Computer Architecture (CSE511/ ECE511)
Assignment 2

General Note: In assignment 1, you were able to use the available gem5 SimObjects to build your own system and play with it. In this assignment, you will get your hands dirty by diving into the gem5 coding environment and getting acquainted with its coding style. You can create your own SimObjects and make it work on the gem5 platform.

The steps and deliverables for this assignment are listed below:

1. Create a SimObject called "VectorOperations." VectorOperations will call three events, namely, VectorCrossProduct, NormalizeVector, and VectorSubtraction. VectorCrossProduct will be called at tick "150". NormalizeVector will be called at tick "1500". VectorSubtraction will be called at tick "15000".
2. VectorCrossProduct will perform the cross-product of two vectors and print the value on the command line. NormalizeVector will print the normalized vectors generated from two initial vectors. VectorSubtraction will subtract two vectors and print the resultant vector on the command line. You can specify the two starting vectors without taking any user input (i.e., you can hardcode them within the SimObject). The dimensions of the vectors should be 3x1.
3. Add DEBUG flags whose functionalities are given below. Each DEBUG flag should also have a proper description/annotation that is displayed on running the Simulation.
    A. DEBUG flag "VECTOR" will display the vectors.
    B. DEBUG flag "RESULTCROSS" will display the resultant value from VectorCrossProduct.
    C. DEBUG flag "NORMALIZE" will display the resultant vectors from NormalizeVector.
    D. DEBUG flag "RESULTSUB" will display the resultant vector from VectorSubtraction.
4. Now that you have implemented your SimObject and added the required flags, create the configuration script to use your new SimObjects. You do not have to add a CPU or caches to the system for this assignment.
5. You should also create a ReadMe file that explains the codes/scripts submitted.
6. Your README should also include the steps to compile and run the code.
7. Submit all the simulation scripts and codes in a zip file with the naming convention: <SA2_Rollnumber>.zip

Bonus:

1. You can optionally take in the cycles to execute the 3 events from the user as the bonus part instead of hardcoding them to 150, 1500, and 15000.
2. You can also optionally take in the vector elements from the user for the bonus part.

Note that you should NOT be storing the resultant values in your code, rather you should compute them when the events are called. You are free to use any linear algebra library that you may want, such as Eigen, Blas, etc. You may also write the code on your own if you want to do so.

Resource: https://www.gem5.org/documentation/learning_gem5/part2/environment/ and the following tutorials on the same website.