Welcome to The Carpentries Etherpad!

This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try https://etherpad.wikimedia.org).

Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

---

Instructor Training 7-10 Nov 2023 CET

Workshop website:https://rabeamue.github.io/2023-11-07-ttt-online-CET/
Instructor training material: https://carpentries.github.io/instructor-training/instructor/index.html

CodiMD Pages: https://codimd.carpentries.org/dCFaHNlxQrKMuN3ltxP53Q
https://docs.carpentries.org/ - Caprentries Handbook

Pre workshop survey: https://carpentries.typeform.com/to/QVOarK?typeform-source=rabeamue.github.io#slug=2023-11-07-ttt-online-CET
Code of Conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

Usefule links:

- The Instructor Training Material: https://carpentries.github.io/instructor-training/01-welcome.html
- How Learning Works Book: https://www.wiley.com/en-us/How+Learning+Works:+Eight+Research+Based+Principles+for+Smart+Teaching,+2nd+Edition-p-9781119861690
- bell hooks Teaching to Trangress book: https://www.routledge.com/Teaching-to-Transgress-Education-as-the-Practice-of-Freedom/hooks/p/book/9780415908085

Day 4 Attendance: Name, email, website/linkedin

1. Jeremy Pike, j.a.pike@bham.ac.uk
2. Jack Atkinson, jwa34@cam.ac.uk, https://jackatkinson.net/
3.Randa Higazy, randa.higazy@uhn.ca
4. Isabela Paredes Cisneros, isabela.paredes@embl.de
5. Michael Milton, milton.m@wehi.edu.au

6.Adam Taranto, adam.taranto@unimelb.edu.au
7.Louise Grimble, louise.grimble@newcasttle.ac.uk
8.Thomas Kiley hi@tkiley.co.uk https://www.linkedin.com/in/thomas-kiley-4004995b
9. Caitlin Bourke bourke.c@wehi.edu.au
10.
11.
12.

Icebreaker: What is your favourite softdrink?

1.Cloudy lemonade
2. Gren tea?
3. Ginger beer
4. Lemon lime and bitters
5.perrier sparkling water
6.orangina
7.
8.
9.
10.
11.
12.

Add your questions here:

-
-
-

Preparing to teach

Examine a learning objective:

Select one learning objective from the episode you've used for teaching practice. Copy it into the Etherpad then add numbers below your objective to address the following:

1.Suppose a learner had mastered this objective, and wanted to try something more cognitively challenging on the exact same topic (i.e. not a next step in a workflow). Identify an objective they could work towards next.

2.Suppose a learner struggled to meet the specified objective. What might they be missing? Identify one more fundamental thing a learner needs to be able to do in order to be successful in meeting this objective.
This exercise should take about 10 minutes.

OBJECTIVE: Delete, copy and move specified files and/or directories (echo 'task complete').
1. Create and run a bash script that combines two of these actions for multiple files
2. Research the meaning and write a definition of the following terms (ls, cd, mkdir, mv, rm) | open up gui file finder side by side to help understand actions made in the episode. (depends on student issues)

OBJECTIVE: Correctly write for loops to repeat simple calculations (python).
1. Intoduce looping through key, value pairs in a dictonary and performing calulations on these values
2. They might be missing that the commands within a loop need to be indented or they may be stuggling with prerequisites such as lists. They need to understand the basics of lists, the syntax (correct way of writing) for loops in python, and also the general loop concept of repeating actions and iterating through items in a list.

Objective: Correctly write for loops to repeat simple calculations.
Student has mastered this:
Find out if a string is a palindrome (spelt the same forwards as backwards)
Student is struggling with this
They need to understand that the same variable hold different values at different times in the program execution (that is the loop variable will change value over the course of the loop).
They need to understand Python indentation rules, and be able to see if they have idented the loop body

Objective: Use options and arguments to change the behaviour of a shell command.
1. Use the man pages or --help to identify more options beyond those explored in the session and investigate them.
2. Incorrect order of arguments to a shell command e.g. placing options after filename. User needs to understand that the order of options matters, how to interpret the error messages, and where they can find information about available options and ordering.

Objective: Trace changes to a loop variable as the loop runs.
1. They could come up with a function that executes something else using the loop variable, besides printing
2. They might be confused by the use of a loop variable, maybe going back to the part we explain the structure of the for loop with the diagram, to understand how the variable iterates over the list

To be able to read a traceback, and determine where the error took place and what type it is.
1. Try to understand exception chaining, ie raise e from f
2. The learner may not understand the concept of the call stack, namely the way in which each function call adds another layer to the Python execution environment, which is reflected in a traceback

Objective: Write a SQL query that retrieves data from a database table and performs a basic data transformation
1. Create a complex SQL query that involves multiple JOIN operations and aggregates data from multiple tables to generate meaningful business insights.
2. The learner might be missing a fundamental understanding of SQL syntax and database concepts such a writing SQL query that selects all records from a single table without any data transformation.

Objective: Go through the modify-add-commit cycle for one or more files.
1. Advanced: Adding multiple files to a commit. Unstaging a file accidentally added to a commit.
2. Struggling: Has student saved changes to the file? Is the command spelt right, are they getting a useful error message?

Assessing Progress Towards an Objective

Objective: *"Create files in that hierarchy using an editor or by copying and renaming existing files"* (https://swcarpentry.github.io/shell-novice/03-create.html)
How would you know that learners have reached this learning objective? What would you expect them to be able to do? How could you check that they have learned this?

Room 1 (Adam, Caitlin, Jeremy)
- Be up to "Removing files and directories" section
- Expect: know cmds - mkdir, cd, nano, cp, mv

Room 2 (Louise, Michael, Thomas)
- Create a multiple choice style quiz (via something like Kahoot) to get live results and showing distribution give questions such as move file X to directory X,

Room 3 (Isabela, Jack, Randa)
- create new directories within that file hierarchy
- copy existing files and directories from one location to another
- delete files and directories and create new files with specific names using a text editor
- Provide a layout diagram and ask students to replicate it.

