

## OpenMP offload programming: the Mandelbrot set

**Don't use gcc or the Sun compiler for this exercise,- please use nvc++ from now on.**

Use your exercise from last week as the starting point for this exercise. For simplicity, use only the C++ compiler from now on (i.e., change all .c file extensions to .cpp). A new Makefile template is provided on DTU Learn.

### Exercise 3:

1. **OpenMP offload mandelbrot loop:** Write an OpenMP offload implementation for generating the mandelbrot set on the GPU using the `target teams loop` construct. Use the `map` clause to transfer the image from the device to the host. Run the executable and check the image in the `mandelbrot.png` file.
2. **OpenMP offload mandelbrot parallel:** Write OpenMP offload implementation for generating the mandelbrot set on the GPU using the `target teams distribute parallel for` construct such that you launch exactly one thread per image point. Run the executable and check the image in the `mandelbrot.png` file.
3. Time your implementations and calculate the speed-up compared to the OpenMP CPU version from week 2. Consider what significance warming up the device has.
4. How much time is used for executing the offload region (compute) and how much for the transfer of the image? Calculate both the speed-up including transfer and excluding transfer. Hints:
  - Use the `target data` construct instead of the `map` clause to separate the memory transfers and offload region execution and time them separately.

Is the speed-up as you would expect?