CS 378 Research in Computer Architecture — Spring 2019	Programming Assignment #2
Cache Replacement	Due February 25th, 2019

In this assignment you will write code for cache replacement policies, and you will compare their accuracy and performance using a microarchitectural simulator. We will provide the microarchitectural simulator along with the interfaces that you will need to implement your replacement policies. This assignment does not require you to write a lot of code, but does require you to understand different concepts in caching, and to carefully evaluate your designs.

1 Your Assignment

The first part of your assignment is to implement two cache replacement policies.

- `Simulate OPT`: The optimal caching algorithm (OPT, Belady's algorithm) evicts the line reused furthest in the future. You will have to simulate the OPT policy, given a history of cache accesses.
- `Your Design`: You will also have to create your own cache replacement policy, which could be an implementation of a policy we've discussed in class (BIP, BRRIP, etc.), a new idea, or an extension of a policy we've discussed in class.

The second part of your assignment is to run experiments with your replacement policies, to make sure you are simulating OPT correctly as well as explore different design choices for your own design.

The third part of your assignment is to describe what you've done, report your results and draw relevant conclusions from your experiments.

2 Details

2.1 Simulator

Same as last assignment, we will use the ChampSim simulator. ChampSim is a trace-based microarchitectural simulator that simulates the effect of executing a program on a software CPU model. ChampSim allows you to observe a variety of useful statistics about the underlying hardware's performance. For this assignment, you will measure the replacement policy's MPKI (Misses Per Kilo Instruction) and its IPC (# Instructions executed Per Cycle). The `README` provides the instructions to compile and run the simulator.

2.2 Cache Access Trace

We have provided a file containing a list of all the cache references, one full memory address per line. The assignment code will open the file for you, and the `README` contains some helpful information about how to addresses from the file. Each line in the file corresponds to a call to `llc_update_replacement`.

2.3 Benchmarks

We have provided a set of 10 benchmarks for this assignment. For each benchmark, we have provided a trace of 1 billion instructions from a representative region of the program (large programs typically run for billions of instructions, which is too slow to simulate on microarchitectural simulator). For this assignment, you are required to execute each trace for 100 million instructions. This time around, we've given you access to the cluster, which distributes your simulations to dedicated machines in parallel, meaning execution times will be shorter ($\sim$5-15 minutes to complete all simulations).

2.4 Report

You should write a report that describes your experiments, the results you got from these experiments (relative to LRU or another baseline) and your conclusions from the results. If you've experimented with design points that were not described in the assignment, please include a description of those with an explanation for why you chose to do those experiments.

3 What To Turn In

Please submit your replacement policies' code and your report in a .zip file as a Private Note on Piazza.

4 Grading

Your grade will be based on a few principles:

- `Implementation:` Your code successfully simulates OPT and your own replacement policy noticeably differs from LRU.
- `Experiments:` You showed some relevant characteristic of your replacement policy using empirical results.
- `Presentation:` Your experiments and analysis are concise and logically displayed.