Serie OpenMP

Exercise 1. OpenMP: hello world

- In the `pi.cc` add a function call to get the number of threads.
- Compile using the proper options for OpenMP

Exercise 2. Parallelize the loop

- Add a `parallel for` work sharing construct around the integral computation
- Run the code
- Run the code
- Run the code
- What can you observe on the value of pi ?

Exercise 3. Naive reduction

- To solve the raise condition from the previous exercise we can protect the computation of the sum.
- Add a **critical** directive to protect the sum
- Run the code
- What can you observe on the execution time while varying the number of threads

Exercise 4. Naive reduction ++

- Create a local variable per thread
- Make each thread compute it's own sum
- After the computation of the integral us a **critical** directive to sum the local sum to a **shared** sum

Exercise 5. Reduction

- Use the **reduction** clause
- Compare the timings to the previous versions

Exercise 6. Poisson

- Now you can apply what you learn to the `poisson` code.
- Remember that 90% of the time is spend in the dumpers. So modify the behavior to dump only at the end of the simulation to get a validation image.