Compiler Design in C – Chapter 6.3

Code generation: Symbol Table

Marek Kotásek
xkotas02

Filip Kozák
xkozak12

The symbol table

- structure similar to database
- contains information about subroutines, variables, etc.
- indexed by a *key* field (e.g. variable's name)

The symbol table

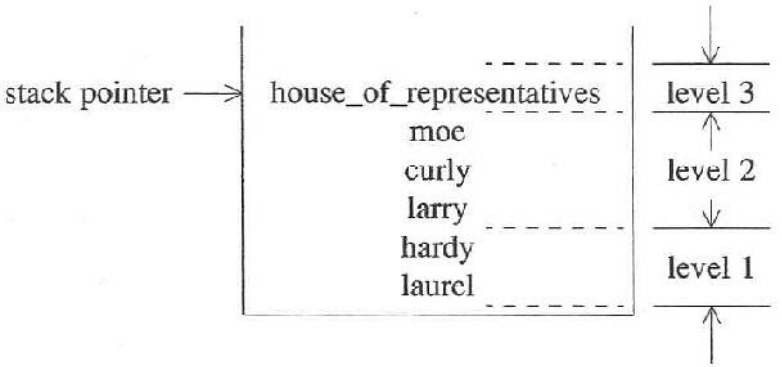
- can be used to communicate with the lexical analyzer (`typedef`)
- special needs:
  - speed
  - ease of maintenance
  - flexibility
  - duplicate entries support
  - quick deletion arbitrary elements

Two layers of the symbol table

- database layer
  - information needed for compilation
  - operations like inserting (finding, deleting) new entries in the table
- maintenance layer
  - used for managing the table at a higher level
  - creating systems of data structures
  - inserting structures into the table using low-level insert function

Data structures – STACK

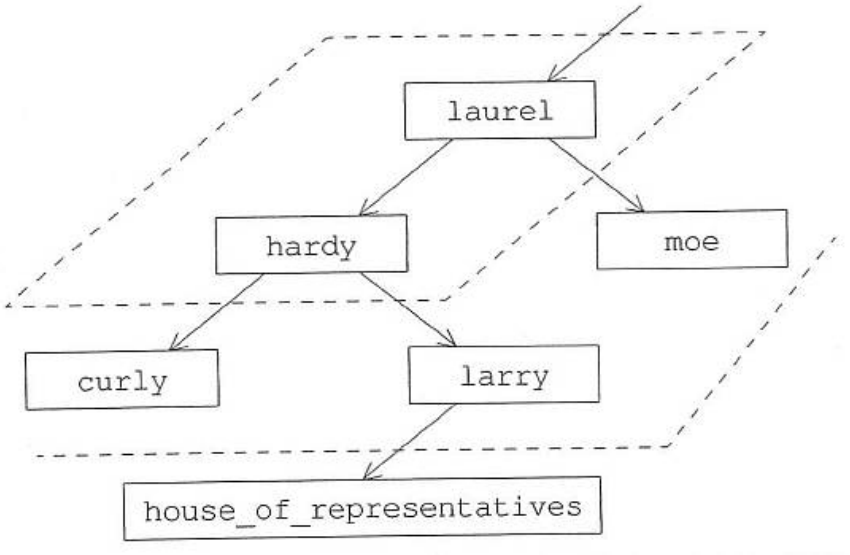
- linear array organized as a stack
- *push* and *pop* operations



- **advantage** of a stack
  - very easy to delete a block of declarations
- **disadvantages** of a stack
  - linear search required to find entries
  - maximum size must be known at compile time

