Section 1.3
Programming Conventions

1. Naming conventions
2. Indentation
3. Commenting
4. Coding practices

1.3.1 Naming Conventions

- Naming conventions are extremely important
  - these are rules known to professional programmers
  - helps others understand your code quickly
  - they are an important communication tool

- Constants
  - names are all uppercase
    - example: MAXNUM
  - compound names are separated by underscores
    - example: MAX_NUM

Naming Conventions (cont.)

- Variables
  - names begin with lowercase letter
    - example: **`num`**
  - compound names use camel-case or underscore style
    - example: **`numElements`** or **`num_elements`**
  - names must be descriptive without being overly long
    - use scope as context
    - example: inside **`Student`** class, use **`id`** instead of **`studentId`**
  - single letters can be used as temporary variables
    - example: loop counters **`i`**, **`j`**, **`k`**

Naming Conventions (cont.)

- Data types
  - names begin with uppercase letter
    - example: **`Student`** class
  - compound names use camel-case or underscore style
    - compound names for data types are less common
    - example: **`BookArray`** class

Naming Conventions (cont.)

- Functions
  - names begin with lowercase letter
    - example: **`sqrt()`**
  - compound names use camel-case or underscore style
    - example: **`computeMaxValue()`** or **`compute_max_value()`**
  - names must be short and descriptive

1.3.2 Indentation

- What is indentation?
  - white space before first character in each line of a program

- Why does it matter?
  - promotes code readability
    - shows which statements belong together as a block
    - allows easy detection of matching braces

Indentation (cont.)

- Terminology
  - block
    - sequence of statements between a pair of braces
  - nested block
    - a block contained within another block
  - nesting level of a block
    - depth from file scope at which a block is nested

Indentation (cont.)

- The indentation rules:
  - all statements within a block must be indented identically
  - all blocks at the same nesting level must have their statements indented identically to each other
  - there are **three** acceptable styles of brace indentation
    - the world does not need a fourth
  - do not mix styles! be consistent

Indentation (cont.)

- How much to indent?
  - by convention
    - two spaces
    - four spaces
    - one tab
      - not as portable between text editors
  - amount is not as important as consistency

1.3.3 Commenting

- Blocks of comments are required to document:
  - the program
  - each class
  - complex or critical statements or variables

Commenting (cont.)

- The program
  - a block of comments in the file containing the **`main`** function
  - specifies:
    - purpose of program
    - usage
      - including command line arguments
    - author(s)
    - revisions

Commenting (cont.)

- Each class
  - a block of comments before the class
  - specifies
    - purpose of class
    - descriptions of complex or critical members (attributes or functions)

Commenting (cont.)

- Complex or critical statements or variables
  - inline comments before the statement or variable declaration
    - keep this short
  - may specify
    - summary of algorithm used
    - ranges of variable

1.3.4 Coding Practices

- Many professional coding practices, some examples:
  - do design classes into object categories
  - don’t use of `structs`, do use classes instead
  - don’t use global variables
  - don’t use global functions extensively
  - don’t pass objects by value
    - this makes unnecessary copies of data, and slows down the program
  - do return data using parameters, not using return values
  - do reuse code wherever possible
    - don’t copy and paste code, call a common function instead
  - do perform basic error checking