CSc 372 — Comparative Programming Languages
Spring 2026 (McCann)

http://www2.cs.arizona.edu/classes/cs372/spring26/

Homework #2

(80 points)

Due Date: February 25th, 2026, at the beginning of class

Directions

1. **This is an INDIVIDUAL assignment; do your own work! Submitting answers created by other people or generated by AIs (e.g., ChatGPT) are a few examples of NOT doing your own work.**
2. Write complete answers to each of the following questions, in accordance with the given directions. Create your solutions as a PDF document such that each question is on a separate page; all parts of a multi–part question may be on the same page. Show your work, when appropriate, for possible partial credit.
3. If you have questions about any aspect of this assignment, help is available from the class staff via piazza.com and our office hours.
4. When your answers are ready to be turned in, do so on gradescope.com. Be sure to assign pages to problems after you upload your PDF. Need help? Visit https://help.gradescope.com/ and search for "Submitting an Assignment."
5. Remember that you can use at most one late day on a homework assignment, because we will be distributing solutions after that time.
6. Start early! Getting help is much easier n days before the due date/time than it will be n hours before.

Louden/Lambert Chapter 2: Language Design Criteria

1. (5 points) We've covered the concepts of generality, orthogonality, and uniformity as they relate to programming languages. Using your choice of Python or Ruby, provide one example of each of those concepts in that single language.
2. (5 points) Considering any of Java, Python, or Ruby, what is a language feature that you wish the language maintainers would add to that language, and, if it were added, how would it need to behave/interact with existing language features?
3. (5 points) Some languages require that variables be declared before they are used, some don't, and some have declaration as an option. Imagine that you're on a debate team, and your task is to defend the position that all languages should be required to declare their variables. Provide four arguments in favor of variable declarations, one from each of these four areas: efficiency, readability, security, and writability.

Louden/Lambert Chapter 5: Object–Oriented Programming

4. (5 points) Java is not a 'pure' object–oriented language, in the sense that it has both primitive types (`int`, `float`, etc.) and reference types (objects). Give one reason in favor of this design decision, and one reason against it.
5. (5 points) In Ruby, the Hash class accepts the "each" method, allowing hashes to be interacted over, like a collection. In Java, however, the Map classes are not formally part of the JCF (Java Collections Framework). For both Ruby and Java, provide (and justify) an advantage of the language's choice of location in the class hierarchy of its form of associative list.

(Continued . . .)

6. (5 points) `Object` is at the top of Java's class hierarchy, but we don't write Java programs with variables declared to be of type `Object`. Why, then, does Java define methods like `equals` and `toString` in `Object`?

7. (5 points) What are the basic differences, if any, in how Java and Ruby handle information hiding (a.k.a. encapsulation)?

8. (5 points) We are familiar with generics (parametric polymorphism). Both Java and C++ provide it. Ruby does not. Is this a significant problem for Ruby programmers? Justify your answer.

9. (5 points) Imagine that class `S` is a subclass of (that is, inherits from) class `C`, and that `t` and `d` are objects of classes `S` and `C`, respectively. Further, assume that `m()` is a method of `S`, but is not a method of `C`. After the assignment `d = t` is (legally!) performed, why is the call `d.m()` illegal?

Louden/Lambert Chapter 6: Syntax

10. (5 points) Many languages (e.g., C and Java) distinguish the character 'c' from the string "c" with different pairs of quotation marks. Others (e.g., Python) use "c" for both single characters and strings of length one. Provide (and justify) one advantage and one disadvantage of Python's approach.

11. (10 points) Add subtraction and division to the EBNF grammar provided in Figure 6.18 on page 222 of our text. Make sure the operators possess the expected precedences.

12. (10 points) Figure 6.17 (p. 220) in our textbook is also a small grammar. For each of the expressions below, draw both the parse tree and the abstract syntax tree.

    (a) `Z + Y * X + W * V`

    (b) `(Z + Y) * (X + W) * V`

13. (10 points) Left–recursion in the grammar of Figure 6.17.

    (a) It may appear that we can remove some left–recursion by replacing $expr \rightarrow expr + term \mid term$ with $expr \rightarrow term + term \mid term$. Why is this replacement not an option?

    (b) The grammar also uses left–recursion to define numbers. This could also be done with a right–recursive rule (specifically, $number \rightarrow digit\ number \mid digit$). In this situation, is the right–recursive version better, worse, or effectively the same as the left–recursive version? Justify your answer.