[Group Activity] Use cache simulator module to evaluate the cache performance. In your sorting code,

    A. Keep the cache size to be a constant(1KB preferred) value. Vary the block size as [4, 16, 64, 256]. Consequently change the number of blocks as well to match cache size. Calculate the Read Hit rate for data cache. Graph your results only for bubble sort. Use the medium random array.

*Input string:*

55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0;

*Block Size* – 4 bytes

*Number of Blocks* – 256

*Associativity* – 1

*Dump Statistics:*

read\_hits: 48186 – 45160 = 3024

read\_accesses: 48268 – 45187 = 3081

*Block Size* – 16 bytes

*Number of Blocks* – 64

*Associativity* – 1

*Dump Statistics:*

read\_hits: 88337 – 85271 = 3065

read\_accesses: 88359 – 85278 = 3081

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 1

*Dump Statistics:*

read\_hits: 68408 – 65331 = 3076

read\_accesses: 68415 – 65334 = 3081

*Block Size* – 256 bytes

*Number of Blocks* – 4

*Associativity* – 1

*Dump Statistics:*

read\_hits: 154514 – 151434 = 3079

read\_accesses: 154517 – 151436 = 3081

    B. Keep the cache size to be a constant (1KB preferred) value. Vary the cache Associativity as [Direct mapped, 2-way set associative, 4-way set associative, fully associative]. Calculate the Read Hit rate for data cache. Graph your results only for bubble sort. Use 64 as block size. Use the medium random array.

*Input string:*

55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0;

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 1

*Dump Statistics:*

read\_hits: 68400 – 65323 = 3076

read\_accesses: 68407 – 65326 = 3081

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 2

*Dump Statistics:*

read\_hits: 54650 – 51573 = 3076

read\_accesses: 54657 – 51576 = 3081

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 4

*Dump Statistics:*

read\_hits: 124431 – 121354 = 3077

read\_accesses: 124438 – 121357 = 3081

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 8

*Dump Statistics:*

read\_hits: 77610 – 74533 = 3076

read\_accesses: 77617 – 74536 = 3081

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 16

*Dump Statistics:*

read\_hits: 93400 – 90323 = 3076

read\_accesses: 93407 – 90326 = 3081

    C. Keep the cache size to be a constant (1KB preferred) value. Use all 3 sorting algorithm on 2 arrays (small unsorted and medium random). Vary either Block size or Associativity as in either activity A or activity B for data cache. Graph your results.

*Input string:*

43 36 44 50 59 17 22 30 57 51;55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0;

*Block Size* – 4 bytes

*Number of Blocks* – 256

*Associativity* – 1

*Bubble Sort Statistics:*

read\_hits: 68 3024

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 0 0

read\_accesses: 24 24

*Selection Sort Statistics:*

read\_hits: 197 6269

read\_accesses: 242 6406

*Block Size* – 4 bytes

*Number of Blocks* – 256

*Associativity* – 2

*Bubble Sort Statistics:*

read\_hits: 68 3024

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 0 0

read\_accesses: 24 24

*Selection Sort Statistics:*

read\_hits: 197 6269

read\_accesses: 242 6406

*Block Size* – 4 bytes

*Number of Blocks* – 256

*Associativity* – 4

*Bubble Sort Statistics:*

read\_hits: 68 3024

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 0 0

read\_accesses: 24 24

*Selection Sort Statistics:*

read\_hits: 197 6269

read\_accesses: 242 6406

*Block Size* – 16 bytes

*Number of Blocks* – 64

*Associativity* – 1

*Bubble Sort Statistics:*

read\_hits: 75 3065

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 17

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 228 6368

read\_accesses: 242 6406

*Block Size* – 16 bytes

*Number of Blocks* – 64

*Associativity* – 2

*Bubble Sort Statistics:*

read\_hits: 75 3065

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 17

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 197 6368

read\_accesses: 242 6406

*Block Size* – 16 bytes

*Number of Blocks* – 64

*Associativity* – 4

*Bubble Sort Statistics:*

read\_hits: 75 3065

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 17

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 197 6368

read\_accesses: 242 6406

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 1

*Bubble Sort Statistics:*

read\_hits: 77 3076

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 21

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 236 6394

read\_accesses: 242 6406

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 2

*Bubble Sort Statistics:*

read\_hits: 77 3076

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 21

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 236 6394

read\_accesses: 242 6406

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 4

*Bubble Sort Statistics:*

read\_hits: 77 3076

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 21

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 236 6394

read\_accesses: 242 6406

*Block Size* – 64 bytes

*Number of Blocks* – 16

*Associativity* – 16

*Bubble Sort Statistics:*

read\_hits: 77 3076

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 21

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 236 6394

read\_accesses: 242 6406

*Block Size* – 256 bytes

*Number of Blocks* – 4

*Associativity* – 1

*Bubble Sort Statistics:*

read\_hits: 77 3079

read\_accesses: 79 3081

*Insertion Sort Statistics:*

read\_hits: 21

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 237 6401

read\_accesses: 242 6406

*Block Size* – 256 bytes

*Number of Blocks* – 4

*Associativity* – 2

*Bubble Sort Statistics:*

read\_hits: 77

read\_accesses: 79

*Insertion Sort Statistics:*

read\_hits: 21

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 237

read\_accesses: 242

*Block Size* – 256 bytes

*Number of Blocks* – 4

*Associativity* – 4

*Bubble Sort Statistics:*

read\_hits: 77

read\_accesses: 79

*Insertion Sort Statistics:*

read\_hits: 21

read\_accesses: 24

*Selection Sort Statistics:*

read\_hits: 237

read\_accesses: 242

2. [Group activity] Write a Report on what do you infer from the graphs. What did you expect? Does your expectation match the obtained plot? Submission link available outside  
  
3. [Group activity] Prepare a 5-7mins minute presentation on what you think is uniqueness of your sorting program or the results you obtained. Submission link available outside  
  
4. [Individual Journal] Do a write-up (journal entry) on what you enjoyed the most out of Project 1B. 

Please let me know if you guys have any questions on this.

All of Project 1B and Project 1A reflection are due Oct1. 9.00AM