Hello World in CUDA C

Exercise 2:

Write a simple "hello world" CUDA kernel which produces output like

```
...
Hello world! I'm thread 30 out of 64 in block 0. My global thread id is 30 out of 256.
Hello world! I'm thread 31 out of 64 in block 0. My global thread id is 31 out of 256.
Hello world! I'm thread 0 out of 64 in block 2. My global thread id is 128 out of 256.
...
```

and write the corresponding code to launch the kernel for a given launch configuration. Remember a `cudaDeviceSynchronize()` after the kernel launch to force the host to wait until the kernel stops. Use the `nvcc` compiler (a Makefile template is provided on CampusNet).

1. Experiment with the number of threads per block and number of blocks. Is the output deterministic? Run with `./helloworld | wc -l` to count the number of lines.
2. Modify the kernel by having thread number 100 make a segmentation fault just before it prints out the Hello world, e. g., try to access memory you haven't allocated:

   ```
   int *a = (int*) 0x10000; *a = 0;
   ```

   What change do you see in the output - how many hellos are missing? Why?
3. Note that the segmentation fault is **not** caught anywhere, although an error must have occured on the GPU. Try to catch the segmentation fault by using

   ```
   cuda-memcheck ./helloworld
   ```

   when you run your program.
4. Try to catch the segmentation fault by using the helper function `checkCudaErrors(...)` around the `cudaDeviceSynchronize()` in your code like

   ```
   #include <helper_cuda.h>
   ...
   checkCudaErrors(cudaDeviceSynchronize())
   ...
   ```

   What error code do you get? Go to

   `http://docs.nvidia.com/cuda/cuda-runtime-api/index.html`

   and search for the explanation of the error (put the error name in the search field).