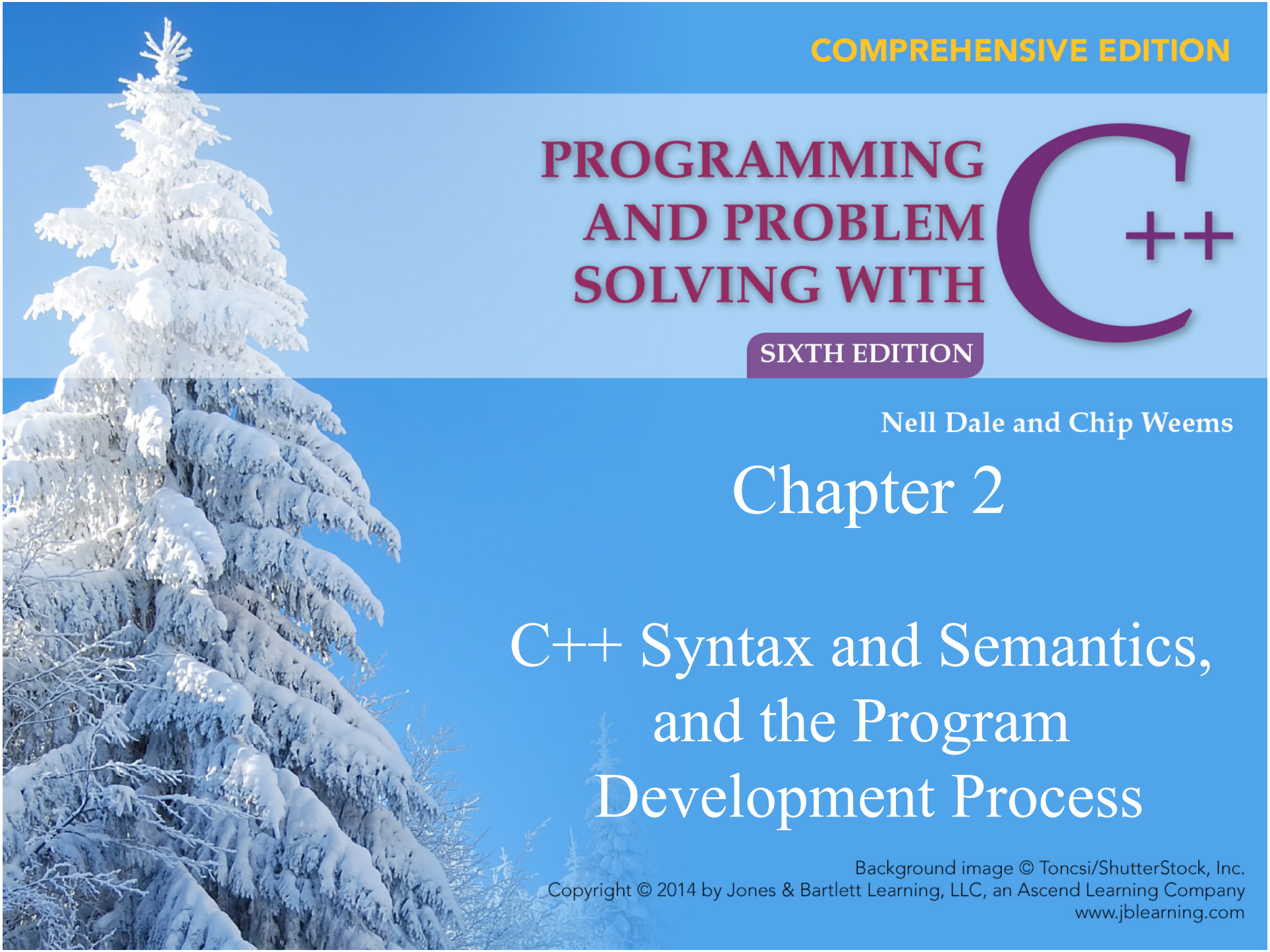


COMPREHENSIVE EDITION

PROGRAMMING AND PROBLEM SOLVING WITH C++

SIXTH EDITION

Nell Dale and Chip Weems

Chapter 2

C++ Syntax and Semantics, and the Program Development Process

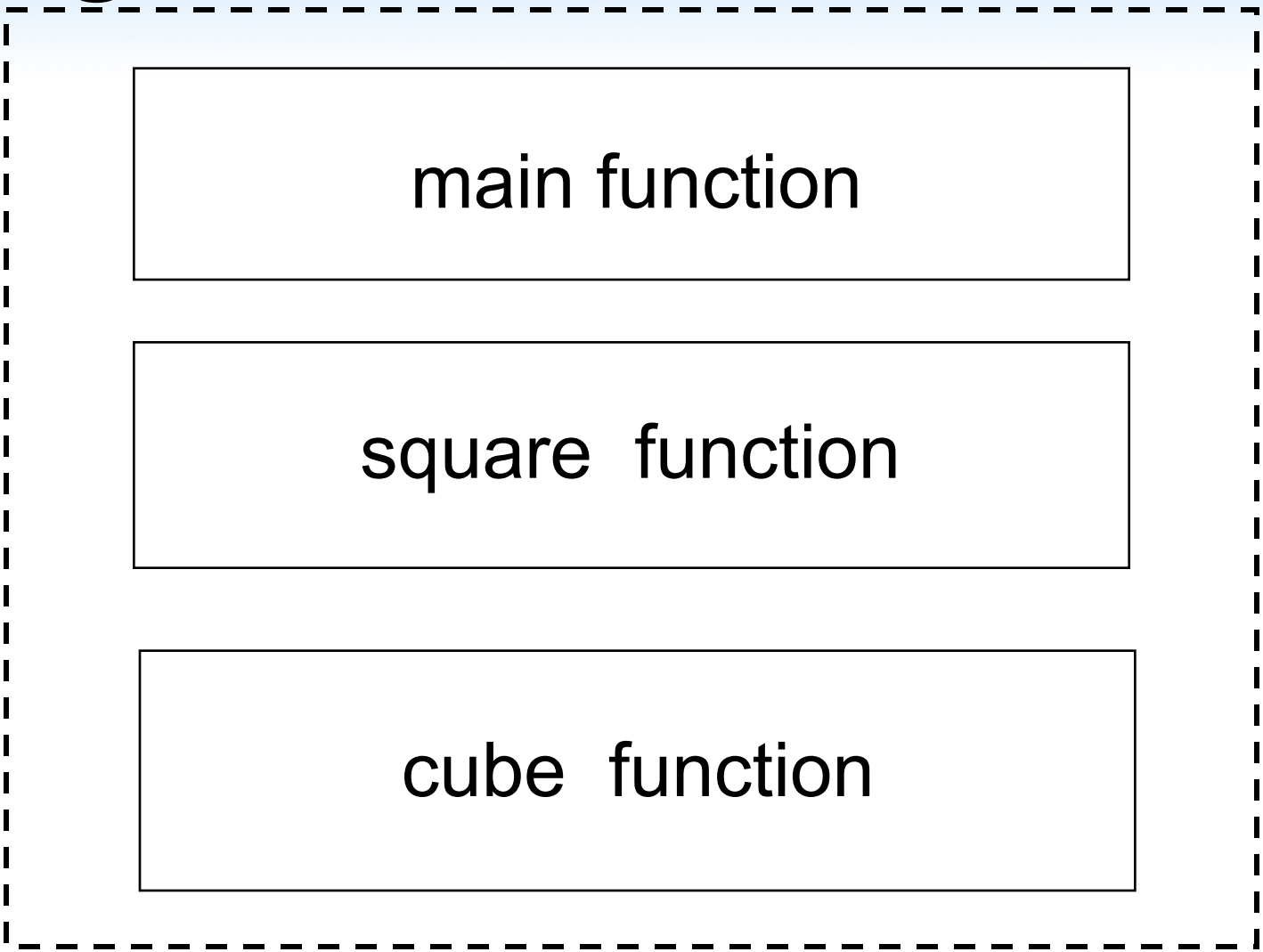
Chapter 2 Topics

- Programs Composed of Several Functions
- Syntax Templates
- Legal C++ Identifiers
- Assigning Values to Variables
- Declaring Named Constants
- String Concatenation
- Output Statements
- C++ Program Comments

A C++ program is a collection of one or more functions

- There must be a function called main()
- Execution always begins with the first statement in function main()
- Any other functions in your program are subprograms and are not executed until they are called

