17th Advanced Scientific Programming in Python

Summer School

21–28 September, 2025. Plovdiv, Bulgaria.

Evaluation Survey Results

Method

The survey has been administered with a web interface created with the LimeSurvey software available at: http://www.limesurvey.org
All answers have been submitted by 13 October 2025.
No answer was mandatory.
The free-text answers have not been edited and are presented in their original form, including typos.

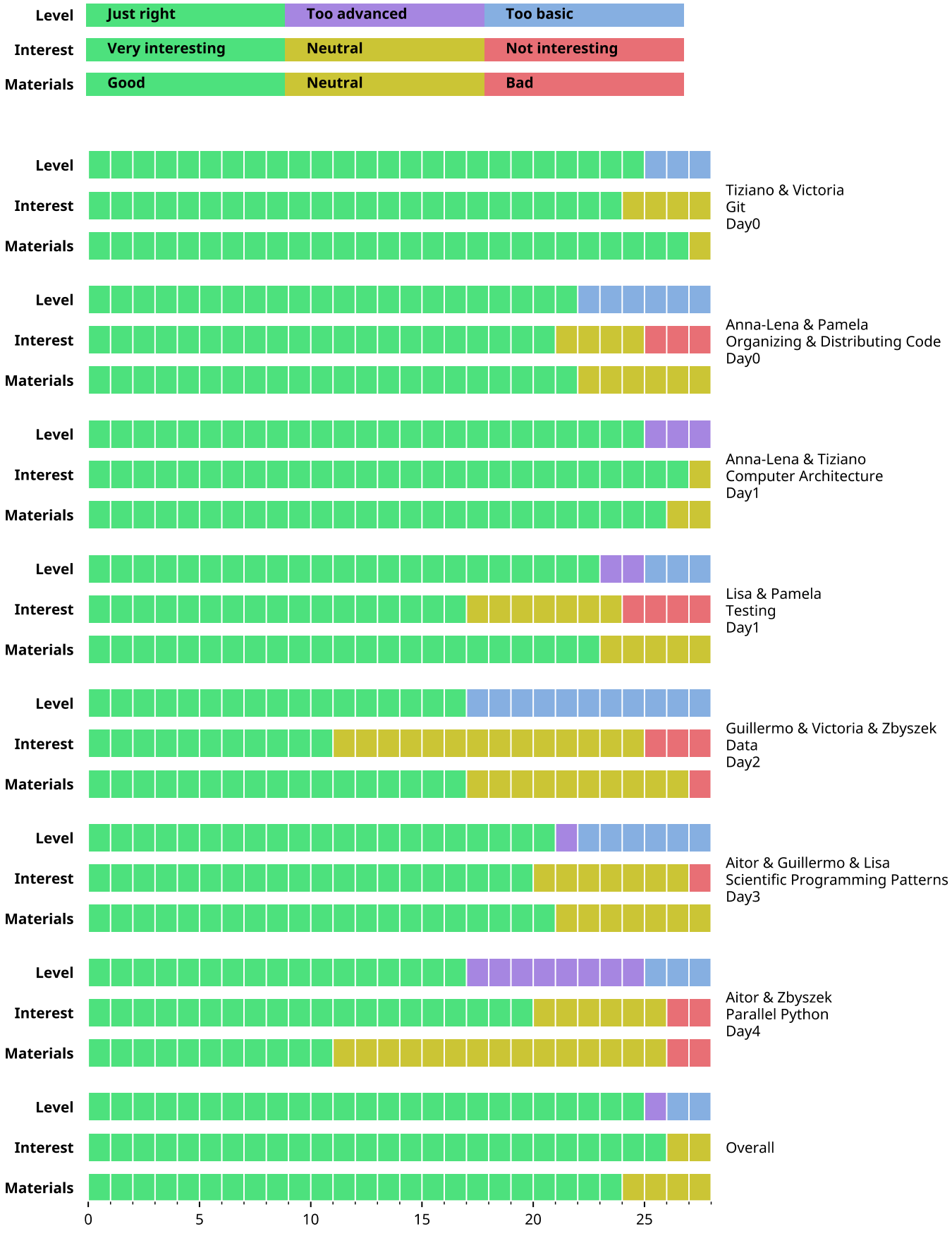
Attendants and Applicants Statistics

	Attendants		Applicants	
	28	27%	102	
Different nationalities	18		34	
Countries of affiliation	12		26	
Gender: other	1	4%	2	2%
Gender: female	19	67%	62	61%
Gender: male	8	29%	38	37%
Already applied	6	21%	16	16%
Bachelor Student	0	0%	2	2%
Master Student	1	4%	20	20%
PhD Students	22	79%	56	55%
Post-Docs	3	7%	10	10%
Professor	1	4%	2	2%
Employee	0	0%	2	2%
Others	2	7%	10	10%
Completed surveys	28	100%		

More stats about attendants are available at: https://archives.aspp.school/2025-plovdiv/students.html

Lectures & Exercises

Q: *Grade the* ***level*** *of the lectures*
Q: *Grade how* ***interesting*** *were the lectures*
Q: *Grade the quality of the presentation style and/or of the teaching* ***material,*** *e.g. the clarity of the slides/code, the exercises and the solutions, etc.*



Q: Are some of the topics presented in the lectures not relevant for a programming scientist?

1. I think everything was relevant. Parallel processing might not apply to scientists who do not use big data sets but I think most will at some point so it's good to have. But some of that lecture was a bit too chaotic for me to understand exactly how I might implement it and when.
2. I thought the lecture about pandas was a bit too specific it would have been nice for it to be a little more general not all large data is tabular. The rest was good.
3. I think that Parallel Python was not relevant for the target audience (i.e. self-taught, application-focused scientists). For this level, most packages such as Numpy/Polars already have a decent implementation of parallelization, and custom implementations start to matter only for package developers. Although the content of the lecture was fun and the explanations by Aitor/Zbigniew were quite illuminating, I think this class could be substituted with something more relevant. Again, my opinion is only for this year/audience.

Q: Are there further topics relevant to the programming scientist that could have been presented, given that the total time is limited. Please also mention which topics should be replaced by the new ones.

