Programming Languages - Lab 1

Provide a short answer under each question. Your audience is the other students in this class; that is, you should provide answers that your classmates would understand. This is an individual assignment, but you may use the notes taken during the small group work completed in class. It is expected that you will read beyond the Tucker-Noonan chapter to complete this lab. This could be material you find online or from a programming languages text. Provide a bibliography of your references using the ACM format (see the [style guide here](https://www.acm.org/publications/acm-latex-style-guide-3jan2017)). You will be evaluated on completeness, clarity and quality of your writing, and use of references. Save your document as a **PDF**. Submit your PDF with the title "lab01-yourlastname.pdf" to Moodle. You can use the notes that you made during discussion.

Name: firstName lastName (#yourTeamNumber, e.g. R2-lisp)

1. Describe some ways in which programming languages differ from natural languages. Who is the audience for a programming language? Is it the same as the audience for a program?
2. Explain static versus dynamic memory.
3. Give some different kinds of examples of abstraction in programming languages.
4. What is meant by *application domain*? A *language paradigm*? How are the two related?
5. Why is providing language features to deal with exceptions important?
6. Look at the criteria for language design on p. 15 and the explanations that follow them. Describe orthogonality in your own words and provide some examples (or counterexamples). Pick two additional design criteria to describe and provide a few examples. You *must* go beyond what is provided in the Tucker-Noonan chapter.
7. What is meant by *nonconformant* when a standard discusses a language feature supported by a particular compiler? Give an example.
8. Look at the following [large list of programming languages](https://en.wikipedia.org/wiki/List_of_programming_languages). For each of the following languages, use the provided link or other resource to learn something about it. Write, in your own words, a brief summary of its features, paradigms, and relationship with languages that preceded or followed it.
   * Java
   * Scheme
   * Another programming language of your choice