Introduction to Compilers

A.G. - Material from The Essence of Compilers by Robin Hunter (Prentice Hall)
A compiler that is written in C and translates Java into Bytecode (the language interpreted by the Java Virtual Machine).
Building compilers

• How do we write the first compiler?
• How do we port one compiler from a machine to another?
Building compilers

• How do we write the first compiler?

Example of a compiler having compilers both as its input and its output.

A more sophisticated compiler for the full language can be built on top of a simple compiler for a subset of the language.

This can be repeated as necessary.
Building compilers

• How do we port one compiler from one machine to another?

A UIL would considerably simplify the problem, but defining it proved to be elusive. We can try though ...
Structure of a compiler

A well-written compiler is highly modular in design, and should present a good example of a well-structured program. Logically, the compilation process is divided into stages, which are in turn divided into phases. Physically, the compiler is divided into passes. We will describe these terms in more detail.

As we have seen, the principal (and often the only) stages that are represented in a compiler are analysis, in which the source code is analysed to determine its structure and meaning, and synthesis in which the object code is built or synthesised. In addition, however, there may be a pre-processor stage in which source files are included, macros expanded and so on. This stage is usually fairly straightforward and is mainly relevant to the languages C and C++. We will not consider it in detail.

Figure 1.6 shows the typical phases of the compilation process.

Analysis

Lexical analysis → Syntax analysis → Semantic analysis

Synthesis

Machine independent code generation → Optimisation of machine independent code → Storage allocation → Machine code generation → Optimisation of machine code
Lexical analysis

Lexical analysis

Lexical analysis generator

Lexical analyser

Lexical structure

Character sequence

Symbol sequence
Syntax analysis

A syntax analyser generator takes as its input the syntactical definition of a language, and produces as output a syntax analyser (a program in C for example) for the language. This is illustrated in Figure 1.9.

Parser generators have been developed which support most popular parsing methods, but probably the most widely known is the Unix-based YACC, which is Character sequence...

Lexical analysis

Lexical generator

Lexical structure

Symbol sequence

Figure 1.9

Symbol sequence

Parser

Parser generator

Parse tree

Figure 1.10