1. I would consider a session to advice People what kind of visualizations are appropriate for the type of data that they have.
2. Some of very relevant topics were planned but unfortunately not dealt with due to time limitations. It would have been really really good to start the organizing and distributing code with virtual environments, that's often how a project starts and that is needed as the very first thing.

   People were interested in and asking about the workflows and the tools that the faculty uses. One person presenting theirs might make it seem like this is how it needs to be done, but seeing the diversity in the workflows could also be very interesting and useful. Many times the questions and demonstrations touched on it anyway. Git lecture could have been a bit shorter maybe for in a longer lecture for organizing and distributing code a fun little tour of faculty's preferred tools could be shown. So maybe a couple people move away from vscode by the end of the course because they realize all of that is possible in many different ways.

   Parallel python was great, I am a bit afraid that some of the things that are very good practice and use numpy's parallelization might be unclear so people know at the end some numpy way of doing it would be better but won't do it because they are not sure how to do it with numpy and linear algebra. In this case it can be a decision from faculty to not try to show different functionalities but instead encourage for people to find these alternatives to say, and convince students that these solutions most likely already exist and are implemented. Not sure where to cut for this, maybe it could be sneaked into exercises.
3. For younger generations I wonder if it will be necessary to incorporate the fact that they use chatgpt a lot. I wonder if in the future, it will be more important to know how to prompt (probably in a piecemeal fashion) and test extensively. Many scientists don't test because they feel they already spent too much time developing the code and time pressure gets the better of us. Could there be an appropriate way to use AI effectively (eg. To suggest completions to pseudocode) and test rigorously. Would the shift of time resources to testing rather than debugging syntax errors lead to more trustable code?
4. Not really.
5. I would like to know programming practices for API development. Like when 2 devices are communicating, how should we organise our code? Should we have a linear structure or a tree, or how specialised should each module be? etc.
6. How to deal with AI generated code could be addressed a bit more, since this is part of our daily work (even if we don't like it).

