



Eidgenössische Technische Hochschule Zürich  
Swiss Federal Institute of Technology Zurich

High-Performance Computing Lab for CSE

2024

Student: Benedict Armstrong

Discussed with: FULL NAME

---

Solution for Project 02

Due date: 25 March 2024, 23:59

---

**HPC Lab for CSE 2024 — Submission Instructions**

(Please, notice that following instructions are mandatory:  
submissions that don't comply with, won't be considered)

- Assignments must be submitted to Moodle (i.e. in electronic format).
- Provide both executable package and sources (e.g. C/C++ files, Matlab). If you are using libraries, please add them in the file. Sources must be organized in directories called:  
*Project\_number\_lastname\_firstname*  
and the file must be called:  
*project\_number\_lastname\_firstname.zip*  
*project\_number\_lastname\_firstname.pdf*
- The TAs will grade your project by reviewing your project write-up, and looking at the implementation you attempted, and benchmarking your code's performance.
- You are allowed to discuss all questions with anyone you like; however: (i) your submission must list anyone you discussed problems with and (ii) you must write up your submission independently.

**1. Computing  $\pi$  with OpenMP [20 points]**

**2. The Mandelbrot set using OpenMP [20 points]**

**3. Bug hunt [10 points]**

1. The first bug is a compile-time bug. The `#pragma` directive must be followed by a for loop. In this case we have a `tid = omp_get_thread_num()` statement immediately after the `#pragma`. This is not allowed. The fix is to move the `tid` assignment into the for loop.
2. There are a couple of errors in the code. The variable `tid` should be made explicitly private as every thread is writing to it. In the last for loop the total sum should be marked as a reduction variable (using the `reduction(+:total)` clause). Also by default the second loop will not spawn any new threads as the option `OMP_NESTED` is set to `FALSE` by default (see IBM OpenMP documentation).

**4. Parallel histogram calculation using OpenMP [15 points]**

**5. Parallel loop dependencies with OpenMP [15 points]**

**6. Quicksort using OpenMP tasks [20 points]**