

## Aalto 2023

Index	Contest	Submissions	Pre	0	CP	1	2a	2b	2c	3a	3b	4	5	9a	9c	IS	4	6a	6b	9a
MF	1	2	9a	SO	4	5	6	P	9a	X	0a	0b	9a	9b						

### SO6: fast GPU solution ★★★

Please note that you can still submit, but as the course is already closed, your submissions will not be graded.

To get started with the development, [download the code templates](#), unzip the file, edit `so.cu`, and run `./grading test` or `./grading benchmark` to try it out – see the [instructions](#) for more details!

Upload your solution as a file here...

Please upload here the file `so.cu` that contains your solution to task SO6.

Choose File

No file chosen

... or copy-paste your code here

Submit

### Your submissions

Your submissions to SO6 will appear here; you can simply [reload](#) this page to see the latest updates.

### What you will need to do in this task

Please read the [general instructions for this exercise](#) first. Here are the additional instructions specific to this task:

Implement an efficient parallel sorting algorithm for the GPU. Any sorting algorithm is fine, but radix sort is perhaps the simplest choice.

### What I will try to do with your code

I will first run all kinds of tests to see that your code works correctly. You can try it out locally by running `./grading test`, but please note that your code has to compile and work correctly not only on your own computer but also on our machines.

If all is fine, I will run the benchmarks. You can try it out on your own computer by running `./grading benchmark`, but of course the precise running time on your own computer might be different from the performance on our grading hardware.

### Benchmarks

Name	Parameters
benchmarks/1	n = 100000
the input contains 100000 integers, and the output should contain the same integers in order	
benchmarks/2	n = 1000000
the input contains 1000000 integers, and the output should contain the same integers in order	
benchmarks/3a	n = 10000000
the input contains 10000000 integers, and the output should contain the same integers in order	
benchmarks/3b	n = 10000000
the input contains 10000000 integers, and the output should contain the same integers in order	
benchmarks/3c	n = 9999997
the input contains 9999997 integers, and the output should contain the same integers in order	
benchmarks/3d	n = 9999998
the input contains 9999998 integers, and the output should contain the same integers in order	
benchmarks/3e	n = 9999999
the input contains 9999999 integers, and the output should contain the same integers in order	
benchmarks/3f	n = 10000001
the input contains 10000001 integers, and the output should contain the same integers in order	
benchmarks/3g	n = 10000002
the input contains 10000002 integers, and the output should contain the same integers in order	
benchmarks/3h	n = 10000003
the input contains 10000003 integers, and the output should contain the same integers in order	
benchmarks/4	n = 100000000
the input contains 100000000 integers, and the output should contain the same integers in order	

### Grading

In this task your submission will be graded using **benchmarks/4**: the input contains 100000000 integers, and the output should contain the same integers in order.

The point thresholds are as follows. If you submit your solution no later than on **Sunday, 04 June 2023, at 23:59:59 (Helsinki)**, your score will be:

Running time	Points
≤ 10.000 sec	1
≤ 6.000 sec	2
≤ 3.000 sec	3
≤ 2.000 sec	4
≤ 1.000 sec	5

If you submit your solution after the deadline, but before the course ends on **Sunday, 04 June 2023, at 23:59:59 (Helsinki)**, your score will be:

Running time	Points
≤ 10.000 sec	1
≤ 2.000 sec	2
≤ 1.000 sec	3

### Contest

Your submissions to this task will also automatically take part in the [contest](#), and you can receive **up to 2 additional points** if your code is among the fastest solutions this year!

Running time	Extra points
≤ 1.20 × fastest	1
≤ 1.05 × fastest	2