CSCI-UA.0201-001/2
Computer Systems Organization
Lab 2: Cache Lab

Description

In this lab you will learn how to tweak C code to make it cache friendly. By cache friendly we mean producing the least amount of cache misses.
First download the following file into your VM.

http://cs.nyu.edu/courses/spring15/CSCI-UA.0201-002/cachelab.tar

Then:

- tar -xvf cachelab.tar
- You will get a directory called cache-lab that contains the following files:
    - **run**: a small shell script that you use to compile and run your program. If your program is called refcode.c then command line to compile, execute and get cache miss rates measurements is: *./run refcode*
    - **refcode.c**: a source file that contains naïve implementation of the functions you need to optimize.

For this lab you need to tweak the functions level_1( ), level_2( ), level_3( ), and level_4( ) to make them produce lower number of cache misses.
NOTE: level_4( ) is for honor students only.

Notes

- Do not modify the code where the comments state so.
- You can add any variables or data structure you want, but you are not allowed to remove/modify available variables or data structures.
- Feel free to modify the functions level_x() where x is 1, 2, 3, and 4 under the conditions mentioned above.
- Do not use recursion.
- Always remember that cache friendly means locality of data access, both temporal and spatial.
- You must friend understand what each function does. After that, you are free to change the way the function does its job, as long as the final outcome is correct.

Report

Write a report that explains the following. For each level you solved, provide:

- A bulleted list (do not write stories just short bulleted list) of each cache unfriendly pattern you found. At each bullet specific two sub-bullets:
    - What is the cache unfriendly-pattern?
    - How did you deal with it?

Hand-in instructions

- make another copy of refcode.c and call it optimized.c
- You do your magic in optimized.c (check its results using *./ref optimized*)
- Put the report and optimized.c in a zip file called: lastname.firstname.zip where lastname is your last name and firstname is your first name.
- Send your file to our grader and put your name as a comment in at the top of optimized.c

How will we grade this?

- The main measure of success is to reduce the number of read misses for level 1 data cache, shown in the results under the column d1mr.
- We well compare the d1mr of the naïve implementation with the one you submitted in optimized.c. Basically we will calculate d1mr(optimized)/d1mr(refcode).
- Each level is worth of 10 points, calculated as follows:
    - ratio 0.9 or higher: zero
    - ratio 0.6 to less than 0.9: 7
    - ration less than 0.6: 10
- 10 points for the report.

Enjoy!