ASPP2024

Tübingen AI Center

FORTH
INSTITUTE OF MOLECULAR BIOLOGY & BIOTECHNOLOGY

16th Advanced Scientific Programming in Python

Summer School

25 August – 1 September, 2024. Heraklion, Crete, Greece

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 25 September, 2024.
No answer was mandatory.
The free-text answers have not been edited and are presented in their original form, including typos.

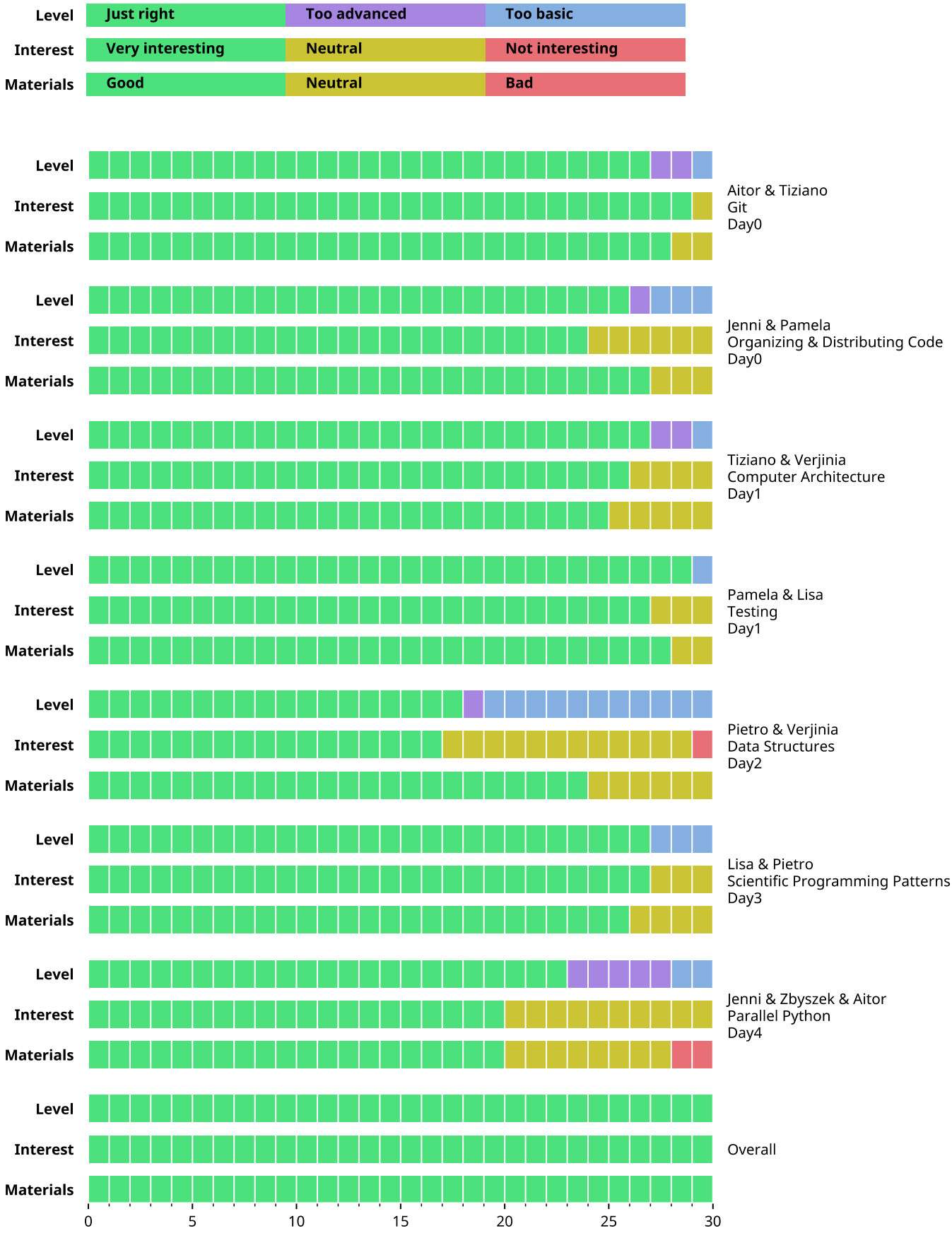
Attendants and Applicants Statistics

	Attendants		Applicants	
	30	19%	141	
Different nationalities	16		39	
Countries of affiliation	10		29	
Gender: other	1	3%	4	3%
Gender: female	14	47%	72	51%
Gender: male	15	50%	65	46%
Already applied	11	37%	26	18%
Bachelor Student	0	0%	7	5%
Master Student	0	0%	21	15%
PhD Students	23	77%	87	62%
Post-Docs	3	10%	10	7%
Professor	1	3%	1	1%
Technician	0	0%	3	2%
Employee	3	10%	9	6%
Others	0	0%	3	2%
Completed surveys	30	100%		

More stats about attendants are available at: https://archives.aspp.school/2024-heraklion/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. the parallel is less relevant to me
2. I think the "computer architecture" lecture was really interesting and well-taught but I am afraid that it might not need to be so extensive.

   If you can make people understand that numpy does better than regular python for loops, I think its enough.
3. although both Numpy and Pandas are relavent for scientists and python in general. I believe that a lot of it is already understood by most in attendance. Certain specific topics such as broadcasting, indexing, and the speed of for loops outside of numpy were all interesting and useful for me, but I think that this can be summarized in less time than a complete morning lecture. Alternatively, I think that other introductory material online can teach the basics of numpy and pandas. That being said, I think that the use of real data for the exercises was a very good touch and made the lecture applicable for us as scientists.
4. No, I found everything to be relevant at least potentially.

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. To my opinion, two topics which could be added are decorators and the different types of class methods by extending the 'Scientific Patterns' course by one hour. To make room for this extra hour, I would consider reducing time spent on some introductory topics in the OOP course, as most students might already be familiar with local imports and virtual environments.
2. Profiling, which used to be there,

   Creating documentation (all the steps including deployment online)

   How to write good documentation and good API

   What could be replaced: pandas