   Maybe some plotting exercises.

   Maybe cut a tiny bit on parallel processing.
7. The choice of topics was excellent, with almost everything being highly relevant to my future work. The only improvement that could be made to the Parallel Python module would be to restructure it slightly and simplify it a bit more. As someone who had never studied multithreading or multiprocessing before, I found it difficult to understand the differences and respective applications. Topics such as the GIL system in Python were covered fairly quickly.
8. Maybe how to estimate the hardware resources required for the particular code (e.g. sometimes we may need to rent a server for particular task, and how to estimate minimum adequate requirements of CPU and RAM), can be short and added to computer architecture part.
9. There are probably many more relevant topics, but I would have a hard time deciding which of the already included ones should be skipped. All the presented topics are relevant, and I would keep them in the ASPP schedule.

10. I'm not sure if this is a full topic, but it might be nice to have github workflows included in either of the git/testing lectures.
11. I don't see how to squeeze other possible important topics in the given time...
12. Containerization could be incredibly useful. Snakemake/Nextflow or other pipeline automation tools also can be useful to many. Maybe that could replace pandas basics?
13. Environment management (virtual environments, environment managers such as conda/mamba/pixi) were touched upon, but I believe they should get a proper full-time treatment. In addition, the topic not present at the course but very important for a scientist is data visualization. This might be less programming and let's be honest Python's data viz ecosystem is a bit too messy, but I'm sure you could find a way to make it work and be useful for the audience.

Q: Do you think that pair-programming during the exercises was useful?

Yes, I have learned from my partner / I have helped my partner	93% (26)
No, it was a waste of time for both me and my partner	4% (1)
Neutral. It was OK, but I could have worked by myself as well.	4% (1)
Other	0% (0)

Other:
1.

Q: What do you think of the balance between lectures and exercises? When answering, please keep in mind that the overall time is limited ;-)

Lectures were too long, there should be more time for exercises	7% (2)
Lectures were too short, there should be more time for lectures	0% (0)
The time dedicated to lectures and exercises was well balanced	79% (22)
Other	14% (4)

Other:
1. For some of the lectures it could have been more fine tuned. For some exercises the planned time was too long for others too short. All in all I would have appreciated more exercises, and slightly more to-the-point.
2. The overall time for each felt balanced, but since many weren't able to finish their exercises there should be less exercises (with optional extra tasks)
3. More consciese excercises would be beneficial.
4. I would tend to say we should have had more time for exercises, but it seemed to be because time was wasted on answering some very specific questions during the class rather than redirecting them to after.

Q: Any further comments about the lectures and exercises?

1. Overall there was a great balance and good level of activities that was keeping people awake.
2. The live coding was sometimes very useful and sometimes rather confusing.
3. I found them useful and mostly easy to follow. Although I didn't find the material too advanced, I needed some more time to process things, but that is a personal, at the same time I didn't feel rushed. I felt there was a good balance. Also, I really loved how safe and non-judgemental the environment felt, I really am incredibly grateful!
4. Overall I really loved them and I think despite being long days I could stay really focused and engaged with the topics. This makes me think that it is already very good and suggestions for improvement may not be such an improvement but if I had to, I would say some presenter teams could have been a little more coordinated especially for the newer topics.