Program With Several Functions



Program With Three Functions

```
#include <iostream>

int Square(int);          // Declares these two
int Cube(int);            // value-returning functions

using namespace std;

int main()
{
    cout << "The square of 27 is "
         << Square(27)<< endl;   // Function call

    cout << "The cube of 27 is "
         << Cube(27)<< endl;     // Function call
    return 0;
}
```

Rest of Program

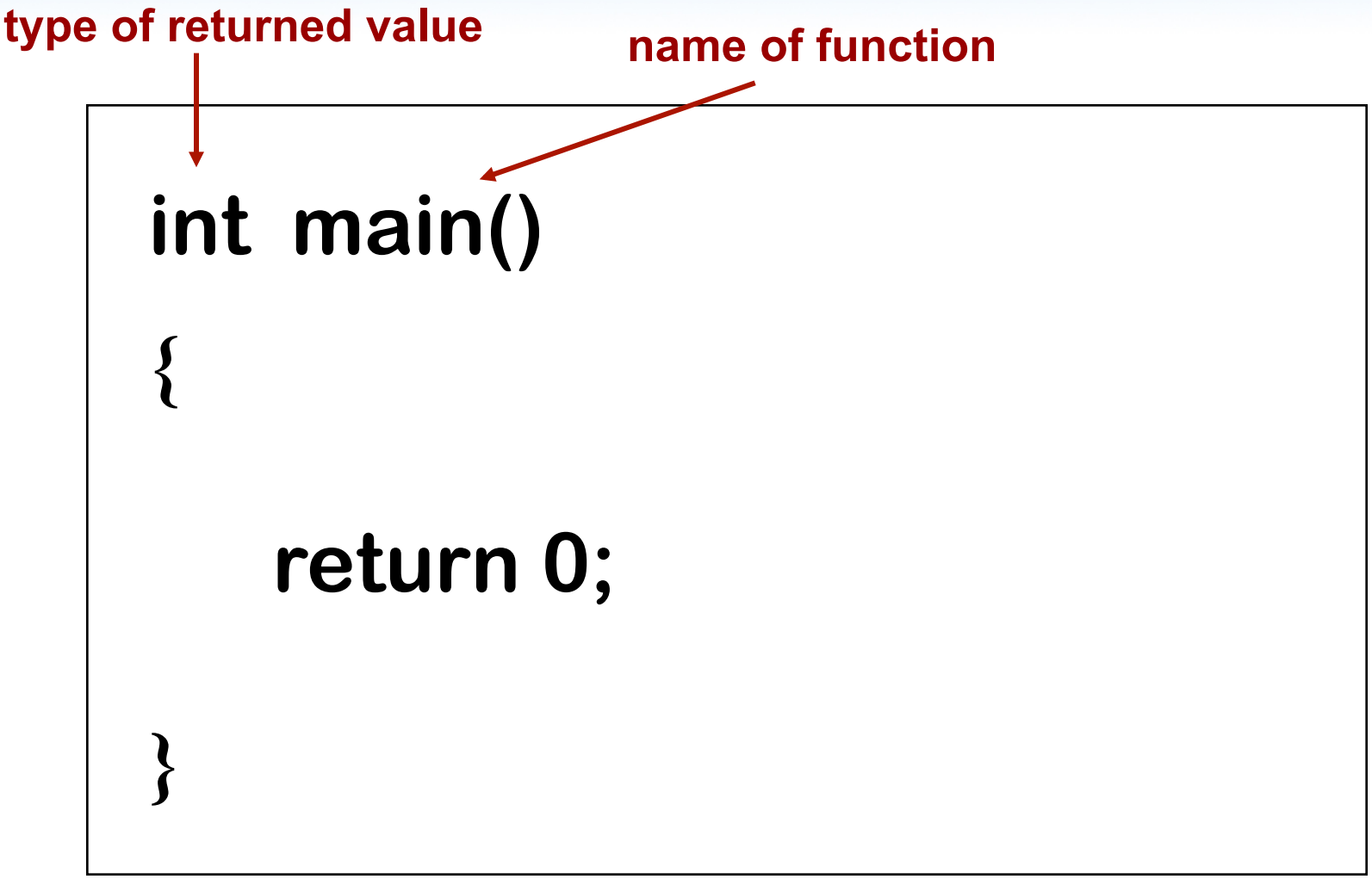
```
int Square(int n)
{
    return n * n;
}


int Cube(int n)
{
    return n * n * n;
}
```