More practice teaching

Practice Teaching part 2 (25 mins)

1. Split into groups of three.
2. Assign roles, which will rotate: presenter, timekeeper, note-taker.
3. Have each group member teach 3 minutes of your chosen lesson episode using live coding. For this exercise, your peers will not “code-along.” Before you begin, briefly describe what you will be teaching and what has been learned previously. Do not record this exercise.
4. After each person finishes, each group member should share feedback (starting with themselves) using the same 2x2 rubric introduced in part 2 The timekeeper should keep feedback discussion to about 1 minute per person; this may leave some time at the end for general discussion. The note-taker should record feedback in the shared document.
5. Rotate roles.

Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

Try this next time:

Room 1 (Caitlin, Louise, Randa)
Louise- keep doing this: clear and good pace
try next time: typing was delayed from the explanation (type then explain command)
try next time: recap a few key terms like directories etc.

Caitlin - keep doing this: recap is good of previous lesson of what content was covered
try next time: recap of some of the concepts from the previous lesson

Randa - keep doing: pacing
try next time: use mistakes as a teaching opportunity

Room 2 (Adam, Isabella, Jeremy)
Content
Keep doing this:

- Jeremy: Senisible variable names, nice adaptation from the content
- Isabela: Good pacing and working though mistakes/process with audience
- Adam: Very clear and easy to follow / understand. I actually liked the brief tactical pauses when you were checking notes as gave me a chance to process what you had just shown me.

Try this next time:

- Jeremy: Avoid over-writing code
- Isabela: Famiarilize ourselves with all enviroments/setups partipants may choose to use
- Adam: Careful useing different names for same thing e.g. terminal/shell

Performance
Keep doing this:

- Jeremy: good pacing
- Isabela: Asking students to help identify an error. Relating content to previous lesson.
- Adam:

Try this next time:

- Jeremy:
- Isabela:
- Adam:

Room 3 (Jack, Michael, Thomas)
Content
Keep doing this:

- Thomas: using individual print statements before loop to link back to old content

Try this next time:

- Thomas: Don't introduce the graph narrative at the start as it was confusing

Performance
Keep doing this:

- Thomas: Ask the audience a simple question to keep them engaged

Try this next time:

- Thomas: Don't say Python construct as is general to all programming languages

Reflection (5 mins)

- What did you change?
    - I made a better use of the Jupyter notebook, saved some time by reusing what I had already put in there
    - Was careful not to copy any code from the material
    - Was more conscious of not speeding up during a mistake to 'try catch up'
- What worked better or worse with the change?
    - I had more flexibility and opportunities to work with the code, maybe closer to what I would do when actually coding in Python, hopefully more relatable and useful demonstration
    - By writing out all the code manually, I made more mistakes, and took longer to do so. This made it more difficult for me, but probably made it more relative and real for learners
    - tried to explain something further but gave an error so then I got flustered (so was almost good and bad)
- How might you do it if you were to teach it again?
    - type out commands either before or as I explain it not after I have explained it
    - Remove all unnecessary jargon such as "call stack" from my explanation

Sticky Situations 1

What are some of the challenges you might expect when teaching learners with a broad range of expertise?

- Keeping everyone with varying abilities and objectives engaged

- An individual feeling uncomfortable asking for help if they believe they are the only person who is struggling / they feel like they're slowing down the class
- More advanced students getting bored waiting for the rest of the class
- Learners getting confused if they have already learnt a different concept, something that might be contradictory, or have already become familiar with certain practices

- Students with a compsci background derailing the class and demotivating students with monologues about Python being "slow" and Fortran/C++/Assembly/Punch cards being a better/truer medium for

software development. Or worse, two of them getting into a feedback loop!

Learners whose environments are causing errors. For example using a newer/older version of Python/packages, or having a different operating system that behaves differently.

Sticky situation 2: Code of Conduct Violations

Activity: Know Your Resources
Take 5 minutes to read through the Code of Conduct Incident Response Guidelines:
https://docs.carpentries.org/topic_folders/policies/incident-response.html
1
Discuss what you have read in small groups. As questions arise, you may wish to refer to our complete Code of Conduct section in The Carpentries Handbook:
https://docs.carpentries.org/topic_folders/policies/index_coc.html or to the Transparency Reports released by The Carpentries Code of Conduct Committee:
https://github.com/carpentries/executive-council-info/tree/master/code-of-conduct-transparency-reports

Room 1 (Isabella, Jack, Thomas)

What kinds of things could your instructional team agree upon in advance of your workshop?
Who will be responsible for overseeing CoC matters in the workshop (probably not the main instructor)

What questions do you have about CoC enforcement?
Some examples of different situations and how to respond would be helpful.
'The person who violated the CoC does not apologise' is surprising, but understand why as some may not want to hear this or be sincere.

Room 2 (Adam, Jeremy, Michael)

What kinds of things could your instructional team agree upon in advance of your workshop?

What questions do you have about CoC enforcement?

Room 3 (Caitlin, Louise, Randa)

What kinds of things could your instructional team agree upon in advance of your workshop?
Discuss with the teaching group what the expectations are. Review the code of conduct together.
Encourage a team effort to identify breaches of the conduct so that everyone is on the lookout to make it a safe learning environment

What questions do you have about CoC enforcement?
What would be the best way to handle dismissing a disruptive person in-person and or zoom

Launches and Landings

What is an introduction?

Get into small groups (3-4 people) and discuss the questions below. Take notes on your answers in the Etherpad.

1. What do you hope to accomplish in a workshop introduction?
2. What information do you need to include in an introduction to accomplish these goals?

After 5 minutes, come together, and combine ideas as a large group.

Room 1 (Caitlin, Jack, Randa)
Overview of expectations, and set out what people want to get from the workshop
Set a friendly and comfortable learning environment

Introduce selves (all instructors/)
Get participants to introduce themselves*
Introduce key objectives/subject

Outline different ways people can ask/get help

Explain the concept behind live coding and the structure of the workshop/lesson

*Question - Do carpentries have limits on the numbers who can participate, as introductions could get long.

