CS330: Programming Language Project (PLP)
Assignment 3: Data types and naming conventions

Understanding how different types of data are represented in your programming language is a key step in being able to use it for bigger projects. Research the naming conventions in your language for variables (i.e., do they have to start with lowercase letters? Can they start with numbers? Symbols? do programmers use underscores, as in "last_name", or do they use camel case (lastName))? Write a piece of code that creates variable of each of these common data types and follows the naming conventions:

- int
- string
- floating-point number
- boolean
- array/list
- hash/dictionary

Even if your programming language only has one data type, or if it doesn't require that types be declared, you should still be able to create variables that store these types of information (well, maybe not the hash table). If your language doesn't have variables or doesn't differentiate between data types, then find out how it stores information and do that.

In your code, experiment with doing different things with the data types: can you add ints and floats? If you do, is the resulting variable an int (narrowing conversion) or a float (widening conversion)? Can you put different data types in the same array or list? Can one data type be converted to another (int to float, string to int, etc)?

Discussion questions:

1. What are the naming requirements for variables in your language?
    1. What about naming conventions? Are they enforced by the compiler/interpreter, or are they just standards in the community?
2. Is your language statically or dynamically typed?
3. Strongly typed or weakly typed?
4. If you put this line (or something similar) in a program and try to print x, what does it do? If it doesn't compile, why? Is there something you can do to make it compile?
    ```
    x = "5" + 6
    ```
5. Describe the limitations (or lack thereof) of your programming language as they relate to the coding portion of the assignment (adding ints and floats, storing different types in lists, etc). Are there other restrictions or pitfalls that the documentation mentions that you need to be aware of?
6. How do type conversions work in your language? Are the conversions narrowing or widening, and do they work by default or do they have to be declared by the programmer?

Make sure that your answers are clear, accurate, and fully-formed: remember that these tutorials are public, and GitHub users don't have the context of the assignment that you do. Explain the reasoning behind the answers as much as possible. If there is no clear-cut answer to a question, explain why not. And cite your sources! You can incorporate code into your tutorial to show examples, but you should also have a file in your repository that is just code, which someone could download and run.