Your assignment is to evaluate and recommend a configuration for the L1 data cache system (DL1) of a processor. You will be evaluating different cache configurations using the simulator set up in the last project lab. In particular you will be using the sim-cache simulator. Papers and descriptions were referenced in the last lab.

A general search program is provided for you to conduct this evaluation. This program is available under:

http://cs.nyu.edu/courses/fall11/CSCI-GA.2233-001/labs/lab2/msort.cpp

Download the program and compile as follows:

```bash
sslittle-na-sstrix-gcc -O3 msort.cpp -o msort
```

This program sorts 100K random integers based on either quicksort (-aq) or mergesort (-am).

Familiarize yourself with the output of the sim-cache simulator by running:

```bash
sim-cache -./msort -am
```

You will now change various DL1 parameters.

Sim-cache allows you to change the DL1 configuration. The cache config parameter <config> has the following format:

```plaintext
<name>:<nsets>:<bsize>:<assoc>:<repl>
```

- `<name>` - name of the cache being defined
- `<nsets>` - number of sets in the cache
- `<bsize>` - block size of the cache
- `<assoc>` - associativity of the cache
- `<repl>` - block replacement strategy, 'l'-LRU, 'f'-FIFO, 'r'-random

Example : sim-cache -cache:dl1 dl1:64:64:2:f ./msort -aq

runs quicksort and sets the DL1 cache to sets=64, blocksize=64, 2 way associative and FIFO. Remember that cachesize = numsets * blocksize * associativity, so in this example the DL1 cache size = 64*64*2=8KB.

PART1:

Pick configurations (N:32:4:l) with N resulting in DL1 cache sizes of 2K, 4K, 8K, 16K and 32K. For each size, manually run the simulation once for quicksort (-aq) and once for mergesort (-am).

Examine the output of the simulation, extract the relevant performance numbers (ins, misses, writebacks, ..) and determine the CPI based on the assumption that all instructions take 1 cycles
with the exception of Load/Stores that hit in the L1 take 2 cycles, Load/Stores hitting in the L2 cache take 16 cycles and Load/Stores missing in the L2 take 64 cycles. Provide all used counters (should be only a few) in a table. Make sure you understand the relationship of misses and writeback at L1 level to accesses to L2 and L2 to memory. Enter the counters used in a spreadsheet (col=counters, rows=configuration) and compute the CPI per configuration. Submit a printout of your table from the spreadsheet. Show the CPI computation formula on your printout.

PART2:

You are going to examine the impact of the cache organization for the same cache size DL1=4KB. We are interested in the dl1.miss_rate going forward, as it indicates the efficiency of the cache organization.

I provide the following script:

```
http://cs.nyu.edu/courses/fall11/CSCI-GA.2233-001/labs/lab2/runit.sh
```

This runs all different configurations that you have to add to the script. You must replace the configurations with the desired combinations, each combination is a “<nsets>:<bsize>:<assoc>” specification. Create combinations that yield 4KB DL1 caches sizes and that explore the following parameter space: blocksize={32,64} and associativity={1,2,4,8}, so you must compute the proper numset.

This scripts runs all three replacement policies (LRU (l), FIFO (f), RANDOM (r) see sim-cache output) and runs quicksort and mergesort with 100K elements and bubblesort with 5K elements due to its very long runtime (O(n^2) complexity) vs. (O(n*log(n)) for quick and mergesort) and reports the missrates in a table similar to this:

```
CONF     QS     MS     BS
64:32:1:f 0.0198 0.0212 0.0333
Etc.
```

Present your data in a table with the following format (nicely readable appreciated). Putting this in a spreadsheet and graphically analyzing it might also help.

Now repeat the whole experiment with 8K DL1 cache size, by modifying the script again.

Analyze the data and write a research report (min=2, max=4 page @10pt font) that describes the problem, summarizes your findings and that makes recommendations on which configurations to favor and why for the 4K DL1 only. Try to explain the behavior you are seeing. For this you need to understand how these algorithms work and if the code is not clear, wikipedia has good writeups on them. At the end, in one paragraph, make an argument whether moving to an 8K DL1 cache makes sense or not based on these limited test cases. Treat this as if you were writing a paper.

You must hand in your Table and the research report.