Room 2 (Adam, Isabela, Michael)
To accomplish with the Introduction:
- Clarify: why are they there? What's the workshop content, what they will be able to do after the workshop
- Feeling transmitted to the learners: achievable, reachable goals, motivated to do the work

Information
- Mention The Carpentries' style of teaching: CoC, inclusivity, environment, etc.
- Explain system to give feedback: e.g. sticky notes
- Workshop goals

Room 3 (Jreremy, Louise, Thomas)
Encourage participation and making people feel relaxed
Reiterate that it is a course for novices so questions always welcome
Practical content: when are breaks, where are the toilets etc
Motivation by sharing context for the content
Introduce demonstrators and how they should communicate / ground rules for workshop

Draft your Introduction (optional share)

Imagine you have completed instructor training and you are about to teach a full lesson around the material you have been practicing teaching during this training.

Write out some notes, covering a few of the topics discussed:

1. Introduce yourself effectively
2. Clarify learning objectives and expectations
3. Set the tone for the workshop

Making the last moments counts

You have made it to the end of your workshop! Everyone is exhausted and their brains are full. You could cover more content… or you could use the last few minutes in another way.
In the Etherpad, write down one thing you could do at the end of a workshop.

- Ask them about the three take-home ideas they got from the session
- Summarise what they have learnt so far, highlight the usefulness of that new skill and reassure them, encourage them to go ahead and use it

- Have everyone to write down one thing they have learnt in the session

- Remind people where they can find resources from the day and how/where to learn more.
- Provide some examples of the possibilities/ application of the skill they have just acquired i.e. additional courses, certain careers or even fun things to do like creating a game or algorithm for something. (something easy to listen to but still relevant and informative)
- Formative assessment (quiz) on some key topics
- Request feedback from learners`

Make sure they are aware of th best places to find more information and progress further with their own learning after the course

Relate the content back to their own data/research, how will they use what theyve learned.

Go over a more high-level summary of what has been covered. Ask the students to think of how they could put some of the new skills to use in their everyday work tasks

Picking up the pieces (5 mins)

Based on the content we've discussed throughout this training, add at least one item to each category below:

- **Concepts/Theories**
  - mental models
  - Categories of learners
  - Usage of language, and how so often-use words can discourage learners
  - Formative assessment and finding misconceptions
  - Code of conduct
  - Cognitive load
  - Accessibility
  - Concept maps

- universal design

- **Tools/Practices**

- Formative assessment
- post its
- one up one down
- Preparing an introduction
- 2x2 feedback matrix
- participatory live coding
- Replicate learner's coding environment +1
- Pre/post survey
- Target audience is the novice
- Teaching requires practise
- How to give and receive feedback and to reflect on any feedback received
- Careful of language

Organise your knowledge (5 mins)

Use a concept map or other visual organiser of your choice to connect some of the concepts above.

You don’t have to use them all!

How are the terms you have chosen to include related to each other?

Work on this on your own.

There is no “right answer” – this is about you building up a mental model, moving from “novice” to “competent practitioner”.

One Up, One Down

Provide one up, one down feedback on the entire Instructor Training course.
Remember:

- Say only one thing, and try not to duplicate. This gets harder for those who come later!

Keep doing:
1. Creating a friendly and inclusive environment / vibe, both instructors did this very well but with their own style
2. Getting practise of live coding with feedback
3. Continuously asking for feedback and addressing queries - esp having someone to monitor the chat.
4. Having breakout rooms with the same people helps
5. Having Toby available to do sidebars in the etherpad +1
6. over 4 shorter days made the amount of content more manageable and easier to digest
7. Keeping the time for breaks and encouraging us to move!

8.Discussions around how what we are learning could be applied practically
9. Giving us our own choice of episode to present

Think about for next time:

1. It would be nice to know in advance how we plan to work with the Etherpad, since it was a bit confusing
2.There was a diagram of the carpentries course that was far too small to read - either a simplified version or not showing it at all would have been clearer
3. Content wise perhaps more focus on the practical aspects of setting up a workshop like making the website, it would be good to fit this in the course if possible but I get theres already a lot to cover.
4. Changing instructors midway through the course was a bit jarring as each had a different teaching style
5. Too many external links to Carpentries docs with too many sub-sections. Would be good to have a **minimal** handbook of useful elements from this training that would directly carry over to running our first workshop. i.e. 10 handy ways to gather feedback; 5 examples of formative assessment; Things to check before starting your workshop.
6. Consider having a central website (I know there was one) with key links - I found we had to follow a lot of links which often appeared in different places (chat, etherpad, website, email) which could be hard to track down later on.
7.Potentially a more condensed summary of the learning material shared at either the start of the course or at the beginning of each day- i.e. a slightly expanded version of what we summarised in the last activity might have been helpful (I found all the links and different places to find information a little confusing at times)
8.
9.

Post-workshop Survey

Assessment is very important to us! Please take the remaining time to complete this ~5 minute post-workshop survey (https://carpentries.typeform.com/to/cjJ9UP#slug=/).

Day 3 Attendance: Name, email, website/linkedin

1. Isabela Paredes Cisneros, isabela.paredes@embl.de, https://www.linkedin.com/in/isabela-paredes-cisneros/
2. Jack Atkinson, jwa34@cam.ac.uk, https://jackatkinson.net/
3. Thomas Kiley, hi@tkiley.co.uk https://www.linkedin.com/in/thomas-kiley-4004995b
4.Alex Southgate, southgateJA@cardiff.ac.uk
5.Randa Higazy, randa.higazy@uhn.ca
6. Michael Milton, milton.m@wehi.edu.au
7. Jeremy Pike, j.a.pike@bham.ac.uk
8. Adam Taranto, adam.taranto@unimelb.edu.au
9.Caitlin Bourke bourke.c@wehi.edu.au
10.Louise Grimble, louise.grimble@newcastle.ac.uk
11.
12.

Icebreaker: What is your favourite breakfast food?

1. normal breakfast: peanut butter on toast; special occasion: breakfast burrito!
2. I usually skip breakfast but when I don't, I love eggs + avocado toast
3. Oats/Porage, Coffee
4. Scrambled egg on toast
5.Coffee
6.Avocado toast
7. Shakshuka
8.
9.
10.
11.
12.

