Welcome to Codeup!

Thought for the day

> "The secret to building great products is not creating awesome features, it's to make your users awesome." - Kathy Sierra

Mindset

1. **Mistakes are fuel for learning**
2. Attitude is contageous
3. You learn more when you're having a good time. Have fun & enjoy the journey!
4. What you get out of this program is a function of what you put into it.

Expectations - know what's coming

- Expect to read and write a *tremendous* amount of code and analysis.
- Expect to *learn by doing*. Form hypotheses, try things, and adjust.
- Building new skills requires a little bit of study and **lots** of practice.
- Work together, search for answers, and ask for help.

Code of Conduct

- Be Cool. Act responsibly. Behave with integrity.
- No isms (racism, sexism, ageism, ablism, etc...) Only **optimism**
- Treat each other with excellence.
- Treat the program like your job. No BS. Bring your best.

How to Succeed as a Codeup student

1. Own your learning experience
    - Make sure you're **solid** on vocabulary, fundamentals, and first principles
    - Be honest with yourself and your comprehension. Be open with us. Ask for help.
    - We can explain things to you, but we can't understand it for you
2. Overcome learned helplessness
    - The **number one** learning objective: "Working with others effectively to find and solve problems in the face of uncertainty"
    - Expect to work with things before fully understanding all the ins and outs

    - The big take-away from Codeup is learned resourcefulness and increased automaticity.
3. Learn to problem solve (breaking down into small pieces)
4. Put in the reps (we can't lift the weights for you)

What you will learn beyond the curriculum

Not only how to practice data science, but *how to keep learning this material*, long after the class is done

How to form effective questions and find your own answers

How to debug and troubleshoot using the scientific method

How to get the most from your Codeup experience

Adopt the mindset that you're learning a craft and the bare minimum won't cut it.

Read ahead in the curriculum. Read more deeply into concepts. Do extra research.

When you finish exercises, play with the code and give yourself new challenges.

Use what you've learned to build stuff and do new analysis.

Find or make yourself passion projects that are motivating for you to build.

Standing Homework

- Read ahead, finish exercises, and push your work to GitHub every day.
- Work on your own projects. It's the best way to learn!
- Get a good night's rest.

Mac Tips / Shortcuts

cmd + space - Spotlight search to search for applications/documents

cmd + a - Select All

cmd + c - Copy

cmd + v - Paste

cmd + x - Cut

cmd + z - Undo

cmd + q - Close application

cmd + f - Search for a keyword

cmd + tab - Cycle between applications

cmd + ` - Cycle between windows of the same application

cmd + arrows - Move cursor to the end of a line/beginning of a line

3 fingers swiping up - View all windows

2 fingers + up/down - Scroll

The Exercise Completion / Problem Solving Process

1. Set an overall goal and break the goal down into smaller steps / problems. In the case of curriculum exercises, this is largely done for you and the goal is to simply "complete the lesson exercise steps." In larger projects, this step will become increasingly important.
2. Ensure you know, in plain English, what the exercise or problem is asking you to do.
3. Understand the problem needing to be solved and how to succinctly describe it. For example...
    - "There is a syntax error in my `prepare.py` file on line 33."
    - "I'm not connecting to the database and my credentials appear to be correct."
4. Research the problem and potential solutions further if you don't have clear context.
    - Reread exercise instructions (if applicable)
    - Refer to the curriculum (check the appendix too)
    - Check the official documentation for that language or library
    - Google search and search **StackOverflow**
5. Create a solution hypothesis and test it like a scientist. (try to always have an expectation for what output you will get)
    - "I postulate that if I run my code now having changed the names of my local variables, I will get an output of "test" and "loop ran" in the console.
    - Where possible, identify places where you can print out values to better determine where the problem is occuring and if a potential solution solves the problem.
6. Keep track of various possible solutions tested. Be prepared to articulate them.
    - "I tried possible solution a, b, and c and got x, y, z results but I need to get a q result."
7. Ask your peers and instructors for guidance.

How to Get Help

1. Try to state the problem to yourself in plain English. Can you explain the solution in English?
2. Form a hypothesis, test that hypothesis, observe the results, alter your approach.
3. If that doesn't work, ask for help from `#jemison-queries` or in channel
    - Communicate with technical clarity. "I have a syntax error on line 23. Can you help me identify what am I missing?" vs. "It's not working".
    - Use search engines to find similar code/problems and collaborate with your fellow students.
    - Ask your neighbor, then ask an instructor
4. Rinse, repeat
5. Big problems are always composed of multiple, simpler problems.

Curriculum Access

1. Create a GitHub account. Make a sensible username. Recommend firstname_lastname, last_first, first_middle_last, or other, similarly professional combinations.
2. Go to https://tools.codeup.com/register.
3. Double check your GitHub username & email address before submitting the form.
4. The Codeup Data Science curriculum lives at https://ds.codeup.com