3. If i got it correctly, the numpy lecture used to be more extensive and more advanced. I think I would like to learn more about advanced numpy and pandas to get rid of unnecessary python for loops rather than having the computer architecture lecture.
4. 1. The parallel lecture was too basic and all interesting details/particularities were left out on the last slides and basically were not discussed.

   2. On the computer architecture lecture it would be interesting to discuss the precision of standard numpy types: float32, float64 and the caveats of using np.float128 and what that means.

   3. On data structure lecture it would be nice to discuss more details about build-in python types, e.g. hash tables for discs and so on.

   4. For testing lecture it would be nice to give more time not to pytest but to debugging in IDEs, setting up the debugger and using some exercise for this.

   5. The pattern lecture was a bit vague. It would be nice if it was beyond just about introducing the usage of classes.
5. Something more about where to store data? More info on Research Data Repositories (e.g. Zenodo) and how to link them with Github/lab etc?

   I think the topic naturally arose in the data class. Maybe it is possible to squeeze in this topic there?
6. it's certainly a biased suggestion ;) but I will add image processing (maybe with dask).

   maybe shortening computer architecture
7. For me, a generic intro to tensor-flow or using numba would have been more useful for me than a review of numpy and pandas which I already use daily. I also believe that the documentation for numpy and pandas is very easily understandable and other online sources do a good job explaining these topics to new scientists. Using tensor-flow or numba would be more useful for me since a lot of packages require both packages but I do not really understand why they are used or how to use them independently of the packages that are dependent on them.
8. Maybe switching between cpu and gpu, and more on working with the cluster. I know these depend on the services the home institute provide, but I guess I could use more info on how to navigate when your own computer is not powerful enough for the job.
9. I would appreciate hearing about profiling. I know that was previously included. Pandas might be replaced with that, or the OOP part in the scientific programming patterns lecture.

10. The one skill used by programming scientists that was not covered is creating scientific visualizations. Tools like pyplot can be a bit puzzling at the beginning if one wants to have proper control over visualizations; a speedrun of the package logic could solve this. HOWEVER, I do not really see any topic worth sacrificing for this.