What questions do you have?:

-
-

Day 3 Feedback:

Keep doing this:

Alex: Practicing live coding v useful, breakout rooms good
Caitlin: Interactive feedback, discussion of feedback and thoughts is really useful and interesting. Practical examples are helpful to contextualise how we can adopt changes etc.
Jack: Feedback from peers was really useful, especially around language
Michael: Forcing us to do live coding provided some good pressure
Louise:
Randa: Feedback sessions were very helpful
Thomas:
Adam: Live coding with peer feedback. Async question time in etherpad was helpful.
Isabella: I really liked the pace of this lesson
Jeremy: Break out rooms for practice and feedback :)

Try this next time:

Alex: More structure in this document would help me
Caitlin:More breakout rooms / time to discuss. I was a little lost when we were discussing the guidelines for running/organising/structuring etc a course and wasn't sure what the key take aways from this section were.
Jack: Those in a group of 4 rather than 3 didn't quite get enough time. Watching the videos was a bit unclear in breakout - perhaps watch as a group and then move to breakout?
Michael: Maybe breakout rooms weren't needed for some of the tasks?
Louise:

Randa:
Thomas:
Adam: Etherpad is getting mental. More structure. +1 - been struggling to find stuff.
Isabella: For me it wasn't clear every time how this Etherpad was being used. Maybe either fill it in chronologically (newest section goes down) or add new sections at the top
Jeremy: More breaks if possible tomorrow please, I struggled with 15mins over 4 hours.

Live Coding is a Skill

Anticipate the Impact (5 mins)

List some advantages and challenges of participatory live coding from both a learner's and an instructor's point of view in the Etherpad.

Learner
Advantages:
- Can see how code gets written, mistakes and all, by someone who has some prior knowledge
-practical learning- "Learning by doing" +1
- Software development is a practical skill
- Learn what kinds or errors or challenges you might find while in a learning environment and can easily ask for help (i.e. syntax etc.) and start to develop how to deal with these

Disadvantages:
- Can't refer back to previous material
- May feel rushed to keep up
-Can get lost quite easily if too fast and particularly if dependencies/results generated in earlier
- Might seem disorganised
- Depending on structure may lack clear notes to return to
- Ensuring everyone has a similar setup/system is hard

Instructor
Advantages:
-makes the lesson easy for the instructor to teach (clear outlines and structure)
- Don't have to prepare slides in advance
-

Disadvantages:
- Difficult to present abstract concepts and definitions that are not grounded in pure code
- Need very good helpers to make sure students are lost or left behind - challenging to balance speed and keeping everyone together
- can be a bit unstructured
- If not confident with the material, can be more stressful without practise
- Timekeeping!
- Need to be very confident with/practise material beforehand - this can be hard if you didn't write the lesson IME.

Compare and Contrast (15mins)

Watch this first participatory live coding demo video: https://youtu.be/bXxBeNkKmJE and this second demo video: https://youtu.be/SkPmwe_WjeY as a group and then summarize your feedback on both in the Etherpad. Use the 2x2 rubric for feedback we discussed earlier.
In the videos, the bash shell for loop is taught, and it is assumed learners are familiar with how to use a variable, the head command and the content of the basilisk.dat and unicorn.dat files.

Content
Keep doing this:

Try this next time:

- Still be aware of lingo/jargon - especially everyday parlance e.g. '*Home* directory', 'Call stack'
- Consider ommitting unnecessary details to reduce cognitive burden.

Performance
Keep doing this:

- Reasurring language - 'this will inevatably happen'
- Explain what you might expect to happen before executing, and then clarify why the behaviour is different.

Try this next time:

Room 1 (Alex, Caitlin, Jack, Michael)
Content
Keep doing this:

- Explains what/why is hapenning clearly at each step

Try this next time:

Performance
Keep doing this:

- Text is nice and large on screen
- Clearly explains the result of mistakes
- Interactive body language

Try this next time:

- Specifics about bash syntax not necessarily needed
- Sometimes deviated a bit too much from main content?

Room 2 (Louise, Randa, Thomas)

https://youtu.be/bXxBeNkKmJE (Less good example)

Content
Keep doing this:

Try this next time:

- Lack of context from previous sessions
- Didn't explain how to fix problem

Performance
Keep doing this:

- Taking pauses giving people time to catch up

Try this next time:

- standing up and make eye contact, be more engaging

https://youtu.be/SkPmwe_WjeY (Better example)

Content
Keep doing this:

- Appropriate amount of content in the time (didn't rush through content)
- good explanantion of the variable 'filename' can be different.
- Explanation error message coming up twice was good

Try this next time:

- Explaining how the prompt change when you start a for loop in bash (can be confusing)
- more detail in syntax of the for loop.

Performance
Keep doing this:

- Stood up and pointing at the screen was more engaging
- text seemed easier to read (increased text size)

Try this next time:

- nothing to comment here

Room 3 (Adam, Isabela, Jeremy)

Content
Keep doing this:

- Explain that var can be named anything
- Explain why errors happen

Try this next time:

Performance
Keep doing this:

- Standing, gesturing

Try this next time:

-

Practice Teaching (25 mins)

1. Split into groups of three.
2. Assign roles, which will rotate: presenter, timekeeper, note-taker.
3. Have each group member teach 3 minutes of your chosen lesson episode using live coding. For this exercise, your peers will not “code-along.” Before you begin, briefly describe what you will be teaching and what has been learned previously. Do not record this exercise.
4. After each person finishes, each group member should share feedback (starting with themselves) using the same 2x2 rubric introduced in part 2 The timekeeper should keep feedback discussion to about 1 minute per person; this may leave some time at the end for general discussion. The note-taker should record feedback in the shared document.
5. Rotate roles.

Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

Try this next time:

Room 1 (Alex, Caitlin, Jack, Michael)

[name]
Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

Try this next time:

[name]
Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

Try this next time:

[name]
Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

Try this next time:

Room 2 (Louise, Randa, Thomas)

[Thomas]
Content
Keep doing this:

- appropriate amount of content
- good explanation of syntax

Try this next time:

Performance
Keep doing this:

- good pace
- typo resulted in good explanation of output differences

Try this next time:

- less filler words like 'uhm'

[Randa]
Content

Keep doing this:

- good explanation of code

Try this next time:

- had a lack of time due to only having 20 minutes

Performance
Keep doing this:

- Good pace of delivary
- Errors allowed for explanation of code

Try this next time:

- had a lack of time due to only having 20 minutes

[Louise]
Content
Keep doing this:

- Explained the meaning of commands

Try this next time:

- didnt explain that yo can add multiple flags in one (-lrt rather than -l -r -t)

Performance
Keep doing this:

- good pace is first half

Try this next time:

- Slow down pace in second half

Room 3 (Adam, Isabela, Jeremy)

Jeremy
Content
Keep doing this:

- Refreshing the topics covered previously
- Mention the syntax requirements

Try this next time:

- Print each element manually THEN show that you can automate with loop
- Show that var name can be anything

Performance
Keep doing this:

- Good pace

Try this next time:

Isabela
Content
Keep doing this:

- Used typo as learning opportunity

Try this next time:

- Reuse lists in the Jupyter notebook

Performance
Keep doing this:

Try this next time:

- Maybe type slower

Adam
Content
Keep doing this:

- Nice to show how to open the terminal window, taking it step by step

Try this next time:

- In case they don't know what the command "touch" does, then clarify

Performance
Keep doing this:

- Adjust as you go to make it easier for learners (e.g. adjusting the size of the window)
- Nice pace

Try this next time:

- More audience participation

The Carpentries: How we operate

Test yourself!

As a class or in groups, see how many of the following terms you can define.

- Lesson A series of episodes make up a lesson, like a sub-course
- Episode: a section of a workshop that focuses on a single topic
- Workshop: A carpentries lesson led by instructor(s) with live coding of the lesson
- Lesson Program The sequence of lessons that make up a carpentries course
- Instructor: the person presenting the Carpentries workshop
- (Instructor) Trainer: the person training instructors

This should take about 5 minutes.

Community Roles

Select one role from the list below that interests you. Using the the descriptions on The Carpentries community website (https://carpentries.org/community/), write
1)a short definition of the role and
2) a question that you have (or that you imagine someone else might have) about the role. Are there roles you would like to see that are not listed? Note that, too!

1. Executive Council
2. Mentors - Help other members of the community in small groups with developing their skills as instructors, lesson developers and organisers. Do mentors recieve any training for effective mentorship?
3. Instructor Trainers
4. Lesson Developers: Create new lessons for the SWC community
5. Code of Conduct Committee reviews violations of the code of conduct and discusses how the code should evolve to be inclusive for all
6. Instructor Development Committee
7. Community Facilitators
8. Maintainers: keep lessons up-to-date, and handle community issues and contributions. How might a maintainer know when to reject a community contribution?
9. Curriculum Advisors: provide guidance on overall structure of courses bring domain expertise to guide content/datasets
10. Maintainers- make sure lessons stay uptodate and relvent and handle things like merge requests

This exercise should take about 5 minutes.

Instructor Checkout

BE THE EXPERT: CHECKOUT Q & A

In small groups, read and discuss one of the three checkout procedures described on this page: https://carpentries.github.io/instructor-training/checkout

Make notes in the Etherpad:

- What points do you think it is most important or helpful for people to remember?
- What questions or points of confusion do you have, or think others might have? When you are done, report back to the full group about that stage of the process.

This exercise should take about 10 minutes.

Room 1 (Alex, Jack, Louise, Randa)
- Louise pointed out the Incubator - lots of options to contribute that we didn't realise existed
- A couple of people planning to help on workshops already
- Does it have to be a Carpentries session, or can we count codealong carpentries style sessions we have delivered?

Room 2 (Isabela, Jeremy, Michael)
-
-
-

Room 3 (Adam, Caitlin, Thomas)
- Key point: about going to a session to get embedded in the software carpentries community
  Question: not clear exactly what these sessions will be like (will they be attended by lots of people etc)
-How many people are in the call for the demo?
- Do you have to help out at an "official" carpentries workshop to meet the get involved criteria?
- Is there an invite to th Slack channel to find out about community calls?
- Contributing to an existing lesson seems hard - how many typos are even left at this point? Would be good to have a dummy repo with errors to comment on and push fixes for.

Schedule a Checkout Step (5 mins)

Take a moment to review your calendar and sign up for one or more sessions to get your checkout process rolling!

Visit the Welcome Session Etherpad: https://pad.carpentries.org/welcome-sessions-2023
Visit the Teaching Demonstrations Etherpad: https://pad.carpentries.org/teaching-demos

If you would like to attend another community session for your ‘Get Involved’ step, visit the Community Sessions

Etherpad: https://pad.carpentries.org/community-sessions-2023

This exercise should take 5 minutes.

Space for asynchronous Q&A

- Can the episodes be drawn from different lessons?
    - Toby: Workshops combining content from different curricula are usually recorded in The Carpentries database as 'mix-and-match' - we still want to know that you are teaching these events, so please register them as a self-organised workshop.
    - Toby: Overall, my rule of thumb is 'if in doubt, register it as a self-organised workshop with The Carpentries'. But try to remember to tell people when you are teaching parts of

official lessons, and when you are switching over to teach something you developed yourself or took from elsewhere.

- The R workshops use R Studio as an IDE, which is a great way to keep everything students need to be thinking about in a single view. Is it kosher to use a similar approach with the Shell/Git workshops by using the terminal in VScode?
    - Toby: yes, I think this falls well within that kind of customisation that I was talking about on the call: different Instructors will take different approaches to teach the lesson content, and use different environments etc, depending on their audience and their own preferences. For example, some Instructors like to teach Python with an IDE, others using Jupyter Notebooks - this choice of platform is not really dictated within the lesson materials. (There are exceptions, e.g. the DC Image Processing curriculum makes a strong recommendation to teach via Jupyter https://datacarpentry.org/image-processing/instructor/instructor-notes.html#working-with-jupyter-notebooks). I suppose things are less clear in the opposite direction e.g. what if the Instructor wanted to teach the R lessons without using R Studio? Or the Git lesson with GitLab instead of GitHub? I'm not sure of the official answer to those questions (although perhaps I should be!) but on the whole I encourage Instructors to focus on the *skills and concepts* they are teaching, and not to worry so much about the choice of platform etc they are using to communicate those.
- Follow up, if using VScode for git would it be fine to skip the bits when we use nano/vim and just view the file directly? or to include viewing the commit history as a graph as well as from cmd line?
    - Yes, absolutely. You would probably want to replace those with equivalent discussions of how to view and edit files in your chosen IDE.
    - If teaching a centrally-organised workshop, there would be more of an expectation for you to follow the curriculum as it is written. But really the important thing is to provide training that is relevant to the audience (remember the 'authentic tasks' Martin mentioned in the first parts of the training) - if you are teaching a group that already uses VS Code or to whom you think VS Code is more relevant than a vanilla shell, you should run with that. However, if your audience includes a lot of people who are likely to need to work in a terminal after the workshop (maybe they are connecting to a remote server as I used to do often when I worked in bioinformatics), then learning all of this through VS Code may not be as helpful.
    - Cool, so this would be an acceptible minor tweak for a locally organised workshop that could still use Carpentries branding?
    - I would say absolutely yes. Honestly, I do not think we have written down the exact boundaries at which modifications become too much. And The Carpentries benefits from community members using the branding, so please do not interpret all this discussion as us trying to prevent you from talking about us!

Teaching is a Skill

Giving Feedback (10 min)

We will start by observing some examples of teaching and providing some feedback.
Watch this example teaching video as a group and then give feedback on it.

https://www.youtube.com/watch?v=-ApVt04rB4U Put your feedback in the Etherpad. Organize your feedback along two axes: positive vs. opportunities for growth (sometimes called “negative”) and content (what was said) vs. presentation (how it was said).
Note: there is a version of this video with subtitles in both Spanish and English here: https://www.youtube.com/watch?v=jxgMVwQamO0
This exercise should take about 10 minutes.
Add your notes into the etherpad below

Room 1 (Alex, Jack, Thomas)

Content

Positive

-

-

Negative

- Lots of Jargon
- Surface level descriptions of what is being done but not why

Delivery

Positive

-

-

Negative

- Small text on screen
- Rude at start but not very commanding of the class

Room 2 (Caitlin, Isabela, Jeremy)

Content

Positive

- Content was useful

-

Negative

- Slow down the delivery of new concepts, explaining all the technical ter

-

Delivery

Positive

- The trainer seemed enthusiastic and passionate about the content being delivered

-

Negative

- Sentences such as "even users can understand this" or "this is very easy" might put some students down. Try to use more phrases to motivate, and encourage an empathic open environment

-

Room 3 (Adam, Michael, Randa)

Content

Positive

- The double function was a good simple example for learning functions
- Functions are a useful topic to introduce to beginner programmers

Negative
- Obscure variable names

Delivery
Positive
- Used lived coding
- Asked for feedback
Negative
-Snapped at student to sit down
- Too fast
- Not stopping to explain an error he was correcting
- Used jargon terms
- text on screen was too small and hard to read
- said things like "just trust me" and "this should be really easy" which may act as a barrier to learners and deter those who are not sure from asking questions
- Left questions until right at the end

Sharing Feedback (25 min)

1. Individually, spend 5 minutes preparing a 90-second introduction to the topic of the lesson episode you chose before the start of the training course. You will not be live coding.
2. Get together with your group and have one person teach their segment to the group. Keep a strict time limit of 90 seconds per person (one person should be responsible for the timekeeping).
3. After the first person has finished teaching, share feedback. The person who performed should start by offering feedback on themselves. The timekeeper should help to keep feedback to about 5 minutes per person to ensure everyone has time to perform and discuss.
4. Rotate roles and repeat steps 3 & 4
5. Return to the main group and briefly summarize the feedback you received in the Etherpad.

Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

Try this next time:

Room 1 (Alex, Jack, Thomas)
Content
Keep doing this:

- Relate this session to previous content

Try this next time:

- Think about what words are common parlance to us (for, if, do) that may not be common for new people.
- Avoid going into too much detail about technical concept like file vs directory until you get to the relevant part of the lesson

Performance
Keep doing this:

- Enthusiastic but steady/eloquent delivery

Try this next time:

- Use 'may be familiar' rather than 'will be familiar or 'should be familiar"

Ambiguous
- 2 approaches: set out theoretical concepts vs jump in with examples.

Room 2 (Caitlin, Isabela, Jeremy)
Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

Try this next time:

Room 3 (Adam, Michael, Randa)
Content
Keep doing this:

Try this next time:

Performance
Keep doing this:

- Orient the audience by recapping what we just learnt and how next topic follows on

Try this next time:

- Less front loading of terms, instead let terms come up naturally and define as we go

Using Feedback (5 min)

Look back at the feedback you received on your teaching. How do you feel about this feedback? Is it fair and reasonable? Do you agree with it?
Identify at least one specific change you will make to your teaching based on this feedback. Describe your change in the Etherpad.
This exercise should take about 5 minutes.

Isabela:
- Provide a precise example of how the skill they will learn could be applied. I could mention what they will be able to do at the end of the lesson
- Keep pace and engagement

Jack
- Think more carefully about the language used to be fully inclusive (may, not should; could, not should etc.)
- Happy with the feedback.

Thomas:
- Bring in a practical example as motivation in the introduction
-

Alex
-
-

Randa
-using examples to support explanations (e.g., when defining the concept of 'feature engineering' it would have been better to show how that works with an example)
-

Michael
- Linking this topic to the previous topic
- Defining my terms clearly

Jeremy
- If I get in a muddle dont accidently use words like "clearly" :)
- Continue to ask audience to think about why the topic is important for them and their work.