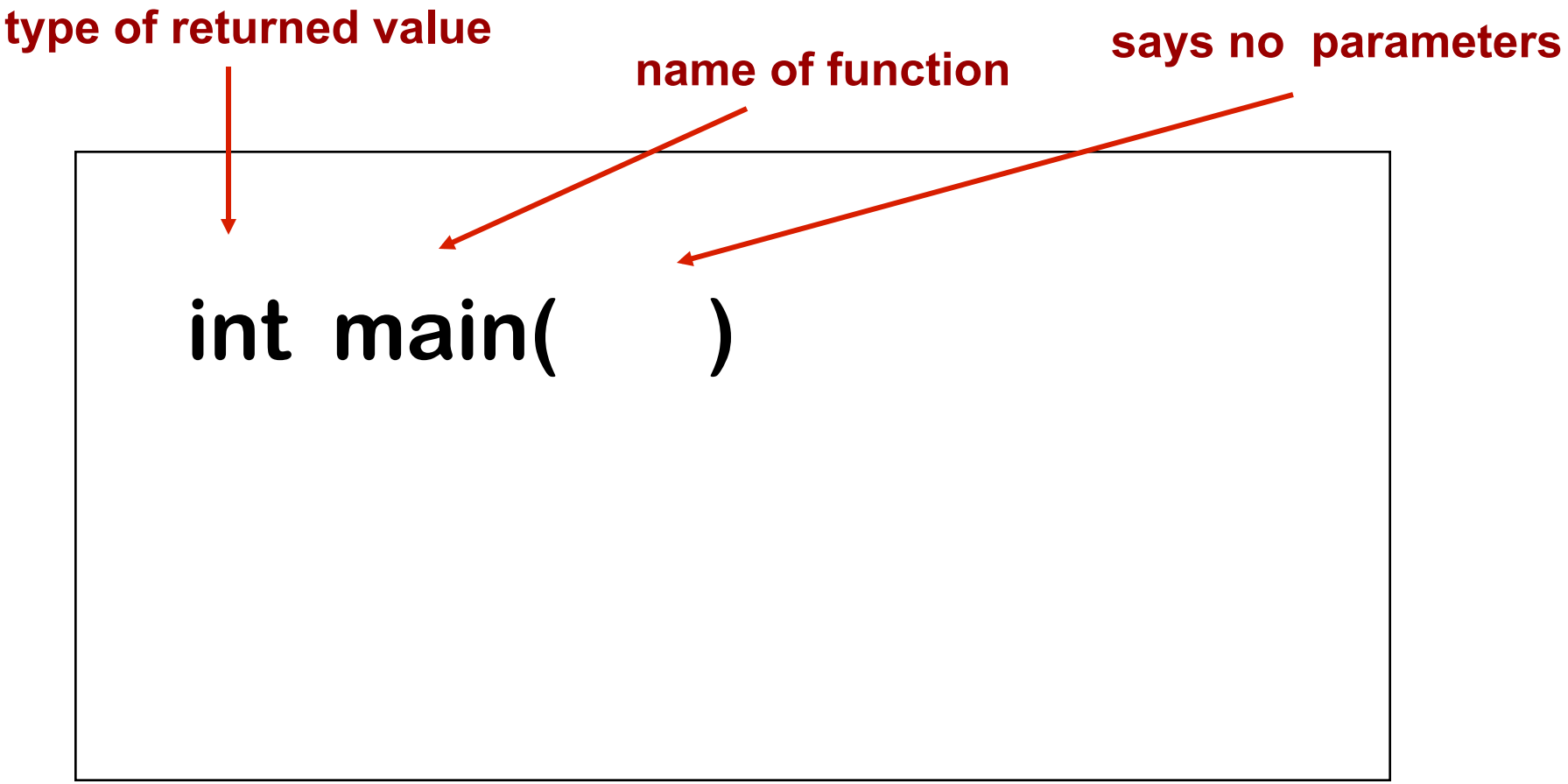
Output of program

```
The square of 27 is 729
The cube of 27 is 19683
```

