**Language Design Proposal Template**

**Student Name(s):** Your name(s)

**Language Name:** The name of your language

**Compiler Implementation Language and Reasoning:** The language you're planning to write the compiler in, and why. It's completely acceptable to pick a language because you're familiar with it. If you're not familiar with the implementation language already, say so.

**Target Language:** The output language for your compiler. Can be anything (assembly, LLVM bitcode, JVM bytecode, CLR bytecode, JavaScript, etc.). See the project information page for details.

**Language Description:** a description of the language, from a high level. Why this language? What can this language do?

**Planned Restrictions:** is there anything that would make this language impractical to actually use? It's expected that you'll have something here. We don't have enough time to make a fully-fledged compiler and language, only enough time to get a complete start on it.

**Syntax:** EBNF definition of the syntax of your language. This can be either concrete, abstract, or some combination thereof. It's ok if this changes later. You should annotate non-obvious parts with what they mean.

**Computation Abstraction Non-Trivial Feature:** A non-trivial feature related to computation abstraction (see project information page for details).

**Non-Trivial Feature #2:** Another non-trivial feature. It may be a computation abstraction, but it's not required to be. It may also be a type-level feature. See the project information for details.

**Non-Trivial Feature #3:** Another non-trivial feature. It may be a computation abstraction, but it's not required to be. It may also be a type-level feature. See the project information for details.

**Work Planned for Custom Component:** What you plan to do for your custom component. Likely, you will want to implement one of your aforementioned non-trivial features for this. The work for the custom component isn't graded until the end of the course, so if there is any feature you think will take a long time, it's best to leave it for this component. If you're using a low-level target language, you do not need to plan any work for the custom component (you automatically get credit for it), though you can still list something here if you're planning something for the end.