Advanced - Shell

EECS 201 Spring 2026

Submission Instructions

Answer the bolded text that begins with a "**Q**". This assignment is an "online assignment" on Gradescope, where you fill out your answers directly.

Presentation/Checkoff

In addition to submitting this assignment, you will need to attend a lab session after the deadline to discuss your work with the course staff. Refer to the course schedule for the window for doing so for this assignment.

As part of the discussion, be prepared to do the following:

- Explain your approach for the assignment. What documentation did you read, and how did you find it? What did you try that didn't work, and what was your process for figuring out how to fix it?
- Summarize what you learned from the assignment.
- Discuss how you plan on implementing the tools in this assignment and what you learned in your future projects.
- Apply your approach for solving the problems in this assignment to a new problem that you have not seen before.

I suggest keeping these in mind while doing the assignment, and writing down your thoughts so that you can review them before the checkoff.

Your overall score for the assignment will be a combination of your submission score and the presentation/checkoff score. Refer to the syllabus for the details.

1 Customizing your prompt (10)

At the end of this week's content we talked a bit about customizing your shell. For this advanced assignment let's trick out your prompt! The prompt is going to be an ever-present thing as you work at the command line, and can be a handy way to communicate info to you. For example, a lot of default prompts will tell you what directory you are currently in: something very convenient since I notice a lot of people get lost when their prompt doesn't have this, and it takes away having to manually run `pwd`.

As mentioned in lecture, the `PS1` variable is what determines your primary shell prompt, the one you see the most. In Bash and Zsh this variable can contain special sequences that'll cause the shell to expand those special sequences into runtime information like the current time, the current directory, etc. without you having to change `PS1`.

However, sometimes you might want more advanced functionality than what is built into Bash and Zsh's special prompt characters. What if you want to display the exit status (`?`)if the last command failed? What if you wanted to display a countdown timer? What if you wanted your prompt to have the information of a Git repository?

Bash and Zsh both offer "hooks" for you to write scripts that runs right before a prompt is shown, allowing you to dynamically set `PS1` to have certain information. Bash utilizes a variable called `PROMPT_COMMAND` while Zsh has the `precmd` function you can define, as well as an entire framework for handling and setting prompts and their themes.

Try playing around with setting the `PS1` variable at the command line, and check out the special characters that get expanded to special information for your shell! If you want changes to persist, put them in the approriate configuration file (e.g. `~/.bashrc`, `~/.zshrc`). Try using the prompt hooks.

Now for the requirements of this assignment. You may use any shell you want: I only mentioned Bash and Zsh since I know them best. **Write a shell prompt** that meets the following requirements. I'm purposefully leaving a lot

of these underdefined so you can stretch your creative muscles and write something that you'd fine useful. **A shell script that does some random task is not a shell prompt in itself and will receive no credit.**

- **Utilizes colors (3).** Do some research on ANSI color escape codes. Your shell might even have special characters for colors: look into its documentation.
- **Takes advantage of the built-in special expanded characters (3)**, These include things like current time, date, hostname (the computer's "name"), current directory etc.
- **Reports some "dynamic" information not provided by the canned built-in special expansions. (4)** For example, my prompt displays the exit code of the last command if it failed, information about the current Git repo I am in, and whether or not my shell is in an SSH session. I don't know what you might find useful, but chances are there'll be something that you just wish your prompt would tell you up front. The complexity of what you implement here will determine how many points you gain. Here is a rough guideline:
  - Simple command expansions: 2 points
  - A bit of control flow: 3 points
  - Pretty much a full on script: 4+ points

  The more mastery over shell scripting you demonstrate, the more credit you will earn. You can get credit for this by leveraging certain tools: sometimes figuring out the tool is most of the work (e.g. taking advantage of Zsh's prompt framework).
- Have fun! This isn't a grading criteria, by the way. Make it look pretty with non-language characters like box-drawing characters, symbols, and emoji! It's your shell, after all!

Submit whatever files/code needed to implement your shell prompt (if you are submitting your `.bashrc` or `.zshrc`, **just submit the stuff you wrote and not the whole configuration**) and submit screenshots demonstrating your shell's dynamic component in action, and answer the following question:

Q: What sort of dynamic prompt enhancements did you make and why?