Instructions

We will use cub::BlockRadixSort to sort some random numbers on the GPU.

Simple Kernel

Add the `typedef`, shared memory, and the call to the sorting routine to the kernel.

Tip

Most calls to the CUDA API return a status. The following two macros make understanding these status messages easier:

```
#define CUDA_CALL(x) do { if((x)!=cudaSuccess) { \
  printf("Error at %s:%d\n",__FILE__,__LINE__);\
  return EXIT_FAILURE;}} while(0)


#define CURAND_CALL(x) do { if((x)!=CURAND_STATUS_SUCCESS) { \
  printf("Error at %s:%d\n",__FILE__,__LINE__);\
  return EXIT_FAILURE;}} while(0)
```

They are used like this:

```
CUDA_CALL(cudaMalloc(&d_gpu, 4096 * sizeof(double)));
```

Better data load and store

Use cub::BlockLoad and cub::BlockStore to load and store your data instead of

```
for (int i = 0; i < 8; ++i){
    items[i] = data_in[i0 + i];
}
```

Tip

You may use a union for the shared memory.

Flexibility for tuning

Let's make our kernel more flexible by adding more template parameters. The data type is already given by a template parameter.

1. Replace the hardcoded 512 by `BLOCK_WIDTH` and pass it as a template parameter. Don't forget to adjust the main program.

2. Replace the hardcoded 8 (number of items per thread). with `ITEMS_PER_THREAD` and pass it as a template parameter. Don't forget to adjust the main program.

This is better already. Add a couple of kernel calls with different parameters and take a look at the profile.

3. There are different ways of loading data (see cub::BlockLoadAlgorithm). Add a template parameter `BLOCK_LOAD_ALGO` and use it in the typedef of BlockLoadT. Don't forget to adjust the main program.
4. Do the same thing as in 3.) for BlockStore.
5. Finally, add at least 3 calls using different values for the template parameters and look at the program with the profiler. To make the problem more realistic, you can use, e.g, 1000 times the data and run 1000 blocks. Don't forget to calculate the offset for each block. Otherwise, you'll sort the same data 1000 times.

Extra credit

Clean up the code and add proper error handling. The following two macros may be useful: