Name: _____

Name: _____

This section will be unlike any other section in 232. It will be more like a physics lab. You're given a problem to solve and it's up to you to devise a procedure to solve it. You will write down the steps and turn in this writeup at the end of the section to get credit for participation. First, write down your names at the top of the page.

Rules and Regulations

1. Please work in groups of 2.
2. SSH to a machine specified by the TA. The command is `ssh -Y csil-coreX.cs.uiuc.edu`, where `X` indicates the number of the machine we will assign you.
3. Describe your approach in the space provided after each step. Be succinct.
4. We are assuming that you've already tried the VTune tutorial. This handout makes some references to the information in the tutorial.

Problem

Your task is to optimize a simple matrix multiplication code. This should be similar to MP6, but in this case, instead of examining performance under a variety of hardware parameters, you will be looking at code performance on actual hardware. You will use VTune to determine which part of the code needs to be optimized and to verify that your optimizations do indeed improve performance.

Tasks

1. The starting code is provided by us. Copy it to your working directory. Use `make` to compile it. Note that the code is compiled with `O2` flag, so it's already optimized by the compiler, but running it will still take about 10 seconds.

```
% wget http://www.cs.uiuc.edu/class/cs232/section/Discussion12/Makefile
% wget http://www.cs.uiuc.edu/class/cs232/section/Discussion12/matrix.c
% make
% ./matrix
```

Study the code and try to predict how much time each part will take. Write down your predictions.

2. Run the **First Use Wizard** to validate your predictions from the previous step. Sections 2a and 2b from the VTune tutorial explain how to run the wizard. Note any interesting observations below.

3. Looking at the code, how much space does matrix A take up? How large would the L1 cache need to be to hold all three matrices? Is that a reasonable size for the L1 cache?

4. What kinds of metrics would you want to collect to determine the performance of the L1 and the L2 caches?

5. Section 4 of the tutorial describes how to use the **Sampling Wizard**. More specifically, it explains how to collect various types of performance metrics. Select some interesting metrics and collect them for the given code. Write down your results.

6. Based on the results you collected, describe what kind of optimization you'd like to implement first.

7. Implement the change in the `matrix.c` file. Rerun the **Sampling Wizard** to see if the metrics have improved. Summarize the result.

8. Lather, rinse, repeat. You can use the extra page if you need more space to describe your results. Try to find an optimization that improves performance as much as possible, as well as an optimization that improves performance significantly, while minimizing code changes.

9. Calculate roughly how long a miss in the L1 cache takes. Make simplifying assumptions, like the change in running time is due to the decreased number of misses.

```
% time ./matrix
```

will give you a running time for the `matrix` program.

10. Copy the final code here. What makes this code better?

Don't forget to turn this paper in after the section.