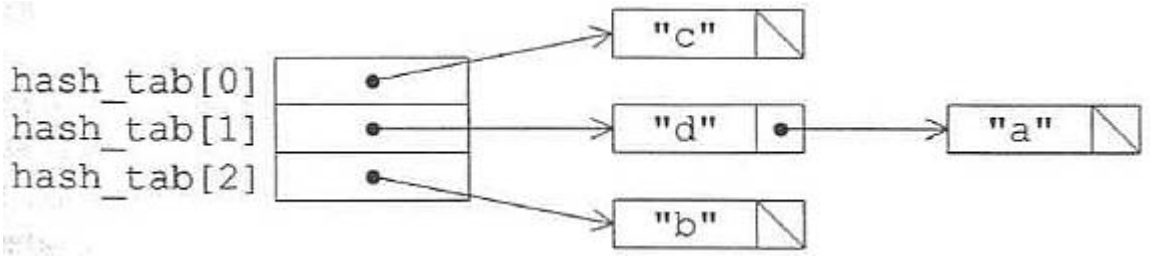
Data structures – TREE

- binary tree as the basic data structure
- logarithmic search time
- node deletion
- **disadvantages** of a tree
  - degradation to a linked list
  - collision situations



Data structures – HASH TABLE

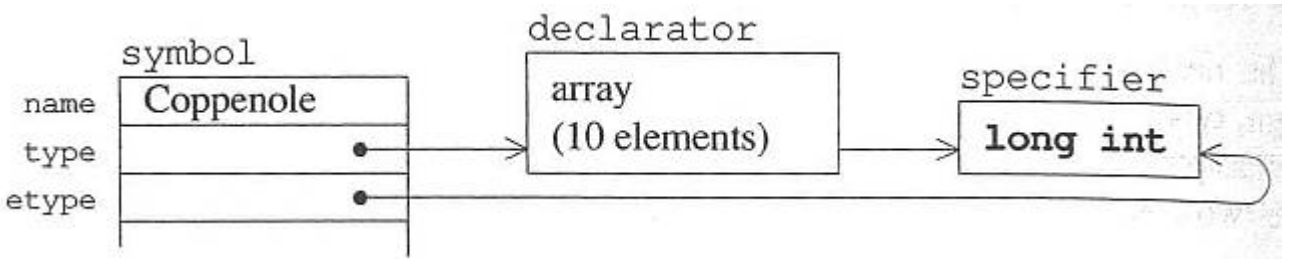
- the best data structure for the symbol table
- using compressed array and special function to compute the key's *hash value*
- resolving collisions by making a linked list



Representing types I.

- a **C** variable – represented by a system of data structures working in concert

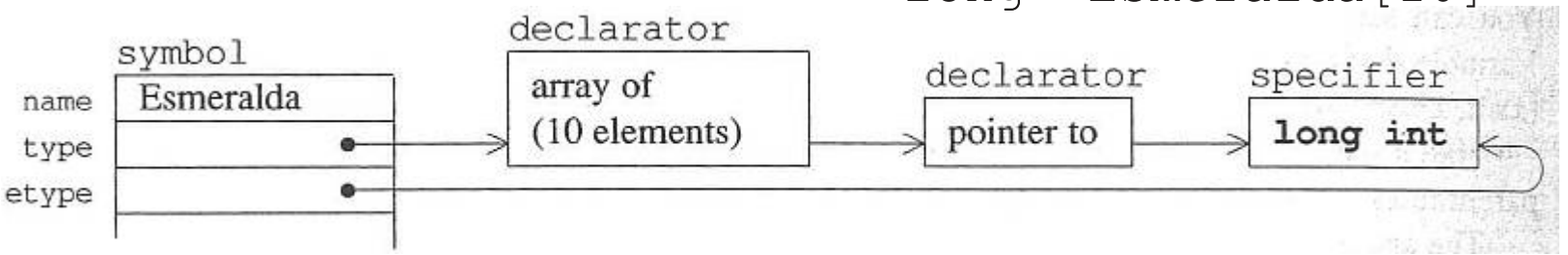
- variable declaration have 2 parts:
  - *specifier* part
  - *declarator* part



Representing types II.

- specifier part
  - list of various keywords – variable types (`int`, `long`, ...)
- declarator part
  - variable name
  - number of stars
  - array-size specifiers
  - parentheses

array of pointers to longs:
`long *Esmeralda[10]`



Compiler Design in C, chapter **6.3**

- complete implementation including C source code with explanation
- using hash table structure for the symbol table

Thank you for your attention.