compiler design

Lecture 1

Computer Science
Rensselaer Polytechnic

66.648 Lecture 1 (01/12/98)

- Overview of Compilers
- Introduction to Lexical Analysis
- Course Administration

Overview of Compiler

- Compiler is a program (written in a high-level language) that converts / translates / compiles source program written in a high level language into an equivalent machine code.

source program $\xrightarrow{\text{compiler}}$ machine code

Example source language: Java
Example target language: Bytecode

Sample Program

```
public class first {
public static void main(String argsv[])

        int  x;
        x = 19;
        x = x*x;
}
}
```

Output Bytecode

```
Compiled  from first.java
public class first extends java.lang.Object {
        public static void main(java.lanag.String[]);
        public first();
Method void main(java.lang.String[])
0       bipush 19
2       istore_1
3       iload_1
4       iload_1
5       imul
6       istore_1
7       return
```

Byte Code Continued

```
Method first()
0       aload_0
1       invokenovirtual #3 <Method java.lang.Object.<init>()V>
4       return
```

Comments: There are two methods:
main method
constructor method.

Byte Code Continued

Bytecode instructions are 1,2 or 3 bytes long.
Bytecodes are executed in a postfix manner.
In the main method, one can see how x=x*x
is assembled.

```
iload_1
iload_1
imul
istore_1
```

Output Code (optimized)

Optimized Bytecode for Main Method will be

```
0       return
```

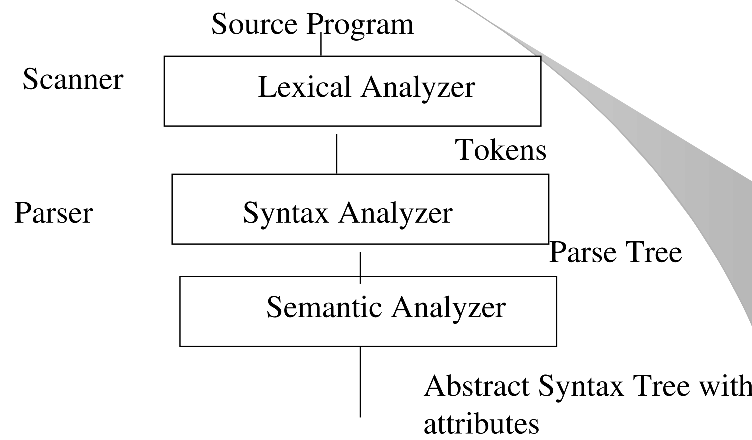
This is so because main method does not use the variable x in any "meaningful" manner.

Implementation

Compilers are written in a high level language.

Sometimes a compiler is written in the same language for which one is writing a compiler. This is done through Bootstrapping.

Phases of the compiler



Phases of Compiler continued

- Intermediate-Code Generator (produces Intermediate Code)
- Intermediate-Code Optimizer(produces Optimized Intermediate Code)
- Target-code Generator (produces target machine code)

One of the primary data-structures that a compiler uses is a Symbol Table. This data-structure is used by all of the phases.

Sample Program Compiled

Scanner takes an input program and breaks them into a series of tokens.

Tokens are entities defined by the compiler writer which are of interest.

Examples of Tokens:

Single Character operator: = + - * > <
More than one character operator: ++, --,==,<=
Key Words: public class static void method if while

Example Program Compiled- Continued

Identifiers: x argsv sample my_name Your_Name

Numeric Constants: 1997 45.89 19.9e+7

String Constants: “Rennselaer” “RSV's course”

Scanner’s task is to partition the sequence of characters into a sequence of tokens.
The tokens will be public , class, first,{, public, static, void, main,(,String,argsv,[,],),{,int,x,;,x,=,19,;,x,=,x,*,x,;,},}

Example Continued

The scanner reports errors if it encounters an invalid character. Often a token number is returned and the identifiers get stored in a symbol table.

The parser produces a parse tree:

```
                     root_node
            stmt1                      stmt2
stmt1                                   stmt2
   =                                     =
x      19                          x          *
                                          x       x
```

Administration

- Compiler Project (3) - 75%
- Test - 25%
- Compiler project is a group effort. All group members get the same grade.
  Test has to be taken individually. No discussion is allowed.

The course URL is
http://www.cs.rpi.edu/~moorthy/Courses/compiler98

I am assuming that you are all proficient in C/C++.

Read Chapter 1 of the Text Book.