5. All in all everyone in the faculty did a great job, everybody deserves a lot of praise! <3

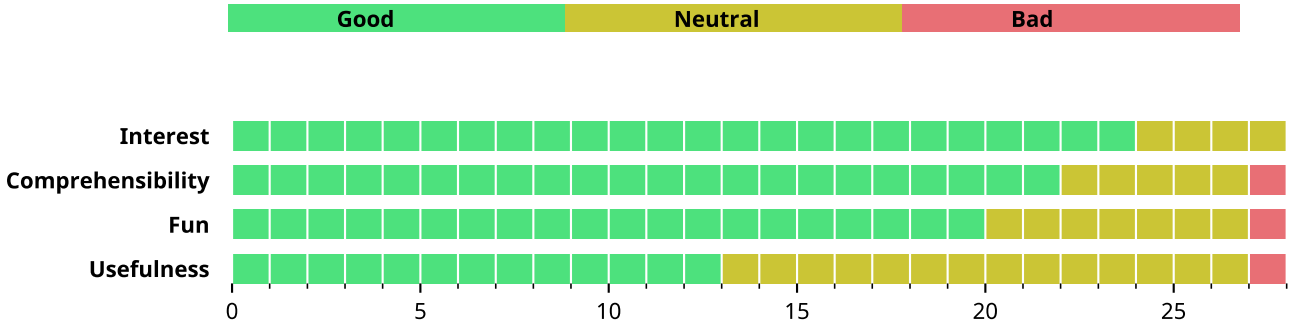
   Loved all the live coding, it is always very useful to 1. see things in action, 2. see experienced people debugging on the go and demonstrating there is nothing to be ashamed of, this is how it goes sometimes. It can make people more comfortable writing code in front of others as well. Where time was running out and it was difficult to gather the group after an exercise some parts could also be converted into live coding. Maybe this already happened behind the scenes.

   At the start of each day a really brief recap could be good so the learnings carry over to the next day. Time is limited but this can even be a scheduled message or an email that is adapted to what has been done and what was left out due to time. Or something that the students should prepare as just a few bulletpoints.

   Questions were thoroughly addressed, even ones that were sometimes dragging on. It is great to not brush over questions, but it's a shame if that eats away from the time for exercises where the point even could have become more clear. Some analogies to wet lab could also be useful.
6. I think it's important to leave enough time for each lair to complete their exercise. Advanced pairs should have optional additional tasks while tutors help less advanced pairs. So I would plan less tasks and more time for each task, as truncated tasks have a greater cost than the possible benefits of many truncated tasks.
7. The balance between lectures and exercises was fine but the exercises could be more concise, maybe a bit shorter to be able to discuss the results.
8. They were both very interesting. Exercises are key to understand the content and they were well designed.
9. Very interesting, fun and useful content!
10. A huge thank you to the entire faculty! You did a fantastic job of explaining the topics at just the right level of abstraction, and you created a very interactive and safe learning environment. Your enthusiam (for teaching) was truly inspiring!
11. Sometimes I felt that the exercises were too dummified compared to what was said during the lectures, though I can understand how more complex exercises would require more time spent explaining. Also, when we were running out of time, it was always the exercise that got skipped, not the lecture. Again, understandable, if the exercise is reliant on stuff that was not yet said, but otherwise, the exercise should take precedence to help us improve those muscles. Since this will be published, I'd like to state that I was overall very happy with both lectures and exercises, these are just some ideas for improvement :)
12. Time is limited, and generally, lecturers managed to handle it well.. It was only unfortunate that two arguments that specifically for me were relevant (creating Python environments and debugging) were kind of rushed because of time constraints... but i understand it is difficult

Programming Project

Q: Evaluate the programming project.
Interest: How interesting was the programming project?
Comprehensibility: How clear and comprehensible was the code and the available documentation? Was it easy to work on the programming project
Fun: Was it fun to work on the programming project?
Usefulness: Was it useful to work on the programming project? Do you think you may re-use what you learned?



Q: Do you think the team-programming experience is relevant to your work as a programming scientist?

Yes: 82% (23)
No: 18% (5)

Q: Do you think that the project should be about a real-world scientific problem instead of a video game?

Yes: 14% (4)
No: 86% (24)

Q: Any further comments about the programming project?

