



Programming Language Basics

BCS 307 – Compiler Construction

Agenda

- Language Specifications
- Identifier Rules
- Scopes
- Functions

Language Specifications

- Unlike natural languages, programming languages are unambiguous and do not evolve.
- To design a compiler for your own language, you need to specify the language rules, features and other specifications.
- Language specifications includes, identifiers pattern, keywords, punctuations, constants pattern, syntax of language, semantics, etc.
- Once the specifications are complete, you can begin designing its compiler.
- The language that you are going to design compiler for should be simple.

Static/Dynamic Distinction

- If a language policy allows the compiler to decide an issue, the language is said to use a static policy (issue can be decided at compile time).
- A policy that allows decision making at runtime, is said to be a dynamic policy.
- One such issue is the scope of declarations.
- A language uses static scope if it is possible to determine the scope of a declaration by looking only at the program.
- With dynamic scope, as the program runs, the same use of variable *x* could refer to any of several different declarations of *x*.
- Most languages, including C and Java, use static scope.

Names, Identifiers, and Variables

- An identifier is a string of characters, that refers to an entity, such as an object, a class, or a type.
 - A language has its own pattern or rule to define valid identifiers.
- The name `x.y` might denote the field `y` of a class `x`.
 - Here, `x` and `y` are identifiers, while `x.y` is a name, but not an identifier.
- A variable refers to a particular location of the store (memory).

Static Scope and Block Structure

- In C, the scope of a declaration is determined implicitly by where the declaration appears in the program.
- Languages, such as C++, Java, and C#, also provide explicit control over scopes through the use of keywords like public, private, and protected.

Procedures, Functions, and Methods

- A function generally returns a value of some type (the "return type"),
- A procedure does not return any value.
 - C language only has functions, treat procedures as functions that have a special return type "void," to signify no return value.
- Object-oriented languages like Java and C++ use the term "methods."
- These can behave like either functions or procedures, but are associated with a particular class.

C++ Static Scopes

```
main() {  
    int a = 1;  
    int b = 1;  
    {  
        int b = 2;  
        {  
            int a = 3;  
            cout << a << b; B3  
        }  
        {  
            int b = 4;  
            cout << a << b; B4  
        }  
        cout << a << b;  
    }  
    cout << a << b;  
}
```

The diagram illustrates the static scopes of the provided C++ code. It shows four nested blocks, each represented by a rounded rectangle. The outermost block is labeled B_1 and contains the code for the `main()` function. Inside B_1 , there is a block labeled B_2 which contains two sub-blocks: B_3 and B_4 . Block B_3 contains the code for the first nested scope, and block B_4 contains the code for the second nested scope. The code for the first nested scope is shown inside B_3 , and the code for the second nested scope is shown inside B_4 . The code for the first nested scope is also shown inside B_2 , and the code for the second nested scope is also shown inside B_2 . The code for the first nested scope is also shown inside B_1 , and the code for the second nested scope is also shown inside B_1 . The code for the first nested scope is also shown inside B_2 , and the code for the second nested scope is also shown inside B_2 .

Your Task

- Write specifications for your own language
- Initial specifications should include:
 - Identifier rules (DFA)
 - Keywords
 - Operators
 - Punctutations
 - Constants

Upcoming

- Lexical Analysis Basics