**To Run Program**

The program is written in Java and takes a single input argument. When entering the input please surround it with quotes so as to convey to Java that it is one argument and not multiple. To run the program, perform the following;

1. Open windows terminal and cd to the proper file directory that holds the .java source file.
2. Once in the proper directory, run “javac sim.java” to compile the program.
3. Then run “java sim” followed by the argument in quotation marks.
   1. For Example;
      1. java sim “./SIM 32768 8 1 1 /home/TRACES/MCF.t”
4. It should now run and after approx. 30 - 50 seconds it should spit out the outputs.

Thank you!

**Benchmarks**

1. 4-Way Associative Cache, Write-Back, LRU;
   1. MiniFE
      1. 8KB
         1. The total miss ratio for L1 cache: 0.16162261
         2. The number of writes to memory: 162341
         3. The number of reads from memory: 3258520
      2. 16KB
         1. The total miss ratio for L1 cache: 0.22649898
         2. The number of writes to memory: 64
         3. The number of reads from memory: 157725
      3. 32KB
         1. The total miss ratio for L1 cache: 0.17955509
         2. The number of writes to memory: 178
         3. The number of reads from memory: 600274
      4. 64KB
         1. The total miss ratio for L1 cache: 0.15883686
         2. The number of writes to memory: 422
         3. The number of reads from memory: 751406
      5. 128KB
         1. The total miss ratio for L1 cache: 0.16088048
         2. The number of writes to memory: 910
         3. The number of reads from memory: 747664
   2. XSBench
      1. 8KB
         1. The total miss ratio for L1 cache: 0.15881747
         2. The number of writes to memory: 30
         3. The number of reads from memory: 1354881
      2. 16KB
         1. The total miss ratio for L1 cache: 0.1559301
         2. The number of writes to memory: 50
         3. The number of reads from memory: 1679231
      3. 32KB
         1. The total miss ratio for L1 cache: 0.13202854
         2. The number of writes to memory: 62
         3. The number of reads from memory: 2511808
      4. 64KB
         1. The total miss ratio for L1 cache: 0.12623516
         2. The number of writes to memory: 64
         3. The number of reads from memory: 2734704
      5. 128KB
         1. The total miss ratio for L1 cache: 0.12386875
         2. The number of writes to memory: 80
         3. The number of reads from memory: 2834922
2. 4-Way Associative Cache, Write-Through, LRU;
   1. MiniFE
      1. 8KB
         1. The total miss ratio for L1 cache: 0.23183842
         2. The number of writes to memory: 253228
         3. The number of reads from memory: 113734
      2. 16KB
         1. The total miss ratio for L1 cache: 0.22649898
         2. The number of writes to memory: 285460
         3. The number of reads from memory: 157725
      3. 32KB
         1. The total miss ratio for L1 cache: 0.17955509
         2. The number of writes to memory: 345720
         3. The number of reads from memory: 600274
      4. 64KB
         1. The total miss ratio for L1 cache: 0.15883686
         2. The number of writes to memory: 549046
         3. The number of reads from memory: 751406
      5. 128KB
         1. The total miss ratio for L1 cache: 0.16088048
         2. The number of writes to memory: 484330
         3. The number of reads from memory: 747664
   2. XSBench
      1. 8KB
         1. The total miss ratio for L1 cache: 0.15881747
         2. The number of writes to memory: 10026770
         3. The number of reads from memory: 1354881
      2. 16KB
         1. The total miss ratio for L1 cache: 0.1559301
         2. The number of writes to memory: 9218486
         3. The number of reads from memory: 1679231
      3. 32KB
         1. The total miss ratio for L1 cache: 0.13202854
         2. The number of writes to memory: 9937096
         3. The number of reads from memory: 2511808
      4. 64KB
         1. The total miss ratio for L1 cache: 0.12623516
         2. The number of writes to memory: 10026910
         3. The number of reads from memory: 2734704
      5. 128KB
         1. The total miss ratio for L1 cache: 0.12386875
         2. The number of writes to memory: 10026906
         3. The number of reads from memory: 2834922

Here we can see that XSBench has an overall lower miss rate while maintaining high writes to memory and high reads from memory. While MiniFE has a higher miss rate and lower writes to memory and lower reads from memory.

