Requirements:

To compile your code for POWER ISA, you will need a cross-compiler (gcc-powerpc64-linux-gnu) installed on your PC. We recommend using any one of the following options.

Option 1 –
If you have Ubuntu 18.04 (or 20.04) installed on your PC, install the cross compiler using the command below

```
sudo apt install gcc-powerpc64-linux-gnu
```

Option 2 (For windows users only) –
Install Windows subsystem for Linux https://docs.microsoft.com/en-us/windows/wsl/install . Launch WSL and install the cross compiler.

```
sudo apt install gcc-powerpc64-linux-gnu
```

Option 3 –

If you do not have access to Ubuntu 18.04 (or 20.04) machine, install VirtualBox https://www.virtualbox.org/wiki/Downloads. Download the appropriate platform package for your host OS. Follow the instructions on https://ubuntu.com/tutorials/how-to-run-ubuntu-desktop-on-a-virtual-machine-using-virtualbox#1-overview to start Ubuntu VM. Install the cross compiler using the command below

```
sudo apt install gcc-powerpc64-linux-gnu
```

Compilation

On the Ubuntu terminal, compile the provided matmul.c (Exercises→Code→CH1→matmul.c) using the command below:

```
powerpc64-linux-gnu-gcc -O0 -ggdb3 -std=c99 -static matmul.c -o matmul.out
```

M.1 Preparing the configuration script to run any program of your choice

Refer to the "Creating your configuration script" section on https://www.gem5.org/documentation/learning_gem5/part1/simple_config/ and write your GEM5 script to simulate any program of your choice in POWER ISA

Simulate matmul.c. Run the simulation using the command below

```
$ build/POWER/gem5.opt -d stats_matmul configs/tutorial/<your script>.py
```

`"-d stats_matmul"` Stores the simulation statistics for your script in gem5/stats_matmul

To receive the full credit for this question, upload the stats.txt file

M.2: Adding caches to your configuration script

Refer to the section on “Adding caches to the simple config file” from
https://www.gem5.org/documentation/learning_gem5/part1/cache_config/
And add L1 and L2 caches in your configuration script

Using both qualitative and quantitative analysis, discuss the performance improvements of adding L1 and L2 caches over the same architecture without caches.

M.3: Cache performance profiling for matrix multiplication

For this problem, set L1D cache size as 1kB and L2 cache size as 4KB. To change cache size, make the required changes in gem5/configs/tutorial/caches.py

Make appropriate changes in the provided code (matmul.c) and answer the following questions

A. Report the execution time for all six possible loop orderings, i.e., “i, j, k”, “j, k, i”, “k, i, j”, etc., for multiplication of two matrices with size 64*64. Report the execution times in a table or graph so as to assist you with answering next question

B. Identify which loop ordering(s) performed the best and explain why they performed the best with both qualitative analysis and quantitative analysis. For quantitative analysis, you may use the metrics retrieved from GEM5.

M.4: Evaluating the L1 and L2 cache performance for Power9 and Power 10 CPUs

For each power ISA in the table below, make appropriate changes in configs/tutorial/caches.py to set L1D cache size, L2 cache size, and L2 data latency. Set the matrix size to 256 * 256

	Power 9	**Power 10**
L1D cache size	**32kB**	**32kB**
L2 cache size	**512kB**	**2048kB**
L2 data latency	**20**	**10**

(a) Report the execution time for the two configurations listed above for loop orders i-k-j and j-k-i

(b) Evaluate the performance of the matrix multiplication programs with respect to POWER9 and POWER10. How did the changes to the memory hierarchy affect the performance of the matrix multiplication programs? Explain *why* you think the performance changed the way that it did.

(c) Now, vary the associativity of L1 cache in your GEM5 simulations for POWER9 and POWER10. Which associativity performs the best for POWER9? What about for POWER10? Does the best-performing L1 cache associativity jibe with the L1 associativity for the POWER9 architecture, i.e., 8-way set associative? What about the L1 associativity for the POWER10 architecture, i.e., 8-way set associative for the data cache?

(d) Based on the GEM5-based problems assigned in this and previous homework assignments, write a detailed paragraph about extensions to this case study on the POWER memory hierarchy that could be pitched as potential (modest) project proposals.