    (Personally I could have done without Pandas and most of numpy lectures, but I understand that if someone is not already familiar with them, they are indispensible tools to learn).
11. A small sub-module on logging would be nice. It could be added to debugging or testing.
12. Nothing comes to my mind, really.
13. I can't think of any that would be broad enough to be relevant for programming scientists of all fields. A topic like working with PyTorch / Tensorflow would be interesting for me but I am sure it would not be that interesting for everyone.
14. I really cannot say what I would leave out, but if you had to drop something, I think that Context Managers and Exception handling could be an interesting addition
15. Git could get more attention.

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

Yes, I have learned from my partner / I have helped my partner	90% (27)
No, it was a waste of time for both me and my partner	0% (0)
Neutral. It was OK, but I could have worked by myself as well.	7% (2)
Other	3% (1)

Other:
1. Yes, I do think it was useful, but I found it extremely hard to get my thoughts into code while also having to deal with someone else's thoughts simultaneously within the limited time we had for exercises.

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	2% (7)
Lectures were too short, there should be more time for lectures	0% (0)
The time dedicated to lectures and exercises was well balanced	87% (26)
Other	7% (2)

Other:
1. There should be less but more advanced exercises that then hava enough time dedicated to them
2. I think lectures were just long enough but wish we could also complete all the exercises like we did on the first day.

Q: Any further comments about the lectures and exercises?

1. sometimes the time for an excercise was too short -->maybe fewer exercises but longer or fewer topics

   If the time is so limited and I did not finish multiple exercises in a row it was frustrating
2. I think there was a great balance between lectures and exercises, which allowed us to not only understand the topic but to also gain practical experience with the concepts.
3. I am very impressed by how much you managed to cover in such a limited time. The exercises were very well designed and each day I was disappointed when we had to leave the classroom.

   My personal highlight was the computer architecture lecture. The parallel python lecture would be incomprehensible without it. That lecture was very important, but to me this escalated from a very easy example (we spent a lot of time making dakos) to a lot of complexity very quickly, I would have enjoyed more time to take in the second part.

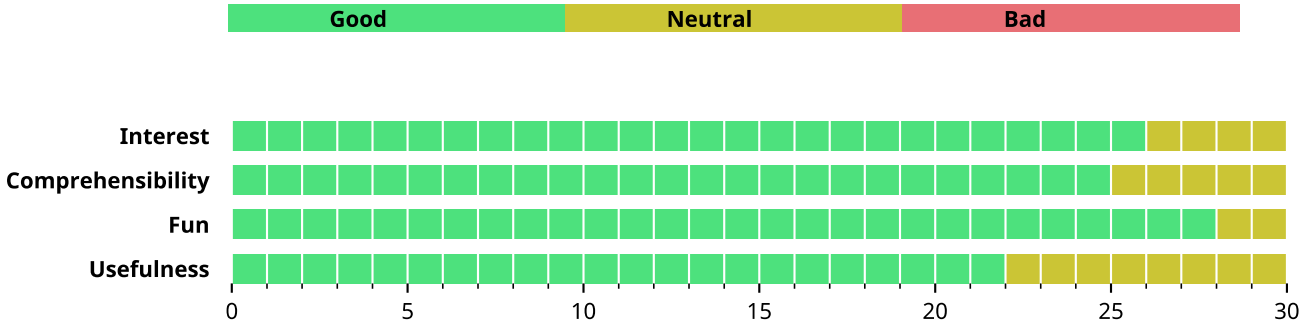
   I also appreciated the first part about data, measuring the complexity of various operations.

4. Pair programming and changing partners was great, not only for learning but it also brought the group a lot closer together
5. I think the git PR strategy for submitting the task is good however, it takes an additional time to do it for people who are not used to use git. Therefore, even though I completed the exercises, there were some times that I couldn't do a PR (mainly because people didn't sign out of their account after switching computers).
6. Many times the exercises at the beginning of the lectures were very basic and did not really teach anything because the lectures were already very comprehensible. Later exercises that were more advanced however then sometimes did not have enough time to finish them properly. I would rather have less exercises late in the lecture but then more time for them.
7. Lectures were read by passionate people and usually organized very nicely but I would personally prefer allocating more time on exercises since frequently it was not enough time for it.
8. Lectures were very informative and well planned. Presenters' style was great! They were all star(t)s ;)

