Interoperability: Task 2

In this task, data is copied to the GPU device by means of OpenACC. Computation on the GPU is done via CUDA. All OpenACC-related code should go into the main source code file, `vecAddRed.c`, all CUDA-related code into a dedicated file, `cudaWrapper.cu`. This enables compilation of the individual files by different compilers.

Program Idea

- `vecAddRed.c`: The main part of the program sets up the data and initializes it. `cudaVecAddWrapper()` is defined as an external C function (to suppress C++ name mangling). Data is copied with OpenACC statements to the GPU device before calling the CUDA wrapper.
- `cudaWrapper.cu::cudaVecAddWrapper()`: This function is an interface between plain C and CUDA. It takes in some configuration variables and calls a kernel (`cudaVecAdd()`) with the information
- `cudaWrapper.cu::cudaVecAdd()`: Calculates $c = a + b$ in parallel.

Tasks

- `vecAddRed.c`: `cudaVecAddWrapper()` accepts pointers to data which is already on the GPU. Tell OpenACC to use the device pointers for the arrays.
- `cudaWrapper.cu`
  - `cudaVecAddWrapper()`: Call the kernel (Use the <<< syntax)
  - `cudaVecAdd()`: Implement vector addition
- Compile with `make`, run with `make run`