General Coding Presentation Checklist

Variable Names

Quality variable names (highly descriptive/painfully obvious) should simplify your presentation significantly.

Slides

Consider using a light colored background with a dark colored font in a presentation even if you code in dark mode.
I once saw these fonts recommended for TEXT in any general presentations:

- Calibri
- Corbel
- Candara
- Consolas
- Constantia
- Cambria

Monospace fonts are recommended for CODE in presentations:

- Courier New
- Lucida Sans Typewriter

Did you acquire all the data requested of you in the manual or by the instructor?
Did you use the equation editor to make your own equations?
Did you create your own figures?
Did you ensure all text was at least 18-point font?

Compare text size in code snippets to 18-point font as well…

EXCEPTION: When citing another's work, the citation should be 14-point font on the slide.

Consider checking out this set of slides with some ideas for presenting code. I went through it in about 10 minutes.
https://sqadays.com/files/autoupload/10/47/49/v0agh1yc38171.pdf
If the link is busted, do some web searching for "coding presentation example" to get more ideas.

Images on slides (include images on Title & Goal/Question slides)

Most coders will initially make a bunch of slides with images of code fragments and text.
Don't forget to include code output on those same slides.

- If output is visual, include some screen shots to help explain the code.
  - Lines 131 thru 151 of following link should work as a pause button
  - https://glowscript.org/#/user/robjorstadahc/folder/ForceSims/program/Atwoods01
- Include arrows with labels in your visualizations.
  - Labels should employ formatting such as italics & subscripts
    - Line 66 of the code link above shows you how to do italics, subscripts, and superscripts
  - Including numbers is useful, but format them to reduce sig figs
    - Line 66 also shows you how to format the digits with an `f`-statement (thing in quotes after letter `f`)
    - Using `{your_number:.1f}` gives 1 decimal place
    - Using `{your_number:.2e}` gives engineering notation with two decimal places
- If output is graphical, include the graphs in your talk!
  - Standard formatting rules apply.
  - It is almost certainly easier to output the data to Excel so you can use your templates
    - Watch this vid: https://www.youtube.com/watch?v=oAbYQWcBj1w&list=PL4Sl1ZPMcTDX-cQkqRjsVD4HvaLVvJPE8&index=19
    - Incorporate an appropriate print statement into your loop (numbers separated by commas)
    - Copy the appropriate text from the print output of glowscript
    - Paste the text into Notepad (TextEdit might work for Macs?)
    - Save the file without any formatting
    - Open Excel and select open files (be sure you look for *all* file types, not just Excel Files)
    - Open your text file and use the *comma delimited* option

Running the code during the talk versus pre-recording gifs or vids?

Pre-recording videos ensures you get just the right viewing angles for your presentation.
Pre-recording vids allows you to get the display speed just right.
Pre-recording vids allows you to loop the vid and allows more time to explain various things the code is doing.
That said, be prepared to run the code with new parameters suggested by the audience in case a question arises…

Plotting

Showing code outputs match expectations visually for a single case is must.
Furthermore, any reasonable coding talk show how the code output varies as various inputs are changed.
Ideally, showing a plot of *something* usually shows your code is more robust.
Possible plot ideas:

- Percent difference of code output compared to theory *as a function of increment size*
- Code output as a function of one particular code input (i.e. time to fall as a function of drag constant chosen)

See general guidelines for info on plot formatting…

Euler-Cromer Method

If you have animated motion, chances are you used the Euler-Cromer Method.
It is sensible to include a slide on how the method was employed in your talk.

- Euler is pronounced "Oiler" (not Yooler).
- Make your own version of a slide similar to what I show below (do NOT copy mine…make your own)
- Include either the left column or the right column depending on how you coded things.
- In the slide, also include snippets of your actual code showing the implementation
- If possible, include an image on the slide to keep it from being boring.

$\vec{F}_{NET} \rightarrow \vec{a} \rightarrow \vec{v} \rightarrow \vec{r}$	$\vec{F}_{NET} \rightarrow \vec{p} \rightarrow \vec{r}$
• Compute $\vec{F}_{NET}$. • Use $\vec{a} = \frac{\vec{F}_{NET}}{m}$ (Newton's 2nd Law) • Use $\Delta\vec{v} = \vec{a}\Delta t$ ($\vec{a} = \frac{\Delta\vec{v}}{\Delta t}$) Equivalent to $\vec{v}_f = \vec{v}_i + \vec{a}t$ In code `v += a*dt` • Use $\Delta\vec{r} = \vec{v}\Delta t$ ($\vec{v} = \frac{\Delta\vec{r}}{\Delta t}$) Equivalent to $\vec{r}_f = \vec{r}_i + \vec{v}t$ In code `ball.pos += v*dt` • Increment time using `t += dt`	• Compute $\vec{F}_{NET}$. • Use $\Delta\vec{p} = \vec{F}\Delta t$ ($\vec{F} = \frac{\Delta\vec{p}}{\Delta t}$) This is a variation of Newton's 2nd Law Equivalent to $\vec{p}_f = \vec{p}_i + \frac{\vec{F}}{m}t$ In code `p += F/m * dt` • Use $\vec{v} = \frac{\vec{p}}{m}$ (def'n of momentum) • Use $\Delta\vec{r} = \vec{v}\Delta t$ ($\vec{v} = \frac{\Delta\vec{r}}{\Delta t}$) Equivalent to $\vec{r}_f = \vec{r}_i + \vec{v}t$ In code `ball.pos += v*dt` • Increment time using `t += dt`