BLOCK STRUCTURE AND SCOPE

Each block defines a unique scope

Delroy A. Brinkerhoff

INDENTATION

- Indentation helps the physical layout of code reflect its logical behavior
- Indented code "belongs to" or is nested inside a control statement
- Use indentation consistently
  - Don't mix styles
  - Don't mix indentation characters
- Indentation should make the code easier to read

BLOCKS

- A block is delimited by an opening and a closing brace: { and }
  - Creates block or compound statements
- Blocks can be created anywhere in a program but are usually associated with control statements, functions, etc.
  - C++ syntax allows replacing any statement with a block or compound statement
- A block creates a new scope

SCOPE

- Scope is the location in a program where a named item (often a variable) is visible and accessible.
- Three main scopes:
  - Local – inside a function
  - Class – inside a class
  - Global – throughout a program (generally avoided)
- Variables defined in one scope are not visible or accessible outside that scope
- Variable names must be unique within a scope

THE UNIQUENESS RULE

- Variable names must be unique within each scope
- Defining multiple variables with the same name in the same scope is not allowed
- Possible, but potentially confusing, reusing a name in nested scopes

```
if ( . . . )
{
     int counter;

     if ( . . . )
     {
          int counter;
               . . . .
     }
}
```

SCOPE AND VARIABLE INITIALIZATION

LOCAL

- `int counter = 10;`

- Initialized each time the variable comes into scope

GLOBAL

- `int counter = 10;`

- Initialized once when the program is first loaded into memory