1. 32KB Cache, Write-Back, LRU;
   1. MiniFE
      1. Direct-Mapped
         1. The total miss ratio for L1 cache: 0.018861664
         2. The number of writes to memory: 141622
         3. The number of reads from memory: 1948804
      2. 2 – Way
         1. The total miss ratio for L1 cache: 0.21004182
         2. The number of writes to memory: 422
         3. The number of reads from memory: 331859
      3. 4 – Way
         1. The total miss ratio for L1 cache: 0.17955509
         2. The number of writes to memory: 178
         3. The number of reads from memory: 600274
      4. 8 – Way
         1. The total miss ratio for L1 cache: 0.168702
         2. The number of writes to memory: 64
         3. The number of reads from memory: 679245
      5. 16 – Way
         1. The total miss ratio for L1 cache: 0.16504046
         2. The number of writes to memory: 20
         3. The number of reads from memory: 712411
      6. 32 – Way
         1. The total miss ratio for L1 cache: 0.15947539
         2. The number of writes to memory: 8
         3. The number of reads from memory: 746181
      7. 64 – Way
         1. The total miss ratio for L1 cache: 0.15880147
         2. The number of writes to memory: 4
         3. The number of reads from memory: 750184
   2. XSBench
      1. Direct-Mapped
         1. The total miss ratio for L1 cache: 0.033523325
         2. The number of writes to memory: 73190
         3. The number of reads from memory: 6791770
      2. 2 – Way
         1. The total miss ratio for L1 cache: 0.23314568
         2. The number of writes to memory: 64
         3. The number of reads from memory: 493732
      3. 4 – Way
         1. The total miss ratio for L1 cache: 0.13202854
         2. The number of writes to memory: 62
         3. The number of reads from memory: 2511808
      4. 8 – Way
         1. The total miss ratio for L1 cache: 0.12983476
         2. The number of writes to memory: 50
         3. The number of reads from memory: 2582261
      5. 16 – Way
         1. The total miss ratio for L1 cache: 0.12667024
         2. The number of writes to memory: 30
         3. The number of reads from memory: 2716278
      6. 32 – Way
         1. The total miss ratio for L1 cache: 0.12706321
         2. The number of writes to memory: 16
         3. The number of reads from memory: 2699636
      7. 64 – Way
         1. The total miss ratio for L1 cache: 0.1257326
         2. The number of writes to memory: 6
         3. The number of reads from memory: 2755987

We can clearly see that as the cache association gets larger, the miss rate lowers and so does the number of writes to memory. Generally, we see the number of reads increase. However, we can also that there are diminishing returns here. When the cache went from 2 – way to 4 - way associative, we saw the greatest drop in miss rate. Which gives us the best return. Past 4 – way associative, the miss rate still lowers, but by a much smaller factor. Between MiniFE and XSBench, we can see that XSBench had a higher hit rate. Between 2-way and 4-way associative, we can also see the largest jump in reads from memory and the smallest decrease in writes to memory.

(Excluded Direct Mapped because it seems my algorithm may have fallen apart there)

(Excluded Direct Mapped because the number was too large, the data would not scale)

(Excluded Direct Mapped because it seems my algorithm may have fallen apart there)

1. 4-Way Associative Cache, Write-Back, FIFO;
   1. MiniFE
      1. 8KB
         1. The total miss ratio for L1 cache: 0.19487108
         2. The number of writes to memory: 119834
         3. The number of reads from memory: 843484
      2. 16KB
         1. The total miss ratio for L1 cache: 0.12962477
         2. The number of writes to memory: 97885
         3. The number of reads from memory: 915126
      3. 32KB
         1. The total miss ratio for L1 cache: 0.10458155
         2. The number of writes to memory: 86279
         3. The number of reads from memory: 942691
      4. 64KB
         1. The total miss ratio for L1 cache: 0.08804644
         2. The number of writes to memory: 79896
         3. The number of reads from memory: 960978
      5. 128KB
         1. The total miss ratio for L1 cache: 0.07544666
         2. The number of writes to memory: 70811
         3. The number of reads from memory: 974402
   2. XSBench
      1. 8KB
         1. The total miss ratio for L1 cache: 0.22930636
         2. The number of writes to memory: 609685
         3. The number of reads from memory: 2942886
      2. 16KB
         1. The total miss ratio for L1 cache: 0.18328631
         2. The number of writes to memory: 251888
         3. The number of reads from memory: 3136726
      3. 32KB
         1. The total miss ratio for L1 cache: 0.15977433
         2. The number of writes to memory: 145769
         3. The number of reads from memory: 3264753
      4. 64KB
         1. The total miss ratio for L1 cache: 0.14406247
         2. The number of writes to memory: 71654
         3. The number of reads from memory: 3344199
      5. 128KB
         1. The total miss ratio for L1 cache: 0.1340933
         2. The number of writes to memory: 36595
         3. The number of reads from memory: 3395885

Here, you can see the miss ratio lowers as our cache gets larger, but we never get the miss rate lower than when we used LRU replacement because LRU is a more efficient replacement policy.

.