1. It was really a lot of fun ????
2. During the project work I did learn a lot which was part of my goals, but there wasn't any code-based contribution of mine that made it to the final outcome. I understood what was the general improvement to our bot but I had no idea how it was implemented, which makes me feel like I missed on the main objective - learn how to implement improvements. Maybe in the future you can suggest potential contribution paths for more experienced vs less experienced participants so that everyone can feel like they can add something feasible to the code.

   Something that made working in my group difficult was that there were too many natural leaders and at one point it felt like we were competing with each other. In other words I don't think we ever became a team.
3. I loved that the project was a video game and not a random research project. A lot of the principles we learned in the course could be practiced and applied - so it served the purpose really well. I wonder if you could make pelita a cooperative instead of competitive game. That'd be really refreshing and cool! (Not that I didn't enjoy the final game watching but I think it could be as entertaining as a cooperative game would be)
4. Although as time progressed I felt more stressed about finishing and skipped most of the good practices we had learned during the week, I still find the whole experience really useful. Although I really liked my group, I found it difficult at times to figure out when I had to encourage my partner who was struggling a little and where I had to step up, so they don't feel pressure. Also it helped develop my intuition about git further. It was fun!
5. The project was just fine, I just think more attention should be paid to assigning people to the groups, as I had the feeling my group was not well balanced and had a majority of people with little knowledge.
6. Incredible learning experience for collaborative work. I hope my work structure changes in such a way that I can apply what I learned for real.
7. The code base is quite complex for less advanced programmers like myself. It took a lot of effort to understand, which gave me less time to implement things, let alone implement them with a focus on good practice.

   Ideally the project would be more science/data analysis oriented but I understand this is basically impossible with such a diverse group of scientists and the videogame is a “simple” problem we can all discuss and the competition can be motivating.

8. The project was great. It would have been nice to have a little more documentation. I think structuring the bots as "move" functions is a bit counterintuitive. I would have thought that -given the previous lectures and the importance of understanding classes in python- the bots would work as a class with attributes to store data instead of "state".
9. Even though Pelita is far from our real work, it was really fun! For a team project it was the right choice, rather than a scientific project.
10. The game and its documentation are prepared with great attention to detail! It's very engaging and allows to both put the new skills to practice but also to use your creativity while developing the strategies.
11. I would like to see/ discuss solutions of other teams.
12. I thought that the project was really fun, and accessible to everyone!
13. Personally, I couldn't really apply most of the stuff we learned due to time constraints (data management, testing, clean coding). I still found the project to be very useful because it gave us the full-on team coding experience with all of its hurdles and bonuses, and I feel it also helped a lot with understanding and developing some basic muscle memory for git. Perhaps lecturers should emphasize that everyone should take their time and read Pelita docs carefully. You may think this is a no-brainer, but at the same time, it is counter-intuitive to have 6 of us sitting in the same room without any interaction, reading the docs.

    To me, the Pelita structure was quite confusing at first, especially the naming convention for "Bot" and "State". Bot is holding all information about the game*state*, which only makes sense after you realize that some aspects of the gamestate are only "visible" through the perspective of one of the bots.
14. Great fun and perfect playground to apply the topics of the course!
15. I loved the idea a lot. The game and the overall task is designed very well to maintain perfect balance between fun and challenge. Programming can be exhausting when you have to develop a massive system, think about every detail and every edge case. This project was designed in such a way and this mostly boiled down to the fun part.
16. The programming project was fantastic and the highlight of the school. I'd like to thank the organizers for putting so much heart in it!

The School in General

Q: How do you overall evaluate the school?

Good: 96% (27)
Neutral: 4% (1)
Bad: 0% (0)

Q: How do you evaluate the general level of the school? Was it too advanced/too basic with respect to* your expectations*?

Too advanced: 7% (2)
Just Right: 79% (22)
Too basic: 14% (4)

Q: How do you evaluate the general level of the school? Was it too advanced/too basic with respect to* what was advertised in the announcement*?

Too advanced: 0% (0)
Just Right: 96% (27)
Too basic: 4% (1)

Q: Did you learn more from attending the school than you would have learned from reading books, online tutorials or AI tutoring alone?

