Vahe Karamian
Homework # 1 - CS 440 summer 2001
Due Date: Thursday, June 28

1) Answer the following questions briefly. (a) What is a compiler? What are the input and output of a compiler? (b) What are the main phases of a compiler? (c) What does lexical analyzer do? What are the input and output of a lexical analyzer? (d) What does syntax analyzer do? What are the input and output of a syntax analyzer?

(a) A compiler is software, and the input of a compiler is a source program written in a source language, typically a high-level language, the output of the compiler is the error messages, as well as the target program, written in a target language typically low level.

(b) The main phases of a compiler are the lexical analyzer, the syntax analyzer, the semantic analyzer, the intermediate code generator, the code optimizer and the code generator.

(c) A lexical analyzer chops the stream (strings) of input characters into meaningful, simple, static items, called “tokens” and creates a symbol table. The input and out of a lexical analyzer are character stream for the input, and token stream for the output.

(d) The syntax analyzer builds a parse tree from a stream of tokens. The process is called parsing. The input to the syntax analyzer is the token stream and the output is the parse tree with the leaf nodes containing the tokens.

2) Although compilers are designed to translate a particular language, they often allow calls to subprograms coded in some other language (typically, FORTRAN, C, or assembler). Why are such "foreign calls" allowed? In what ways do they complicate compilation? Describe briefly your suggestions to handle such situation in compilation.

This is allowed because maybe the particular language you are using does not support a specific feature and therefore we can use a foreign language which does. This is like writing assembly code within your C code, and people do this for more efficient tasks such as graphics. They complicate the compilation because now you have to support two or more languages instead of just one. I would say to just support a subset of the foreign language for particular functionality that the original language does not support.

3) The model of compilation we have introduced is essentially batch-oriented, that is, it assumes that the program will be fully compiled before the programmer can execute the program, or make any changes. An interesting alternative is an "interactive compiler." An interactive compiler, usually part of a program development environment, allows a programmer to interactively respond to source errors, fixing them as they are detected. It also allows a program to be tested before it is fully written, providing for stepwise implementation and testing. Redesign the compiler structure of Figure 1.9 on page 10 of the textbook to allow incremental compilation. The key idea is to allow individual program structures to be changed and compiled without necessarily recompiling everything. (You may Xerox Figure 1.9 and make changes on this figure. Also briefly document why you made such changes.)