CompPhysHack 2026

Computational Physics Hackathon

T. Misawa, S. Ohno, T. Okubo, H. Shinaoka

March 6–8, 2026

Naha, Okinawa

What is CompPhysHack 2026?

A 3-day hands-on event on **integrating AI tools into computational physics workflows**

- Hands-on experience with agentic coding tools
- Work on **whatever you want** — individually or in groups
- Teams = smallest unit of mutual help and knowledge sharing; cross-team collaboration encouraged
- Free, 40 participants

Website: qc-hybrid.github.io/CompPhysHack2026

Schedule

Day 1 (Mar 6) — Talks · Lectures · Exercises
Day 2 (Mar 7) — Live demo · Team formation · Team work
Day 3 (Mar 8) — Team work · **Results presentations**

Today:

09:30	Short talk on vibe coding — H. Shinaoka ← now
10:15	Lecture — S. Terasaki (AtelierArith)
13:30	Vibe coding exercises
17:30	Short intro + open discussion — L. Wang (IOP, CAS, Beijing)

Two generations of agentic coding tools

	Generation 1	**Generation 2**
Tools	Cursor, VS Code Copilot	Claude Code, Codex CLI, OpenCode, ...
Style	Human watches editor, reads AI suggestions	AI works autonomously, human sets goals
Code reading	Human reads generated code	Human rarely reads code
Good for	Writing papers, light editing	Coding — this is now the main arena

Start with Generation 1 if you are new — that is totally fine.
If you feel ready, explore Generation 2.
The goal is to experience a world you may not have seen yet.

Recommended tools

Codex CLI	Included in ChatGPT Plus · github.com/openai/codex
Claude Code	Pro plan $20/month · claude.ai/code
OpenCode	Open-source CLI · works with any API · github.com/sst/opencode
Cursor	IDE-based Gen 1 · good starting point · cursor.com

Tip: OpenCode + z.ai coding plan ($30/month) — not cheap, but generous token limits

Let's share what works for us over the 3 days!

My Year of Agentic Coding

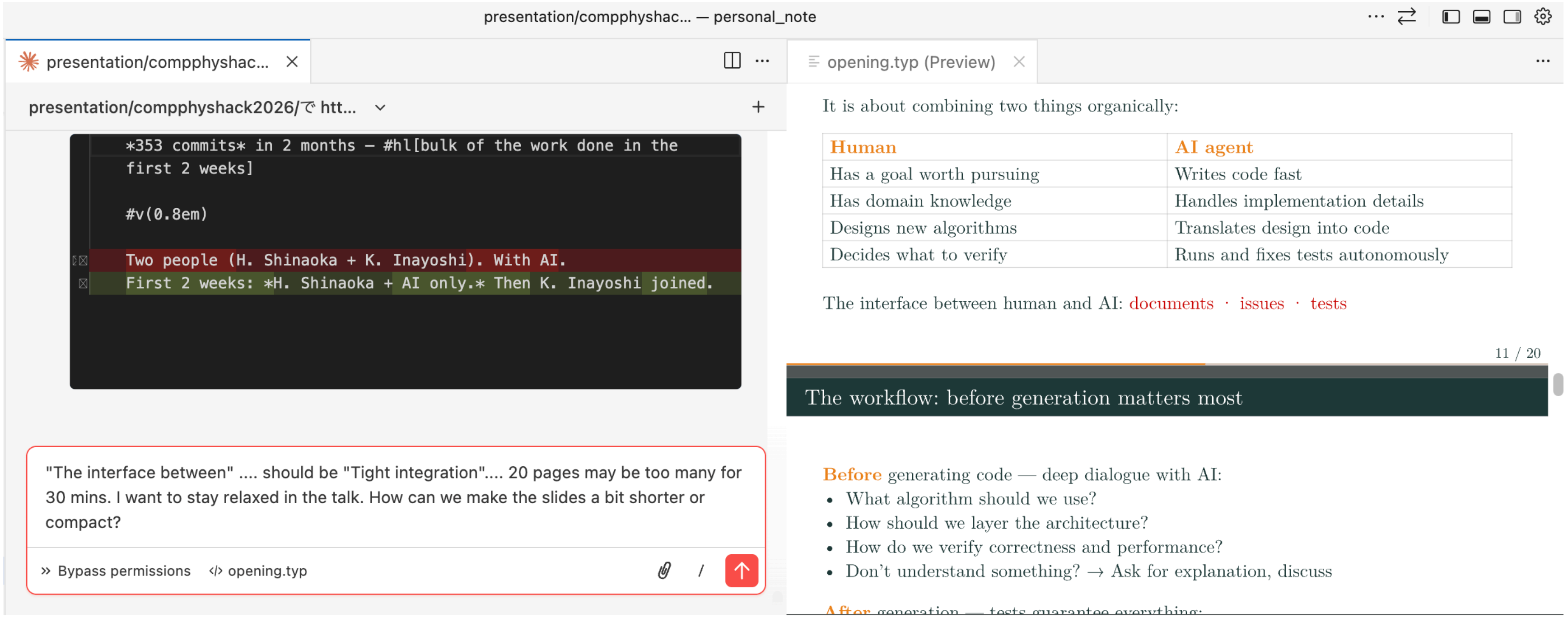
How I got here

Since October 2025, I have not written a single line of code — and I do not read generated code line by line.

