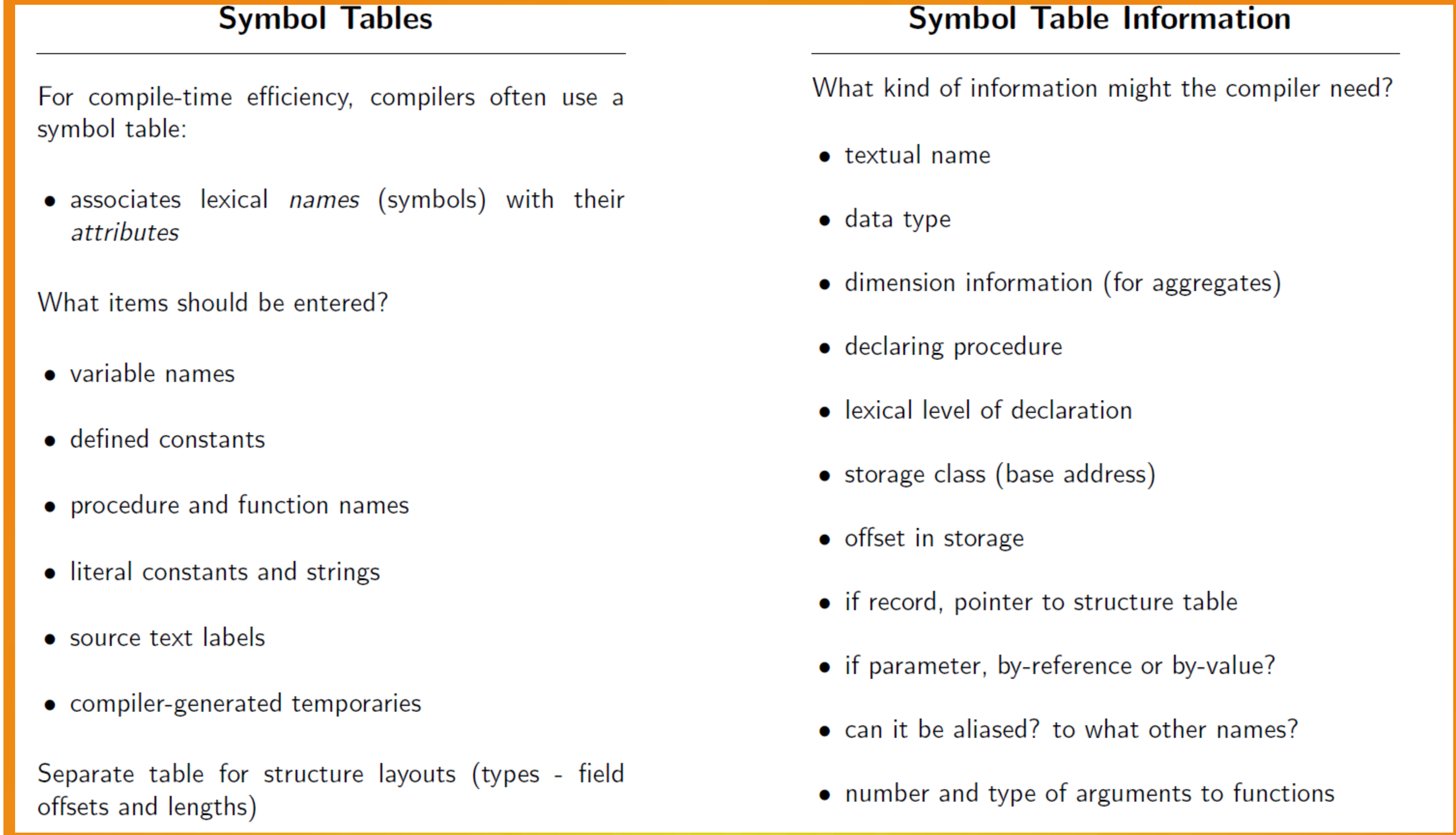
Symbol Tables



Symbol Tables

For compile-time efficiency, compilers often use a symbol table:

- associates lexical *names* (symbols) with their *attributes*

What items should be entered?

- variable names
- defined constants
- procedure and function names
- literal constants and strings
- source text labels
- compiler-generated temporaries

Separate table for structure layouts (types - field offsets and lengths)

Symbol Table Information

What kind of information might the compiler need?

- textual name
- data type
- dimension information (for aggregates)
- declaring procedure
- lexical level of declaration
- storage class (base address)
- offset in storage
- if record, pointer to structure table
- if parameter, by-reference or by-value?
- can it be aliased? to what other names?
- number and type of arguments to functions

Mixed Type Calculator Symbol Record

- Symbol Table Record including:
  - Name: FLOAT, INT, expr, etc.
  - Type: string of type name “float”, “int”, “double”
  - Value: union {

```
float fval;
int ival;
double dval;
} value;
```

Examples

- Table Structure
  - See structTest.c
- Symbol Table Implementation
  - See tableTest.c
- Need lookup function for Symbol Table
  - See lookupTest.c