Yes: 93% (26)
No: 7% (2)

Q: How do you evaluate social interactions and social activities at the school?

Good: 93% (26)
Neutral: 7% (2)
Bad: 0% (0)

Q: Would you recommend this course to other students and colleagues?

Yes: 96% (27)
No: 4% (1)

Q: How did you hear about the school?

Google Search: 1
Professor/Tutor/Supervisor: 8
Colleague/Friend: 16
Website/Mailing list: 9

1. Mail in my Institute
2. LinkedIn
3. mailing list of my institute
4. BCCN
5. Mailing list from Institute
6. Bernstein Network
7. t.me/bioinf_career
8. Telegram groupchat for Russian bioinformaticians
9. BIOINF channel in Telegram

Q: Any further comments or suggestions?

1. Keep up the good work ????
2. You guys were great. And although you manage to find funding, maybe you could reach out to old alumni before the next edition, I think small donations could be a good way to try and fund at least some small things, like socials, lunches, etc. Maybe worth a shot :) Thanks again for everything! :)

3. Just again more attention to the groups and the pairs. Highlight more often how they should be balanced, how more experienced people should be willing to help out the less experienced one and how there should be honesty about one's ability to avoid imbalanced pairs and groups.
4. Keep up the amazing work!:
5. It was incredible!

   The first warm-up event could have benefited from a very brief structured group warm up. So everyone hears everyone else at least once before everything starts, maybe in a fun collaborative game.

   This course taught me so much more than best practices for scientific programming, I genuinely learned a lot about myself and how I work with others, some of us learned to be good managers, some learned to be better team players, some got humbled and some got more confident and it was also possible to learn a lot just observing these things happen all over. Faculty was amazing and everybody brought their own touch. Besides the content of the lectures, I learned all sorts of useful bits and pieces that will help me in my work, and I will definitely integrate some new tools into my workflow.
6. Everything was extremely well organised. One can tell the school has a long history and that everything has a purpose because you have iterated many times and improved all aspects. I really appreciate the dedication of the tutors and the fact that they also do it just for the fun of it. I can see it is a very nice community.

   One personal suggestion, I don't love live coding sessions. I appreciate seeing only the bare minimum code on a slide or perhaps a jupyter notebook that is shown sequentially, no jumping around to explain things, that is dizzying and increases cognitive load a lot.

   Slides with visual representations where possible is the best way to learn for me.

   That said, I did like Tizianos live coding with git, so perhaps there is a way to do it. Those sessions were very slow paced. Tiziano would write one line per 10 mins.
7. I really enjoyed ASPP and learned a lot! Thanks to the entire team! :-)
8. It was impressive how much personal involvement the faculty and organizers showed. I have never been to a similar event where that was so evident. Also, there was no hierarchy between faculty and students, which created a very nice atmosphere. Thank you so much!
9. I am so glad that I got to attend ASPP - it was such an awesome experience!

   I hope many more people get the chance to participate in future editions!!
10. ASPP is the most useful summer school I have ever attended. After the school, I got very motivated to organise my code in accordance to the best practices, and I knew how to do it.

    Also, I surprisingly enjoyed pair programming, I think, the credit here goes to the great work of tutors.
11. ASPP 2025 was without a doubt one of the best events I've participated in so far. It was physically, cognitively, and socially exhausting. Yet the vibe was so good that you wanted to keep going anyway. I really appreciate the effort the tutors put into preparing the course materials, especially for the final project: the Pelita game. I definitely recommend this event to scientists who want to strengthen their programming skills!
12. This was an invaluable experience, many thanks to the people involved in making this happen!
13. Organisation was amazing!! (Thanks verji!! <3)

    Maybe only regret was to have little time to visit plovdiv... Organised lunch was great, but also a sandwich around would be great too, more as an excuse to have a walk, move a bit (the lecture day is long and we sit all the time) and visit the place
14. The school is great. Thank you for your contribution to a better coding culture in science.