Imagine you are a software developer working on a development team about to start a new project. Write a coherent, well-informed argument for why your team should use a particular programming language.

You have much freedom in this assignment:

- You can choose the nature of the company you work for. Examples: a movie studio, a games studio, a university, a research lab, a software consultancy, etc.

- You can choose the nature of the project your team is about to start. Examples: a new interactive website, a new mobile app, a new data processor, a new desktop program, a new component (such as a rendering or physics engine) for some larger software, etc.

- You can choose the nature of the team. Examples: three expert programmers who have worked together for a long time, ten programmers ranging in ability who have recently formed a team, a group distributed around the world in remote locations, etc. You may also choose the level of expertise your team has in various languages.

- You can choose what language to advocate for. It need not be a language discussed in this course.

- You can choose whether you are advocating against some other language (perhaps the language that other team members have used before) or not.

- You can choose whether you are writing to convince the team leader or all the team members.

Despite this freedom, you must make these choices. After all, it is impossible to make a well-informed argument for or against a language without a backdrop of specifics. Take time before writing to make concrete choices about the above. Envision your company, your project, and your team. The more specific this all is, the better your writing will be.

There are several requirements of this assignment:

1. You must include several (at least 3) multi-line samples of code illustrating your arguments.

2. You must make clear in your writing your choices for the above freedoms. This can be done either by having a little preface that explains the situation to me (this preface would be addressed to me as your course instructor; it’s not part of the hypothetical situation in the assignment), or it can be done by working this information into your argument proper.

3. You must use the technical terms we have learned this semester in making your argument, and you must use them accurately. For example, claim that Python is untyped and interpreted, not just that it is, say, forgiving of mistakes and easy to run. (You can talk about the fact that Python is easy to run—but explain that it is easy to run because it is interpreted and need not use a separate compilation step.)
4. You must use fonts and formatting to make your argument clearer. Specifically, code snippets should be obviously distinct from other text. I do this in my writing by using this font, but you can take some other formatting choice. (I configured Microsoft Word to change fonts back and forth via macros assigned to hotkeys to make my life easier; you may want to do this, too.) Multi-line code snippets should be visually distinct from paragraphs of English. If you have code snippets in multiple languages (say, because you arguing against some language as well as for some language), make sure the reader can tell which language each snippet is in. Color might be an effective way to do this.

5. You must use proper, grammatical English to make your argument shine.

I expect submissions to be 2-5 pages long, but this is not a requirement on either end.

Assessment criteria:
• Proper use of English; you are encouraged to ask for feedback from peers to correct any grammatical mistakes.
• Proper use of programming-language terms: this includes correct use of terms that you have written and choosing to use technical terms where they are available (instead of using a non-technical description).
• A tight relationship between your examples and your argument.
• Correct code in your examples.
• Good use of formatting.
• Clarity of argument.

Note that you will not be assessed on whether I agree with your stance! Though I’m certainly opinionated about programming languages, I fully believe there are environments where Python and JavaScript (to name two languages I sometimes disparage) are absolutely the right choice for a project. Even if your setting is not one of these environments, I recognize that reasonable people differ in their informed opinions. What I am after here is to see an informed opinion, cogently presented.

Tips:
• Be very concrete about your hypothetical situation. Give a name to your company. Give a location to it. Give names and identities to the other people in your team—even if you don’t use these names in your write-up. Think about who will use the program your team is tasked with writing. Imagine your history with the company—are you a new hire? have you been there for 20 years?—and your relationship with others. All of these concrete decisions will give texture to your writing, even if you don’t include the details explicitly in what you write. The more fleshed out your hypothetical situation is, the better your writing will be.
• Debate your ideas with your peers before and as you write. This may discover holes in your argument.
• Make sure that your code snippets directly relate to your argument, and aren’t just decoration.

When you are done, you will submit your work via Gradescope as a PDF upload. Do not submit in any format other than PDF, please.