HIP Training Day 2 Exercises

1. Use `rocprof` to profile your "vector add" code from day 1.
   a. Use `--hip-trace` to generate a .json file that can be viewed in Google Chrome.
   b. Scp the .json file to your local machine. In the Chrome browser, go to "chrome://tracing" and load your trace
   c. Create a file "counters.xml" and on the first line put "pmc : FetchSize" and the second line put "pmc : WriteSize"
   d. Run rocprof again with "-i counters.xml --timestamp on". A file "counters.csv" will be generated.
   e. The columns of interest are the FetchSize, WriteSize, BeginNS, and EndNS. The bandwidth, in GB/s, can be computed as (FetchSize+WriteSize)/(EndNS-BeginNS)*1000. That is, FetchSize and WriteSize are the data fetched and written in kilobytes and BeginNS and EndNS are timestamps in nanoseconds.
   f. `rocprof --list-basic` and `rocprof --list-derived` will give many more available counters.
   g. Use the roctxRangePush and roctxRangePop APIs to get a roctx region in your Chrome trace. You'll need to recompile your application and then run rocprof with `--roctx-trace`.
2. Use the "hipconvertinplace-perl.sh" to "hipify" the CUDA samples you used to manually convert to HIP on day 1.
   a. For example, if helloworld.cu is a CUDA program, run "hipconvertinplace-perl.sh helloworld.cu". You'll see a "helloworld.cu.prehip" file that is the original and the helloworld.cu file now has HIP calls.
   b. You'll also see statistics of HIP APIs that were converted.
   c. Compile the HIP programs on both on the Nvidia cloud and the AAC. Fix any compiler issues, e.g. if there was something that didn't hipify correctly. Be on the lookout for hard-coded Nvidia specific things like warpsizes and PTX.
   d. Run the programs on both the Nvidia Cloud and the AAC.
3. **Challenge problem**: Consider the code located at https://github.com/NVIDIA-developer-blog/code-samples/tree/master/posts/cuda-aware-mpi-example/src This is a code that does a Jacobi iteration on the GPU. We will attempt to hipify and profile it.
   a. Download this code
   b. Use hipconvertinplace-perl.sh to hipify the code. What else is required?
   c. Update the build system to build using HIP on the Nvidia cloud. You'll need to "module load gcc openmpi" to obtain the MPI stack.
   d. Once your code compiles and runs on the Nvidia cloud, migrate it to the AAC.
   e. Compile and run on the AAC.
   f. Use rocprof to profile the application.