Welcome to Codee Training, Apr 2022
Introduction

- **Codee (previously known as Parallelware Analyzer)**
  - A programming development tool for C/C++/Fortran parallel codes on multicore CPUs and GPUs using OpenMP and OpenACC
  - Can **automatically insert OpenMP or OpenACC directives** in codes
  - Produces performance **optimization report** with recommended actions
- **Hands-on training provided by Appentra to NERSC/OLCF users**
  - Thanks **Manuel Arenaz (Founder and CEO), Ulises Blanco, Sergio Pardo, Fani Garcia**
- **Organizers**
  - NERSC: Helen He, Woo-Sun Yang, Rebecca Hartman-Baker, Madelyn Blair, Chris Daley, Brandon Cook, Daniel Fulton
  - OLCF: Suzanne Parete-Koon, Tom Papatheodore
Some Logistics (1)

- Muted upon joining Zoom due to large number of attendees
- Please change your name in Zoom session as “first_name last_name”
  - Click “Participants”, then “More” next to your name to rename
- Live “Captions” and “View Full Transcripts” are enabled
- Q&A: use Google Doc (preferred) instead of Zoom chat
  - https://tinyurl.com/codee-QnA-apr2022
- Slides/videos/exercises will be uploaded to the event web page
- NERSC Office Appointments for Codee will be available later
- Please help us with answering the survey after the training
  - https://tinyurl.com/codee-survey-apr2022
Some Logistics (2)

- Users are added to the ntrain2 project for hands-on
  - Training accounts are valid through May 3
- Perlmutter GPU nodes are reserved during the training
  - Apr 26: 9 am - 1 pm, `#SBATCH --reservation=codee_day1 -A ntrain2_g`
  - Apr 27: 9 am - 1 pm, `#SBATCH --reservation=codee_day2 -A ntrain2_g`
- Hands on materials
  - Use your NERSC login or training account on Perlmutter
    - `% ssh saul-p1.nersc.gov` (or ssh perlmutter-p1.nersc.gov)
    - `% cd $SCRATCH`
    - `% cp -r /global/cfs/cdirs/training/2022/Codee_Apr2022 .` (notice the dot)
- Perlmutter documentation
  - [https://docs.nersc.gov/systems/perlmutter/](https://docs.nersc.gov/systems/perlmutter/)
<table>
<thead>
<tr>
<th>Part</th>
<th>Topic and Format</th>
<th>Date and Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>Introduction to Codee tools: Shift Left Performance</td>
<td>Tue, Apr 26 9 am - 12 pm PDT</td>
</tr>
<tr>
<td></td>
<td>• Introduction to Codee and the shift left approach</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Open catalog of coding rules for performance optimization</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Automated code inspection with Codee: Discover and Adopt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Quick start to Codee: Canny image processing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Hands-on: Optimizing PI on Perlmutter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Format:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Remote lectures (~30’), demos. and hands-on sessions</td>
<td></td>
</tr>
<tr>
<td>#2</td>
<td>Usage of Codee for GPU programming (1/2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• The GPU programming challenges</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Memory usage, massive parallelism exploitation, and data transfers minimization</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Codee’s support to address memory usage and massive parallelism</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Hands-on: Optimizing MATMUL on Perlmutter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Format:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Remote lectures (~30’), demos, and hands-on sessions</td>
<td></td>
</tr>
</tbody>
</table>
#3  Usage of Codee for GPU programming (2/2)

- The GPU programming challenges
- Codee’s support to address data transfers minimization
- Hands-on: Optimizing MATMUL on Perlmutter

Format: sessions
- Remote lectures (~30’), demos, and hands-on exercises  
  Wed, Apr 27

#4  Putting it all together

- Hands-on: Optimizing LULESHmk on Perlmutter
- Hands-on: Work on your own code

Format:
- Remote demos and hands-on sessions
Using codee at NERSC

- % module load codee  (default version is 1.3.1)
- % pwreport <options> or % pwdirectives <options> or % pwloops <options>
  - help menu: “pwreport --help”, “pwdirectives --help”
  - docs and examples directories in codee installation on Perlmutter
    - cd $CODEE_DIR

- Can use login node for most of development work above for GPU
- Can use any compiler, such as gcc or nvidia compiler, for own applications
- Can then submit batch jobs to run the generated/improved OpenMP offload or OpenACC codes on GPU nodes
Thank you!