Lecture 7

Chapter 3.3
Topics

- Local variable
- Scope
- Procedural Abstraction
- Global constants
- Global variables
- Blocks
- Scope rule for nested blocks
- Variables declared in a for loop
- Issues with homework assignments
Local variables

• Variables declared within a body of a function are said to be local to that function
  – Example
    
    ```
    double totalCost( int numberOfItem, double price) {
        const double TAXRATE = 0.05; //5% sales tax
        double subTotal;
        ...
    }
    ```
  • TAXRATE and subTotal are local variables, they are not visible outside of this function
Scope

• We say the scope of TAXRATE and subTotal is the function totalCost
• The scope of any variable is the parenthesis that contain it.
Procedural Abstraction

• A person who uses a program should not need to know how it works. (Unless they wrote it and are in my class)
• To them it should appear as a magical black box that takes in input and then gives them the result it is intended to give.
• This is also known as information hiding.
Global constants

• A constant that can be accessed by any function in the program.
  – Example
    ```
    const double PI = 3.14159;
    ```

• Convention dictates that these are placed below your #defines and above your function prototypes.
Global variables

• Similar idea to global constants except that the modifier `const` isn’t used.
• It is considered bad programming practice to use these.
Blocks

• A compound statement with declarations.
• Basically a block is C++ code with declarations that is enclosed in braces.
• It is used to describe the scope of variables.
  – Example
  
  ```
  
  { 
      int theAnswer = 42;
      cout << “The answer to the meaning of life is “ << theAnswer << endl;
  } // As stated in “The hitchhikers guide to the galaxy”
  ```
Scope rule for nested blocks

- If the same variable is declared in each of the two blocks, then these are two different variables.
- The inner one can only be accessed by the inner block.
- The outer one can only be accessed by the outer block.
Variables declared in a for loop

• The ANSI/ISO C++ standard requires that any compiler that claims compliance treat variables declared in a **for** loop initializer as if it were local to the body of the loop.

• Example

  ```cpp
  for( int i = 0; i < 10; i++)
  ```

  — i is a local variable to the block of this for loop.
Issues with homework assignments

• Comments
• Spacing for code inside blocks
• Name variables
• Not finishing all the required objectives.