   Exercises were most of the times fun. I enjoyed a lot the ones about brewing potions and in general if they were 'thematic' and chained one after the other. I preferred when lectures and exercises were blend together (i.e., when there was a succession of explanation-exercise-explanation) rather than having a long lecture and then allotted time for exercises.
9. I liked the way the pairs were mixed up for the exercises and the accessibility of the exercises for different levels. They were well prepared and executed.
10. Good balance of lectures and exercises! Excellent slides and exercises (I might steal some):
11. The paired programming in exercises were very useful and I was lucky to find partners that understood exercises that I did not (e.g. classes) and also was able to explain some exercises for people who were not as advanced (e.g. numpy indexing and making copies of arrays). It might not be possible, but I think also trying to pair people based on (perceived) confidence in certain lectures also could be useful to ensure that situations do not arise where both partners are lost on any particular exercise.
12. Quite often, the lectures were not finished due to time limits and this was quite upsetting. However, I like how teachers found the way to finish these parts in the last 30 minutes of lunch time - this was very helpful!
13. maybe a cheat-sheet for every lecture at the end ^.^
14. They were perfectly curated for the budding scientist. It was dense, with a lot of information packed into one week, but the exercises helped me retain most of it. I will be getting back to the material time and time again while rebuilding my analysis pipelines according to the best practices I learned.
15. All the lectures and exercises were really high quality, they were challenging and I learned a lot. :)
16. Lectures were super helpful and relevant for my day-to-day work and programming skills. The lectures were often funny and very vivid. It was a lot of fun and I learned a lot! Moreover, being able to apply what I had learnt in exercises right after the lecture was very helpful for the learning process. Thank you so much for all the great input!
17. Generally, I found the material very relevant and tasks useful. Stuff that I found "too advanced" (which was not "too advanced", btw, but just more advanced than others) I would definitely need to go through slowly again by myself.
18. Although there was limited time for exercises, I think it was definitely well-balanced with lecture time, because the material was always clear enough and made available in case you would want to come back to a specific exercise and try again or perfect your answer after the lectures. I would not want to take time away from the lectures - they were never boring and in this way there was always time to really go in-depth when answering questions from students during the lectures.
19. I think the balance between exercises and lectures was almost perfect and in general exercises were well thought out and useful for fixing concepts.

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: 87% (26)
No: 13% (4)

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

Yes: 0% (0)
No: 100% (30)

Q: Any further comments about the programming project?

1. I The programming project was really fun and interesting. I applied the concepts from the lectures and doing it with a video game enhances the interactivity
2. It would have been nice to reflect on how the project went within the group and a faculty member. After repo freez, reflect in the group and point out problems and archievements. The speaker interviews were funny but not too productive for the group.
3. I think the programming project was a great experience because it gave a glimpse on how programmer teams work in real life. I really liked several aspects of it, especially the idea of setting rules on how individuals should behave as part of the team, how teams are organized and how work is distributed among the members of the team.
4. I wish the preprogrammed classes/functions were accessible, e.i., it had been possible to initialise the bot object to check its attributes/methods without running the game.
5. I believe it's very hard to work on a project in groups of 6. I believe it would be much better with a group of 4 or 2. Nonetheless, It was an interesting experience to see how chaotic things get :).
6. Working on a real scientific project instead of the video game could have been more sensible in terms of scientific programming patterns and the algorithmic part of it. However, I think the school should also be about fun and teaching people that coding can be fun. I would keep the video game.
7. I think the time limitation should be highly stressed in the beginning. The most fun part (for me) was implementation of different ideas from our team for "bot intelligence". However, we somehow misjudged the available time and spent too much time on discussing organisational aspects (and not using them at the end) and ended up not implementing plenty of nice ideas.
8. It was so much fun and useful at the same time that I am going to propose it to colleagues and friends as well!
9. I liked it a lot. IN terms of structure, I was hesitant at first about the two half days (instead of switching fully to the project on the Friday) - but in fact it balanced the project in relation to wider learning well and made it easier to see the project as a learning experience. I liked how balanced the groups were and the level of support from the tutors was excellent!