Shortest C++ Program



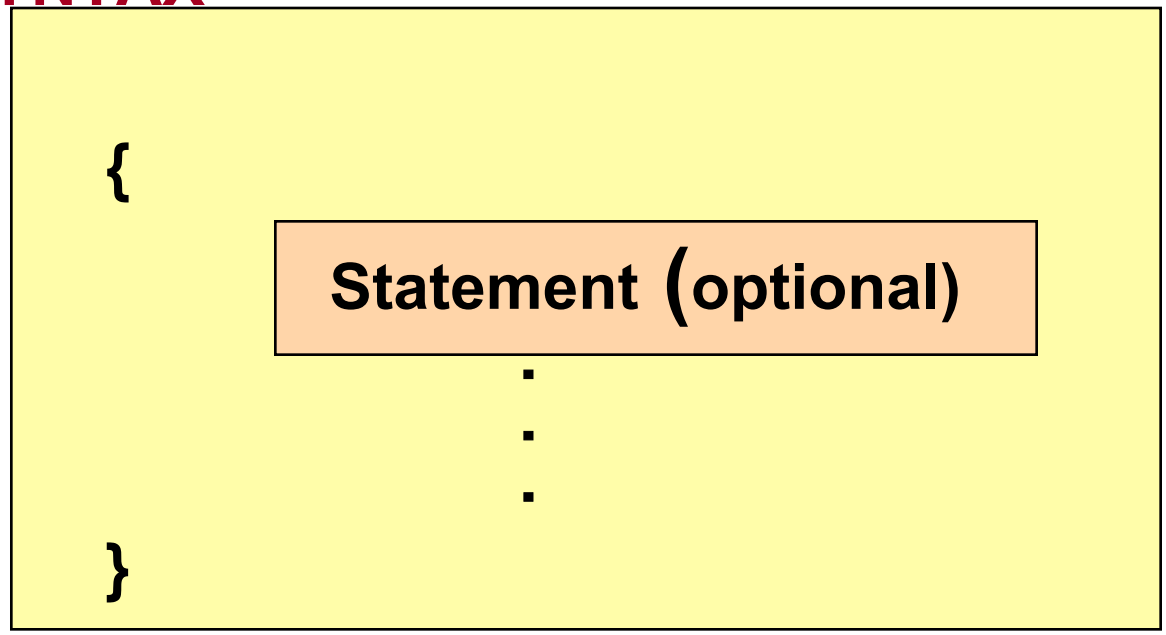
What is in a heading?