Jan 2025	Started **Cursor** — first experience of agentic coding
Oct 2025	Began migrating sparse-ir-rs from **C++ to Rust**
Dec 2025	Switched to **Claude Code** — stopped reading generated code
Jan 1, 2026	Started **tensor4all-rs** — Rust port of Julia TN ecosystem
Feb 2026	Started **tenferro-rs** — PyTorch-like tensor stack in Rust
Mar 2026	Now using **Codex**

Acknowledgements: **Jin-Guo Liu** (HKUST-GZ), **Satoshi Terasaki** (AtelierArith), Ken Inayoshi, **Lei Wang** (IOP, CAS)

This talk was written the same way


presentation/compphyshac... — personal_note

presentation/compphyshac...

presentation/compphyshack2026/で htt...

```
*353 commits* in 2 months — #hl[bulk of the work done in the
first 2 weeks]

#v(0.8em)

Two people (H. Shinaoka + K. Inayoshi). With AI.
First 2 weeks: *H. Shinaoka + AI only.* Then K. Inayoshi joined.
```

"The interface between" .... should be "Tight integration".... 20 pages may be too many for 30 mins. I want to stay relaxed in the talk. How can we make the slides a bit shorter or compact?

It is about combining two things organically:

Human	AI agent
Has a goal worth pursuing	Writes code fast
Has domain knowledge	Handles implementation details
Designs new algorithms	Translates design into code
Decides what to verify	Runs and fixes tests autonomously

The interface between human and AI: documents · issues · tests

11 / 20

The workflow: before generation matters most

Before generating code — deep dialogue with AI:
- What algorithm should we use?
- How should we layer the architecture?
- How do we verify correctness and performance?
- Don't understand something? → Ask for explanation, discuss

After generation — tests guarantee everything:

Typst + Claude Code · **AI-human feedback loop**

Past blog articles + GitHub repos as context

I did not write a single line of Typst.

What I built: tensor4all-rs

Rust port of the **tensor learning software stack** [SciPost Phys. 18, 104 (2025)] · github.com/tensor4all/tensor4all-rs

Original libraries: TensorCrossInterpolation.jl / QuanticsTCI.jl / QuanticsGrids.jl / ITensorMPS.jl / xfac

Jan 1, 2026 · 353 commits in 2 months · first 2 weeks: +61,486 lines (151 files) · H. Shinaoka + AI only

"In the era of AI agents, rewriting is nearly free.
The bottleneck is finding the right abstractions."

This insight → started **tenferro-rs** with Jin-Guo Liu: a tensor stack **built around AI-Human integration**

Why Rust? — fast feedback loop with AI

Python and Julia were designed for **fast development** — easy to write, easy to read.

With AI, this advantage is **reversed**:

Python / Julia	**Rust + AI**
Fast to write by hand	AI writes it — speed is equal
Errors found at runtime	Errors caught at compile time
Must run code to verify	`cargo check` in seconds
AI mistakes found late	AI mistakes found **instantly**

I now feel that **Rust + AI is faster** than Python or Julia for development.

tensor4all-meta/docs/why_rusty_julia.md · Zenn: Why I Migrated from C++ to Rust

Agentic coding is not copy-paste

Agentic coding $\neq$ asking ChatGPT and copying the output

Copy-paste is also **slow** — you context-switch between AI and editor constantly

It is about combining two things organically:

Human	**AI agent**
Has a goal worth pursuing	Writes code fast
Has domain knowledge	Handles implementation details
Designs new algorithms	Translates design into code
Decides what to verify	Runs and fixes tests autonomously

Tight integration via: documents · issues · tests

What is “coding” now?

Before	**Now**
Write code	Dialogue with AI **before** generating
Read and debug code	Design and verify
Implement algorithms	Understand algorithms
Fix bugs line by line	Tests that guarantee behavior

Generating code: discuss algorithm, architecture, verification strategy with AI **first**
Reading code: not line by line — skim, ask AI to investigate anything unclear

Domain knowledge and the ability to verify become **more** important, not less.

Example: tenferro-rs/docs — 18 design docs + per-file coverage thresholds

The Spirit of This Hackathon

This is not a lecture

I am not sure about the **right way** to do agentic coding.

And I am not sure **how to teach coding** anymore.

Nobody does — yet.

This hackathon is a space for exploration.

We try things, make mistakes, and **discover together** what works for computational physics.

How to participate

No fixed theme — work on whatever excites you. Team = smallest unit of mutual help.

Sample projects (ask organizers):

- **Classical Monte Carlo** — Ising model, statistical mechanics
- **tenferro-rs** — PyTorch-like tensor library in Rust (for experienced users)

Or bring your own:

- A script you always wanted to refactor · a simulation you never had time to write
- A library in Python you want to port to Julia or Rust

If stuck: ask your team · ask an organizer · ask AI (then a human to check)

The goal is not a finished product. It is to discover new ways of coding — together.

Let’s explore together

3 days. 40 people. Many AI tools.

No one knows what “correct” agentic coding looks like yet.

Let’s find out.

Welcome to CompPhysHack 2026 🌺