10. I think that working on a video game is better than scientific project. Indeed all the members were really engaged in this project (while with if the science is far from their experience they might not have been so engaged). Even if I won't apply directly what we have done in this project, it was an excellent real case of using git in a collaborative project and we used what we have seen during the course. Finally, the short time and the competition created some pressure which made the overall experience truly fun !
11. My team's approach to the project was to divide the work into different components. This meant that I mostly did statistics and text for the bots. Unfortunately, I have to begin using networkX in my work in lab, so a bit of understanding of that would have been nice as well. But I think that this was more due to my group's construction and what others were comfortable with rather than any particular issue with the design of the groups. I think that a scientific problem could be useful, but a videogame design is nice because it puts most people outside of their comfort zone and prevents situations where certain individuals are already familiar with a scientific problem whereas others are not. What I can take back to my own working environment is the knowledge of git and making pull requests which will be vital as I try and incorperate new students into my workflow and have them make changes to my existing pipeline. For this reason, I think even without learning more about networkX, there will be a clear improvement in my workflow and version control moving forward.
12. The project was fun and I would like to return to it from time to time to recap some practises.
13. maybe the teams for the project should be formed after the first 3 days of the lesson and a self assesment in order to categorize students in groups relevant to their coding skills
14. The chance to use the best practices we learned during week was the most relevant part of the project. It was engaging because there was a competition and everyone likes to win. I found it interesting to observe what role everyone takes on during a team programming project, since during phd most of us usually code for and by ourselves.
15. I enjoyed it a lot, and I think it was an awesome learning experience, though pair programming at that point might not be the only good work framework, releasing expectations concerning pair programming at this phase of the school might be useful.
16. The programming project was a lot of fun and the perfect end to the summer school lectures.
17. The project itself is really fun and helps break out of the scientific coding bubble and gives the chance to use the tools just learned. My only comment would be that it might be worth thinking about more directly encouraging/rewarding the use of good practices.
18. It was amazing! The whole setup was impressive and it was rewarding to be able to work on the intelligent part of the code with all the graphics, structure and demos already set up. The only minor comment is that you could introduce the graph structure which is very relevant for the project in advance, so that everyone is on the same page.
19. I think it was really fun and engaging, and I enjoyed working in a team a lot (which made me think that I like programming more when it's done in teams), I wish we had more of this in academia (but sometimes it is team programming, even if it's future and past you). Maybe I wish we had some kind of feedback session at the end (like, what were the common pitfalls/bad programming practices and how to avoid them), but I guess it was also nice to finish on a high note of the tournament.
20. The things I learnt during the programming project are probably not directly applicable to my current research because I almost never work with several programming scientists on one project,but I am sure that I will remember this experience and the lessons I learned if I ever find myself working in a high-paced environment working on a programming project with multiple people in the future. It definitely made me a less scared of working together on the same programming project than I was before.
21. I think that the choice of a videogame makes the whole exercise way more interesting and creative. It also ensures that there is no initial advantage in the competition and improves the collaborative aspect of the project.
22. Such fun! Seriously, I learned a ton.

The School in General

Q: How do you overall evaluate the school?

Good: 100% (30)
Neutral: 0% (0)
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: 83% (25)
Too basic: 10% (3)

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: 3% (1)
Just Right: 90% (27)
Too basic: 7% (2)

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

Yes: 97% (29)
No: 3% (1)

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

Good: 100% (30)
Neutral: 0% (0)
Bad: 0% (0)

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

Yes: 100% (30)
No: 0% (0)

Q: How did you hear about the school?

Google Search: 4
Professor/Tutor/Supervisor: 7
Colleague/Friend: 17
Website/Mailing list: 5

1. utelegram channel which posts different schools/internships, etc (it's in Russian :D)
2. Mailing list of our building
3. Twitter: https://twitter.com/BrainCourses
4. BCCN mailing list
5. https://docs.google.com/spreadsheets/d/1nezjxkU8kGsA9MUI3Eph60s303VwfQLqMSNrSnnRSzk/edit?gid=1015366364#gid=1015366364

Q: Any further comments or suggestions?

1. Thank you for this amazing experience!
2. I am very grateful to ASPP staff for this summer school. You have achieved the seemingly impossible of squeezing such a large material in just one week. It would have been very tempting to just give very long lectures, but you designed ingenious exercises to let us get the point on our own. The immersion was very good for learning. Initially I didn't think pair programming or working on separate laptops were of much importance, but I am now sold on it and now I miss working on code in a pair. Thank you for all the work you've put into teaching us.
3. Loved the whole experience, thank you all for doing this!