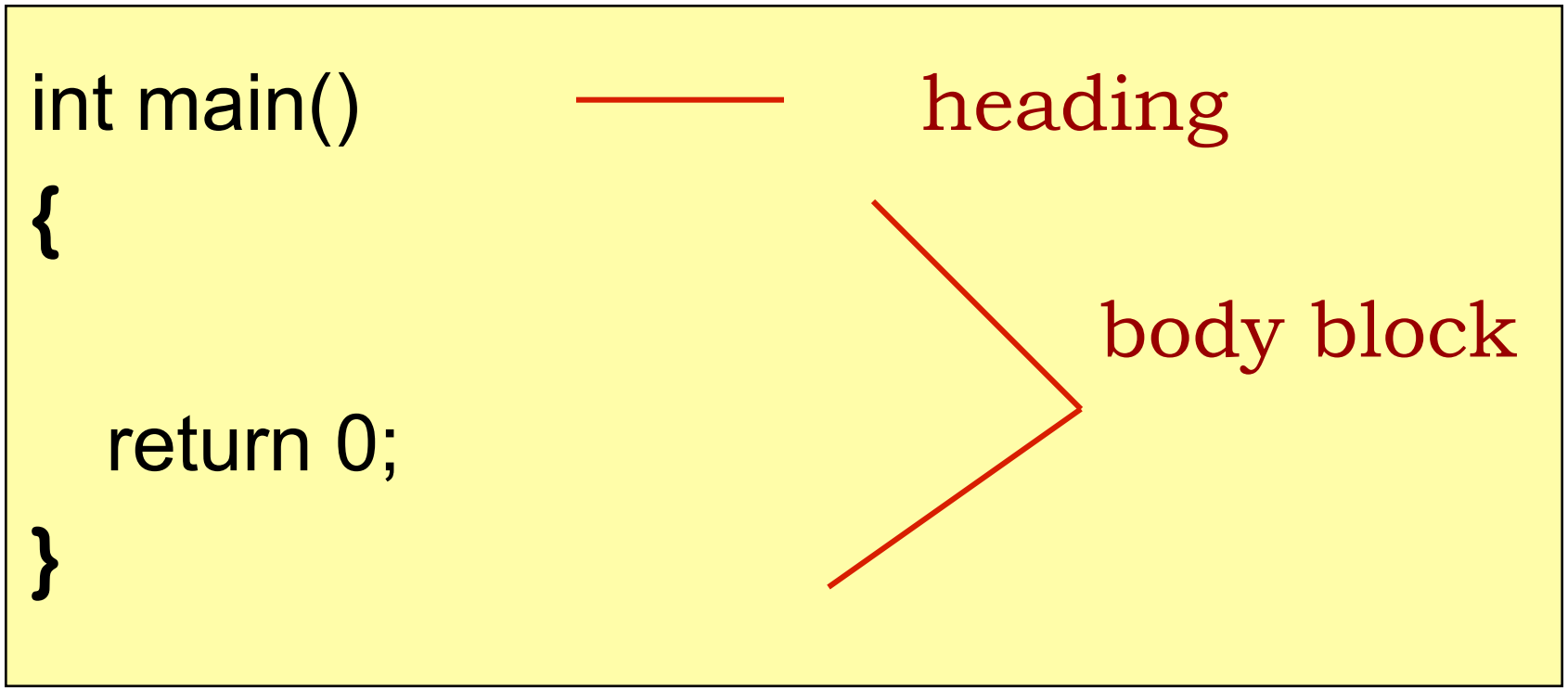
Block(Compound Statement)

- A **block** is a sequence of zero or more statements enclosed by a pair of curly braces { }

SYNTAX



Every C++ function has 2 parts



What is an Identifier?

An identifier is the name used for a data object(a variable or a constant), or for a function, in a C++ program

Beware: C++ is a case-sensitive language

Using meaningful identifiers is a good programming practice

Identifiers

- An identifier must start with a letter or underscore, and be followed by zero or more letters

  (A-Z, a-z), digits(0-9), or underscores _

- VALID

  age_of_dog taxRateY2K

  PrintHeading ageOfHorse

- NOT VALID *(Why?)*

  age# 2000TaxRate Age-Of-Cat

More About Identifiers

- Some C++ compilers recognize only the first 32 characters of an identifier as significant

