Introduction

- Compilers and interpreters contain symbol tables: tables which store information about all the identifiers used in a program.
- Purpose:
  - To verify that identifiers are properly used
  - Compilers: to translate identifier references to references to structures in target language.
  - Interpreters: to find value

Requirements

- Store information about each identifier:
  - What it is: Name, data type, size, structure (primitive or compound)
  - How it fits in the program: scope
  - Where to get the value: binding or binding instructions
  - Other: additional information (for compound variables or functions)
- Support multiple uses of same name
- Support operations:
  - Add new identifier
  - Update existing identifier's information
  - Check usage of identifier
  - Delete identifier?

→Symbol tables are big tables of data, i.e. small databases

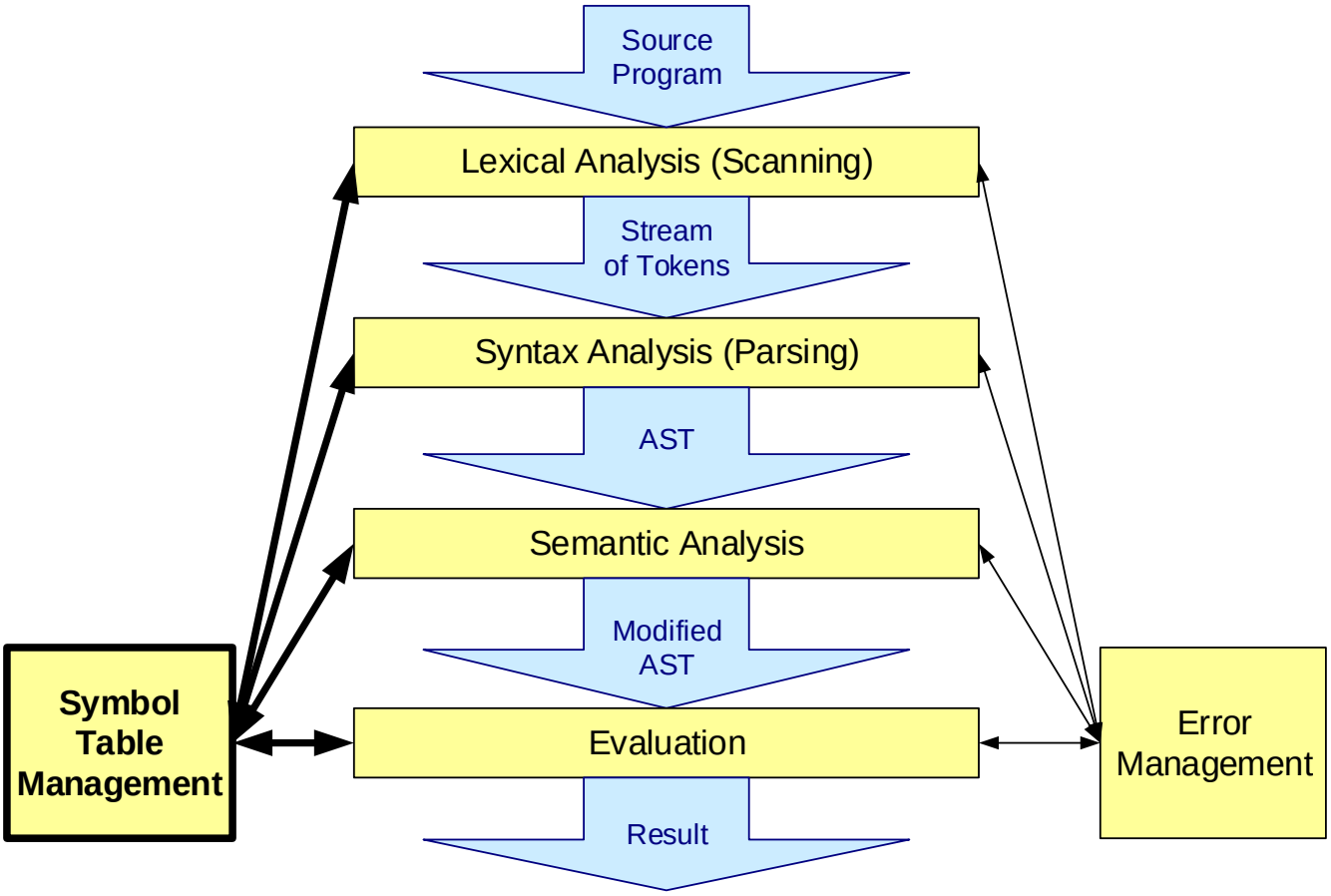
→Many possible data structures

Interaction of Symbol table with Translator Components

- Lifetime of symbol table:
  - Interpreters: whole session
  - Compilers:
    - Transient component used during compilation used to translate references to relative locations.
    - This component could be kept for debugging or profiling purposes.
    - Permanent component also stored with compiled code keeps information about publicly accessible identifiers to resolve external references.
    - For OOP languages, references to methods are often resolved at run time.

Interaction of Symbol Table with Interpreters

INTERPRETATION PHASES



Scanning

As it encounters identifiers, scanner creates a Global name table (= spelling table = identifier table = lexeme table) used to convert scanned identifier names into numeric references:

- Every new name is assigned a number
- Name-number association entered into table.

Later Stages

- Dynamic scoping: scope and type resolved and verified during evaluation
- Static scoping: parser builds a scope stack containing list of identifiers defined in each scope and uses it to associate each identifier encountered with a reference to its definition.