Exercise 01 - My first kernel

In this exercise, we will try and setup a very simple kernel which takes a vector array as input and update the elements within the device kernel defined in the `MyFirst_kernel.cu` routine. The vector is defined in `MyFirst.cu` to have 32 elements and is to be processed in two threads blocks containing 16 threads each. You will need to update the kernel function such that for each element processed, the thread index within the block is use to updated the value of the element processed. Make sure to familiarize yourself with the main program.

Concepts covered

The following concepts are covered in this exercise

- How to compile.
- How to use block and thread indexes.

By inspecting the supplied code, this exercise also exposes the following concepts

- How to allocate and free memory on GPU.
- How to copy data from GPU to CPU.
- How to invoke GPU kernels.

Files needed

The following files will be needed for the exercise

- `MyFirst.cu` (requires no update)
- `MyFirst_kernel.cu` (needs to be updated)

The code will compile, however, the test included in the main program will fail until `MyFirst_kernel.cu` has been modified successfully.

Work steps

You will need to modify the C device code supplied in the file `MyFirst_kernel.cu`.

1. Define an integer `int` such that it attains a unique global vector index such that the corresponding element in the input vector is only updated by one thread.
2. Update the global vector element `x[tid]` such that it is assigned the thread index number for the threads within the block where it is executed as the output value.

Compilation

Compile the final code using the included `Makefile` which includes the common parameters for compilation in `../../common.mk`. To compile successfully after CUDA has been successfully installed, you will need to make sure that the environment variables `CUDA_INSTALL_PATH` and `CUDA_SDK_DIR` are correct. In case of compilation errors you will need to debug the code until it compiles successfully.

Execution

Execute your compiled program by typing

```
$ ./MyFirst
```

If you program executes successfully the output should look like this

```
n,  x  =  0  1.000000
n,  x  =  1  2.000000
n,  x  =  2  3.000000
...
n,  x  =  29  14.000000
n,  x  =  30  15.000000
n,  x  =  31  16.000000
PASSED!
```