Adam
-
-

Caitlin
- Be hyperaware of the words I use to make sure that they are very understandable for everyone. Link to practical examples when introducing any concept.
- Very fair and useful feedback :)

Day2: Attendance: Name, email, website/linkedin

1. Adam Taranto, adam.taranto@unimelb.edu.au
2 Jeremy Pike, j.a.pike@bham.ac.uk
3Thomas Kiley, hi@tkiley.co.uk talks.tkiley.co.uk
4 Randa Higazy, randa.higazy@uhn.ca
5Louise Grimble, louise.grimble@newcastle.ac.uk
6 Michael Milton, milton.m@wehi.edu.au
7Caitlin Bourke bourke.c@wehi.edu.au
8Margarita Novikova
9Alex Southgate, southgateJA@cardiff.ac.uk
10Martin Dreyer, martin.dreyer@nwu.ac.za
11 Isabela Paredes Cisneros, isabela.paredes@embl.de, https://www.linkedin.com/in/isabela-paredes-cisneros/
12 Jack Atkinson, jwa34@cam.ac.uk, https://jackatkinson.net/

Feedback on Day 1:

Something we need to continue with:
Good mix of discussion in breakout rooms and learning via the slides
Change of pace between exercises keeps the lesson engaging
Creating a welcoming environment to discuss and improve learning
Breakout room exercises
Giving practical advice rather than theory
Practical exercises and advice
I really like how often we get to practice and work on the concepts we learn. I also like the pace
Periodic discussion with others to learn from one-another in an informal, low-pressure way.
Breakout rooms have been useful
Breakout rooms also good to introduce ourselves given we are from all over the world!

Something we need to do better:
Keeping the slides in sync with the current topic+1
There are lots of good examples of how to implement the techniques being mentioned, however my cognitive load is maxed out trying to rember them all. I would be able to focus better on the content if I knew there was a handbook/runsheet for workshops that had these things written down. That way would only need to map the idea to a location in the notes. As a specific example - a collection of feedback prompts that instructors have found effective. https://docs.carpentries.org/ - Caprentries Handbook https://carpentries.github.io/instructor-training/instructor/index.html - Instructor Training curriculum

If notes and prompts were aranged around the flow of an actual workshop.
Concepts are sometimes a bit abstract as to how to apply to the different scenarios we may find ourselves in when teaching - i.e. more examples of teaching concepts in data/software etc. (vs car analogy for example!)
Perhaps a little too much technical termonology and lingo which can be a bit abstract/dry, more focus on practical examples would be nice
Sometimes too much info, either per slide or between exercises and I feel like I've lost the thread

There could be more activity during the 'lectures' to keep engagement. I often force myself to take some sort of notes, especially in virtual seminars to stop my attention wandering.
It would be interesting for the materials covered during the course to have examples related to more technical concepts.+1
Could be good to review concepts from yesterday and link to what we are doing today
A lot of info + text in supplementary docs

Actvity 14:
Taught someone about `git commit -u` instead of `git commit .` meaning their git repo wasn't full of junk. Easy to learn and useful.

Used Quarto to generate an HTML table from a simple data frame. This demonstrates how the abstract R data structures can be converted into real world documents which can be used in reports and papers.

Write a function that does a task you might do manually - save time, reduce possible error

Activity 15:

Motivational:
- Feeling the content and skills are achievable and applicable
- Fun presentation style
-show how what you are learning can be applied
- if the teacher/lecturer was super prepared
- when the instructor is passionated about the material

Demotivational:
- Noninclusive environment
- Slides full of text (couple with same or different content mentioned verbally)
- When the teacher assumes you fully understand prior knowledge
- Not feeling relaxed / welcome

Activity 18- Core values
Empowering and valuing contributions - We create an atmosphere where anyone who finds a typo/something that could be clearer etc is encouraged to open a PR to improve the materials.

Activity

Day1: Attendance: Name, email, website/linkedin

1. Martin Dreyer, martin.dreyer@nwu.ac.za, @Amfdrey
2 Jeremy Pike, j.a.pike@bham.ac.uk
3 Michael Milton, milton.m@wehi.edu.au, NA_character_
4 Jack Atkinson, jwa34@cam.ac.uk, https://fosstodon.org/@jatkinson1000
5Louise Grimble, louise.grimble@newcastle.ac.uk

6Thomas Kiley, hi@tkiley.co.uk
7Alex Southgate, southgateJA@cardiff.ac.uk
8Caitlin Bourke bourke.c@wehi.edu.au
9 Isabela Paredes Cisneros, isabela.paredes@embl.de, IsaWalls
10 Adam Taranto, adam.taranto@unimelb.edu.au
11 Toby Hodges, tobyhodges@carpentries.org
12 Angelique Trusler, angelique@carpentries.org
13
14
15

Introductions:

Please fill in the questionnaire: https://forms.gle/23PhSZq8nc7sGzEb7

Activity 2:

A.Why are you taking this course?

To improve my ability to teach and develop courses in the Carpentries style
To learn and apply current pedagogy around teaching coding to novices.
To become a better teacher and deliver an "image processing with python" Carpentries next year
To learn how to deliver information effectively to others and help others more effectively
To learn to teach more effectively and considerately, and to deliver a git training course in Feb.

To better support the trainers in my project, both from the management POV and getting involved as trainer/helper
To learn how to deliver training more effectively as part of work I am doing as an SSI fellow. To improve training materials I have already developed.
To develop a more structured way of teaching which will hopefully improve teaching and learning

To learn best practises with teaching software skills, and to further refine the intermediate software skills course I'm helping to develop.

B.What question/s would you like answered by the end of the course?

Why do the Carpentries workshops not use slides?, how interactive should a course ideally be, how do I avoid my workshop becoming too much like a lecture
How do you cater to the needs of individuals when teaching in a group setting?
How much scope is there is there to alter Carpentries material to taylor to the needs of your participants and your own teaching style?
How I can find out who would like these workshops in my local area, what structure of teaching works best in group enviroments (~~e.g. content of slides~~ etc Update: see later question)
How can I advertise and make more appealing a Carpentries-style workshop beyond Academia?
How to motivate people to sign up for training? How to help people build confidence?
What are some strategies to simultaneously teach people at different stages of their learning
How to accomodate a range of abilities in a class. Adapting and adding new material/courses.