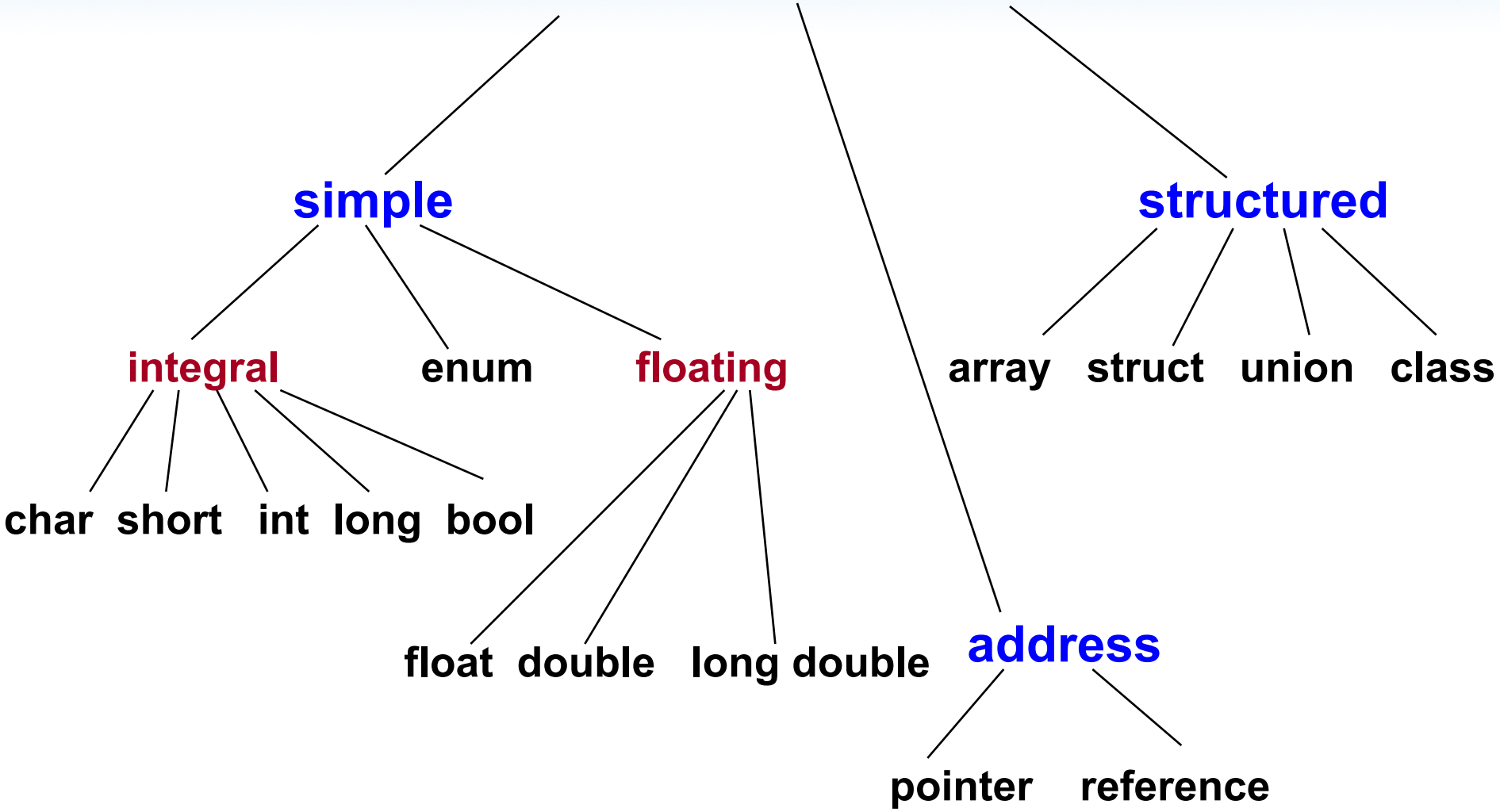
- Then these identifiers are considered the same:

```
age_Of_This_Old_Rhinoceros_At_My_Zoo
age_Of_This_Old_Rhinoceros_At_My_Safari
```

