Language name
Text Manipulation Language (TML)

Language motivation and overview
Many people in software development at one point or another come across situations where they need to process small or large quantities of text data. When the number of files to process reaches hundreds or sometimes even thousands, one must use an efficient tool to automate the task(s). I personally have used Perl and for individual file editing, Vim. Although I have enjoyed powerful functions offered by these tools, when it comes to regular expression, I have always had to educate myself before getting what I needed. For educational purposes of this course (for me to learn how to write a compiler) and with the goal of developing a more intuitive tool of choice for text manipulation, I have chosen to write a text manipulation language.

Problems it can solve
1. Simple string search/delete/replace
TML can find an instance or instances of a user-specified string in a text file. It can also delete lines containing the search string. String replacement is another simple text manipulation TML will offer.

Example:
```
Action = SEARCH;
String = ".obj";
InFilename = "build_log.txt";
OutFilename = "search_results.txt";
OpenFile (InFilename);
OpenFile (OutFilename);

Do {
  foreach line containing String in InFilename {
    result = performanAction on line
     print result OutFilename
  }
}
CloseFile(InFilename);
CloseFile(OutFilename);

Action = DELETE;
String = "debug";
InFilename = "search_results.txt";
OutFilename = "final_search_results.txt";
OpenFile (InFilename);
OutFile(OutFilename);

Do {
  foreach line containing String in InFilename {
```

```
    result = performAction on line;
    print result OutFilename;
  }
}
CloseFile(InFilename);
CloseFile (OutFilename);
```

2. Regular expression made easy

TML will offer pattern searching ability. One main goal I will try to accomplish with this project is to make pattern searching more intuitive and perhaps even more powerful than Perl. Sometimes people have an idea of what they want to search for, but they find it hard to express in terms of regular expression or a particular (scripting) language. My goal on this feature is to give the user a more expressive language than regular expression. The following example gives a very rough idea of where I am headed with this.

Example:

```
Action = PATTERN_SEARCH;
String = file path ending with ".obj";   // equivalent to .*.obj
InFilename = "build_log.txt";
OutFilename = "search_results.txt";
OpenFile (InFilename);
OpenFile (OutFilename);

Do {
  foreach line containing String in InFilename {
    print line OutFilename
  }
}
CloseFile(InFilename);
CloseFile(OutFilename);
```

3. Support for Unicode characters

This is an optional feature that may or may not be implemented this semester. I would very much like to add it to my language as I believe it will be very useful.