4. This was a fantastic experience! It allowed me to fill the gaps in programming project management and to unlearn the habits that would slow me down. If anyone ever asks me what class they should take to boost their scientific programming practices, I'll forever recommend ASPP!
5. ASPP is outstanding in merging high-quality lectures, hands on exercises, and fun times with cool people. It gave me an unforgettable experience and made me excited about applying what I've learned in my projects now.
6. ASPP rules!
7. Just a big thank you - I want to come back and do it all over again!
8. It was amazing, thank you for the great work and thought all of you put into this!
9. I loved that summer school and I wish I could do it again! Keep up the good work, you're all Starts!
10. I found the school very balanced in terms of difficulty, time scheduling, interaction with students and teachers!
11. It was a great experience with just the right amount of work and play :)
12. In general, I really enjoyed the school and learned a lot, I'm grateful that I could participate. I already feel more confident about programming and distributing issues, and I'm excited to turn into practice the learned knowledge in my everyday work.

    One minor point: I went through the prerequisites material very conscientiously, though with git, I felt, that the lecture would have been much more digestible if I had reviewed those materials in the software carpentry webpage, which weren't included in the prerequisites material (e.g. chapter 7. and above), thus I suggest to include a few additional chapters to the prerequisites.
13. ASPP was an amazing experience! I learnt a lot and met super interesting and nice people! It was really fun and I am very grateful that I was able to take part. Thank you to the organisers for this wonderful summer school in Heraklion!
14. The course does a great job at introducing scientists to standard programming practices, and even when topics are not detailed I found it still useful to receive a first priming of issues and tools I had never considered. I would have liked, however, if more time were dedicated to transferable skills like packaging and testing, even sacrificing one or two other topics (e.g. parallel python, which is very interesting but not frequently used in my experience).

    This being said the general course organization is really great, from the hands-on teaching style, to the stress on collaboration, to the plenty of chances to just mingle and network.
15. This was one of the most useful weeks of my academic life. Thank you :)
16. Thank you very much for this opportunity! I learned a lot, and as someone who's always unsure in their programming skills, and felt the urgent need to improve them, - this was a perfect match! Also, I really appreciated a very warm atmosphere - it felt safe to ask "stupid" questions - which in my experience was not always the case with programming courses I took before. Normally, programming is one my least favourite parts of the scientific process, and it was great to see that it can be done less painfully (and more fun, especially, when done in groups). I also want to say that I really appreciated the group - both students and instructors, it was great to see a fairly diverse team (especially, in instructors - no "tech bros" mansplaining vibes!). Thank you again, and I hope the summer school will thrive for many more years to come! I'm already recommending it to all of my friends and colleagues.
17. ASPP was a very valuable experience for me. I learned very much and became more confident in my abilities as a programmer in general, and in my abilities to create reproducible software specifically. This in turn made me more confident in my abilities to conduct reproducible and valuable research. Meeting researchers from other fields during the school and noticing they are struggling with similar issues, for which apparently there are plenty of solutions, amplified this. There is no way I could have had the same experience learning from books or online courses. The teachers are incredibly motivated, driven and fun. Thank you!
18. The school was outstanding, and easily the best "academic" experience I had! A small suggestion: add more options for the answers to the evaluation questions. Every aspect of the school was comfortably in the "good" range, so a scale from 1 to 10 or at least 1 to 5 would have been better.
19. Keep up the great work, I would have loved to have taken the course years ago, but it was still super useful.
20. ASPP is really an amazing summer school with a wonderful community of both students and teachers who all really care about what they're doing and about sharing their knowledge and experience with each other. It is such a treasure to meet such a group of like-minded people, and to learn so much while also having so much fun. Given the limited time and the number of topics relevant to programming scientists which could be covered, the organizers do an excellent job of balancing both the number of topics covered and the depth in which each topic is explored. That said, there is so much more to learn, and if ASPP ever offered either (1) more in-depth courses on the topics already covered or (2) a complementary course on the topics that didn't make the first cut, I would be one of the first people to sign up!