Access to citations for evidence based teaching methods used by the carpentries - so I can convince admins that carpentries is worth investing in.

How do the course notes connect to the workshops - e.g. should people be following along, use them as a reference after the fact etc

Mental Models

As a person learns a topic, they build up a mental model of that domain.
Different people have different mental models, and nobody's model is complete.
It is important to choose analogies that are appropriate to the mental model that your learners have of the topic, i.e. analogies with too much complexity will not be helpful to novices, because the analogy will not fit to the mental model they currently have.

Analogy Brainstorm:

1. Think of an analogy to explore. Perhaps you have a favorite that relates to your area of professional interest, or a hobby. If you prefer to work with an example, consider this common analogy from education: “teaching is like gardening.”
2. Share your analogy with a partner or group. (If you have not yet done so, be sure to take a moment to introduce yourself, first!) What does your analogy convey about the topic? How is it useful? In what ways is it wrong?

Mental models and expertise:
We describe three broad categories of expertise: novice, competent practitioner, and expert. The novice has only a very rudimentary mental model, and does not know what they do not know. The competent practitioner has a mental model that allows them to carry out most common tasks/achieve a set of things that allow them to operate on a day to day basis, but still needs help/has to look things up often, e.g. when trying to do something new for the first time. Experts have a complex mental model that allows them to quickly handle even new situations by extrapolating from the knowledge they already have.
The difference between these levels of expertise is about complexity of mental model: in part that means that experts are aware of more concepts, but more than that it usually means they know a lot more connections between those concepts: their mental models contain more information about the relationships between concepts.

Misconceptions:
Simple factual errors can be fixed by demonstrating that misconception is incorrect and presenting correct information
Broken mental models can be harder to identify and correct because e.g. they do not always result in incorrect outcomes, but misconceptions in underlying model can eventually result in wrong outcomes and can slow down learning, because learners will try to fit new information into their existing (broken) model. They need help to identify and correct what is broken in their model.
Fundamental beliefs are difficult to change because they are deeply embedded, connected to their identity. We do not try to change these in a workshop.

Concept Mapping:

1.Draw a concept map of the topic you discussed just now (the analogy in Activity 4)
2.Identify 1 - 2 core concepts, and their relationships
3.Identify 1 (or more) types of misconception that can occur and classify them as factual error, broken model, or fundamental belief

I found it tricky to draw a map that wasn't very linear (like the car example).

Activity6:

Based on your previous educational experience (or even this training so far!) what types of formative assessments do you know about?
Write them in the Etherpad.
Ask question which tests understanding with two answers and ask for raised hands for each choice. If people dont raise hands for either this probably means they dont know whats going on or have lost attention.
Programming exercises with solutions
Essay
Lab reports
Exams/tests
Quiz
Multiple choice
Fill in the blanks
Using 'slido' to submit questions during a lecture/workshop, and using 'polls' during workshop to check understanding.
Polls during a lecture

Activity 7:

Suppose we are teaching children multi-digit addition. A well-designed MCQ might be:
Q: what is 27 + 15 ?
a) 42
b) 32
c) 312
d) 33
Choose one wrong answer and write in the Etherpad what the misconception is associated with that wrong answer

b) Adding in columns and the ~~user~~ student has forgotten to carry the 1 from (5+7=12)
c) 312: 2+1 =3 and 7+5 = 12, student has concatenated these two values.
d) 33 - This is thinking that 15 is 1 and 5 that are being added separately i.e. 27 + 1 + 5

Activity 8:

Formative assessments allow us as instructors to adapt our instruction to our audience. What should we do as instructors if the class chooses:

1. Mostly one of the wrong answers?

Directly address the misconception leading to this answer, and then reiterate the correct understanding.
Go through a similar question step-by-step as a group to ensure all students can approach the question
Address the misconception / misunderstanding that would have led to this answer, go through an example where it would be relevant and get to the right answer, give another exercise to check understanding
Identify what are the likely sources of misunderstanding so that it can be clarified to improve understanding

If it's not clear what the students are not understanding you might see if people are willing to explain how they reached the different answers and this could help identify where the gaps in knowledge are

2. Mostly the right answer?

Briefly mention the misconceptions that lead to the incorrect answers.

Talk about any misunderstandings brielfy that did come up, with the option of discussing them in more detail if anyone has any questions (either immediately or after)

Move onto the next topic to avoid slowing down the workshop. Maybe with a brief reminder about revisiting the slides if someone is stuck.

3. An even spread among options?
prodiving a revision session to help consolidate ideas and gain an understanding of misconceptions
Revisit the concept, address misconceptions
Repeat the explanation of the concept, since it seems it is generally unclear
Explain how one might have arrived at each of the answers, using them as an opportunity to explain how the main concept works.

Go through the questions step by step, identifying and correcting each misconception that could lead learners off the desired path

For one of the above, enter your answer in the Etherpad

Activity 9:

What is something that **you** are an expert in?

How does your experience when you are acting as an expert differ from when you are not an expert?

More confident teaching and explaining/answering questions on topics
Both depth and breadth of knowledge and experiences.
Knowing where to get the answers to things you don't know ('knowing what you don't know').
Using jargon / technical terms without realising that they aren't commonly known
Going off on tangents / wanting to share all the knowledge

Being able to identify a core exciting/inspirational hook into a new topic.
Able to problem solve faster
Can be a disadvantage as you have forgotten what it is like to not know everything about a topic. Difficult

to identify entry points.
Being able to solve the fundamental cause of complex problems

Questions with regards to this mornings session?

With regard to formative assements/ MCQ: if mostly the right answer, but not 100% the right answer, is it best to move on, or given the quote about better goin g too slow then too fast should actually do the same as if there are some wrong answers?

How does the concept map and identification of misconception influences the preparation of lessons as an instructor?

If during the lesson preparation I didn't identify a misconception I found while teaching the lesson, how do I face that situation?