Exam Rules & Info

⚠️ Carefully read this document through the end before starting the project and booking/showing to the exam.
I tried to cover all the possibles Q&A, but if anything is unclear let me know and I'll update this document accordingly.

Scoring Info

The exam is composed of a project, a short report describing it, and an oral exam. The finale score will be determined as follows:

- Performance (up to 10 points) — Assigned per project
  - up to 5 points for the MPI+OMP version
  - up to 5 points for the CUDA version

🚨 The performance will be evaluated on the cluster, so be sure your code works and performs well there. If your code works on your laptop, but does not work on the cluster (e.g., it provides a wrong result, it crashes, etc...), you will fail the exam.
Don't wait until the last few days for doing tests on the cluster, since there will be a high load.

👉 For instructions on how to access and use the cluster, please refer to https://hlc-lab.notion.site/Sapienza-CS-Cluster-3ab092e1b1874255bcef81ba37c3a16a?source=copy_link

> 👉 You can find the leaderboard here: https://fra179.github.io/MCPLeaderboard/exercises/final-exam/?task=test-mpi-openmp#leaderboard
> There are two tabs, one for the MPI+OMP and one for the CUDA implementation.

- Report & Analysis (up to 6 points) — Assigned per project
  - up to 4 points for report clarity, and scaling/efficiency analysis (strong/weak scaling)
  - up to 2 points for profiling evidence (identify/solve bottlenecks either using sections timings or profilers)

> 👉 For the scaling/efficiency plots, it is better to use the cluster, due to the higher number of nodes/cores.

- Project Defense (up to 6 points) — Assigned per student
  - up to 3 points for the “elevator pitch” (i.e., describe the key design choices clearly and concisely — max 10 minutes per team/project)
  - up to 3 points for Q&A on the code. I’ll ask question about design choices, potential alternative designs, etc.
- General Oral Exam (up to 8 points) — Assigned per student

> 🚨 Questions can cover any topic we have seen during the course.

Project

- Join the Google Classroom if you did not do it yet (code: 4jn7bzbp). If you just joined the Google classroom, please send me an email so that I can transfer your account from Google Classroom to Github Classroom

- Then, join the project on Github Classrom at the URL https://classroom.github.com/a/Lt5_BlSq

> 🚨 Pay attention when joining Github Classroom when you are asked to link the Github account to your Google Classroom username. Those are the username I'll reference to for assigning you a grade for the project. If you click on someone's else username, write me for fixing it (you will lose points).

- The submission of the project must be made directly through Github Classroom (i.e., push the code).

> 🚨 The project must be submitted **at least two days** before the date the appello starts. I.e., if the appello is on February 6th, I will consider commits made until February 4th 11:59 PM. The last commit made before that time is the one that will be used for grading the performance part (i.e., be sure the last commit you make before that date is the best version of the code you have).

- Teams can be composed of at most 2 students. In that case, the same grade (for the project part) will be assigned to both students (i.e., it does not matter who does what, all students in the team need to know all the details of the project).
- The team must submit an implementation using OpenMP+MPI, and one using CUDA (see instructions on the project repository).
- The team must show proficiency with both GPU programming (CUDA), CPU multicore programming (OpenMP), and distributed memory programming (MPI).
- The parallel implementations MUST scale. If it does not, you must motivate why in the report (see below).
- Verify the correctness of the parallel versions by comparing the results with those obtained by the sequential version. If you find small discrepancies due

to numerical precision, please send me an email. In principle, I will consider differences within 10^-5 (for `float`) to be acceptable.

- I will gather performance data using the two commands you can find in the `Makefile` (`run_mpi` and `run_cuda`). That will tell you which input exactly I am going to use as a reference.

> 👉 If you want, you can modify the flags used to compile/run the applications by modifying the `student.mk` file. That's the only other file you are allowed to modify besides for the `*cuda_core.c` and `*mpi_omp_core.c` files.

- The project will be valid until November appello. From the next academic year, a new project will be assigned.
- You can decide to do a different project, but that must be first approved by me (in that case, send me an email asking for approval of the project, before starting working on it)

Systems to use for implementing and testing the implementations

- My suggestion is to use your laptop/workstation at least in the preliminary stages while debugging/implementing the code.
- If you have an AMD rather than an NVIDIA GPU, you can still write CUDA code, and then convert it to HIP code (the AMD's equivalent of CUDA), using their `hipify` tool.
- If you do not have a laptop/workstation with a GPU, you can use Google's Colab. You can find instructions on how to run CUDA on Colab linked on Moodle.
- After the implementation/debug phase, use the cluster to do performance tuning and to gather the data for the report.

Report

Together with the code, you must present a PDF report. It must be no longer than 8 pages, and must contain the following:

- Description of the implemented solution. Discuss the main implementation choice, as well as alternative choices that you tried and did not work and/or did not perform well. The trial and error process is important and shows maturity.
- Experimental evaluation. Report weak/strong scaling, and use all the good practices we discussed (repeat the experiment multiple times, report variability, etc...)
- Discussion on the limitations and ideas on how to address them (e.g., «we might need a system with a faster network», «we were bottlenecked by the GPU memory bandwidth», etc...)

There is no need to describe the goal of the project etc (unless you are doing a custom project).

🚨 The report must be part of the repository (name it as `report.pdf` and leave it in the main repository folder)

You can find examples of very good reports (i.e., reports to which I would give a high grade) in the following:

zanoni_martinelli.pdf

Oral Exam

- Register to the *appello* on Infostud. I will not allow in the exam students that did not register on Infostud. If you made the project in teams of two students, both students must register on Infostud.
- The exam can happen at any time during that day. I'll communicate exact time after I gather all the registrations. If too many students register for one appello,

some might be moved to the day/days after. For sessions *“su appuntamento”* (e.g. first session 2025/2026) I will send registered students a form to book a specific slot.

🚨 For ACSAI students: I selected exam dates to avoid conflicts with other third-year exam for the Informatica degree. If those dates conflict with other exams you have on the third year, please let me know and we’ll find a solution.