25 cycles. C1 - 2 cycles. C2 - 3 cycles. C3 - 5 cycles Which is the best cache design?

C1 – all miss since all are different address and C1 doesn't take advantage of spatial locality
Hit rate is 0% # of Cycles = 25 * 12 = 300 cycles

C2 – 4 2-word blocks
C2 table: Best Design and hit/miss rate 2 / 12 Hits = 16.66% hit rate # of Cycles = 25 * 10 + 3 * 2 = 256 cycles

Address	Index	Offset	Tag	Hit / Miss
0000 0000 0000 0000 0000 0000 0000 0011	01	1	0000 0000 0000 0000 0000 0000 0000 0	M
0000 0000 0000 0000 0000 0000 1011 0100	10	0	0000 0000 0000 0000 0000 0000 1011 0	M
0000 0000 0000 0000 0000 0000 0010 1011	01	1	0000 0000 0000 0000 0000 0000 0010 1	M
0000 0000 0000 0000 0000 0000 0000 0010	01	0	0000 0000 0000 0000 0000 0000 0000 0	M
0000 0000 0000 0000 0000 0000 1011 1111	11	1	0000 0000 0000 0000 0000 0000 1011 1	M
0000 0000 0000 0000 0000 0000 0101 1000	00	0	0000 0000 0000 0000 0000 0000 0101 1	M
0000 0000 0000 0000 0000 0000 1011 1110	11	0	0000 0000 0000 0000 0000 0000 1010 1	H – 191 loads 190
0000 0000 0000 0000 0000 0000 0000 1110	11	0	0000 0000 0000 0000 0000 0000 0000 1	M
0000 0000 0000 0000 0000 0000 1011 0101	10	1	0000 0000 0000 0000 0000 0000 1011 0	H – 180 loads 181
0000 0000 0000 0000 0000 0000 0010 1100	10	0	0000 0000 0000 0000 0000 0000 0010 1	M
0000 0000 0000 0000 0000 0000 1011 1010	01	0	0000 0000 0000 0000 0000 0000 1011 1	M
0000 0000 0000 0000 0000 0000 1111 1101	10	1	0000 0000 0000 0000 0000 0000 1111 1	M

C3 – 2 4-word blocks Too small index size so many overwrites
C2 table: 1/12 Hits or 8.33% hit rate # of Cycles = 25 * 11 + 5 = 280 cycles

Address	Index	Offset	Tag	Hit / Miss
0000 0000 0000 0000 0000 0000 0000 0011	0	11	0000 0000 0000 0000 0000 0000 0000 0	M
0000 0000 0000 0000 0000 0000 1011 0100	1	00	0000 0000 0000 0000 0000 0000 1011 0	M
0000 0000 0000 0000 0000 0000 0010 1011	0	11	0000 0000 0000 0000 0000 0000 0010 1	M
0000 0000 0000 0000 0000 0000 0000 0010	0	10	0000 0000 0000 0000 0000 0000 0000 0	M
0000 0000 0000 0000 0000 0000 1011 1111	1	11	0000 0000 0000 0000 0000 0000 1011 1	M
0000 0000 0000 0000 0000 0000 0101 1000	0	00	0000 0000 0000 0000 0000 0000 0101 1	M
0000 0000 0000 0000 0000 0000 1011 1110	1	10	0000 0000 0000 0000 0000 0000 1010 1	H – 191 loads 190
0000 0000 0000 0000 0000 0000 0000 1110	1	10	0000 0000 0000 0000 0000 0000 0000 1	M
0000 0000 0000 0000 0000 0000 1011 0101	1	01	0000 0000 0000 0000 0000 0000 1011 0	M
0000 0000 0000 0000 0000 0000 0010 1100	1	00	0000 0000 0000 0000 0000 0000 0010 1	M
0000 0000 0000 0000 0000 0000 1011 1010	0	10	0000 0000 0000 0000 0000 0000 1011 1	M
0000 0000 0000 0000 0000 0000 1111 1101	1	01	0000 0000 0000 0000 0000 0000 1111 1	M

5.2.6 [15] <§5.3> The formula shown in Section 5.3 shows the typical method to index a direct-mapped cache, specifically (Block address) modulo (Number of blocks in the cache).
Assuming a 32-bit address and 1024 blocks in the cache, consider a different indexing function, specifically

(Block address[31:27] XOR Block address[26:22]).

Is it possible to use this to index a direct-mapped cache? If so, explain why and discuss any changes that might need to be made to the cache. If it is not possible, explain why.

This isn't possible to use because there are only 5 bits in the index generated by the function above.
If the cache has 1024 blocks then the index needs to be 10 bits long.