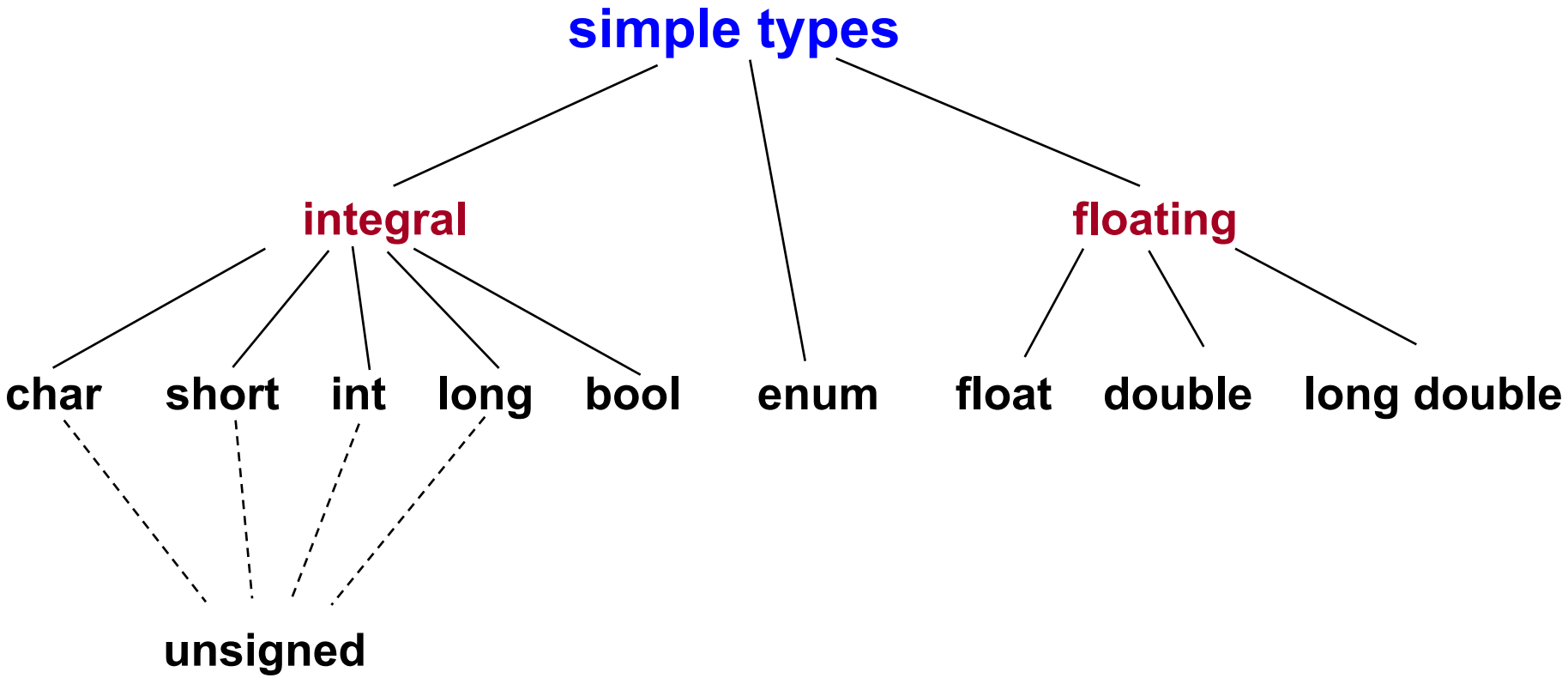
- Consider these:

```
Age_Of_This_Old_Rhinoceros_At_My_Zoo
age_Of_This_Old_Rhinoceros_At_My_Zoo
```

C++ Data Types



C++ Simple Data Types



Standard Data Types in C++

- **Integral Types**
  - represent positive and negative integers
  - declared as **int**, **short**, or **long**
- **Floating Types**
  - represent real numbers with a decimal point
  - declared as **float**, or **double**
- **Character Types**
  - represent single alphanumerical character---a letter, digit, or a special symbol
  - declared as **char**

Samples of C++ Data Values

int sample values

4578 -4578 0

float sample values

95.274 95. .265

char sample values

'B' 'd' '4' '?' '*'

What is a Variable?

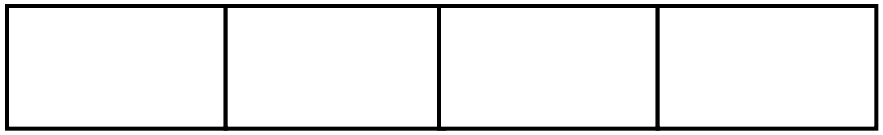
- A **variable** is a location in memory that can be referred to by an identifier and in which a **data value that can be changed** is stored

- Declaring a variable means specifying both its name and its data type

What Does a Variable Declaration Do?

```
int   ageOfDog;
float taxRate;
char  middleInitial;
```

A declaration tells the compiler to allocate enough memory to hold a value of this data type and to associate the identifier with this location



4 bytes for taxRateY2K

1 byte for middleInitial

C++ Data Type String

- A string is a sequence of characters enclosed in double quotes
- Sample string values

  “Hello” “Year 2000” “1234”
- The empty string (null string)contains no characters and is written as ""

More About Type String

- A string is not a built-in(standard)type
  - It is a programmer-defined data type
  - It is provided in the C++ standard library
- String operations include
  - Comparing 2 string values
  - Searching a string for a particular character
  - Joining one string to another

What is a Named Constant?

- A named constant is a location in memory that can be referred to by an identifier and in which a data value that cannot be changed is stored

Valid constant declarations

```
const   string   STARS = "****";
const   float    NORMAL_TEMP = 98.6;
const   char     BLANK = ' ';
const   int      VOTING_AGE = 18;
const   float    MAX_HOURS = 40.0;
```

Giving a Value to a Variable

Assign(give)a value to a variable by using the assignment operator =

Variable declarations

```
string   firstName;
char     middleInitial;
char     letter;
int      ageOfDog;
```

Valid assignment statements

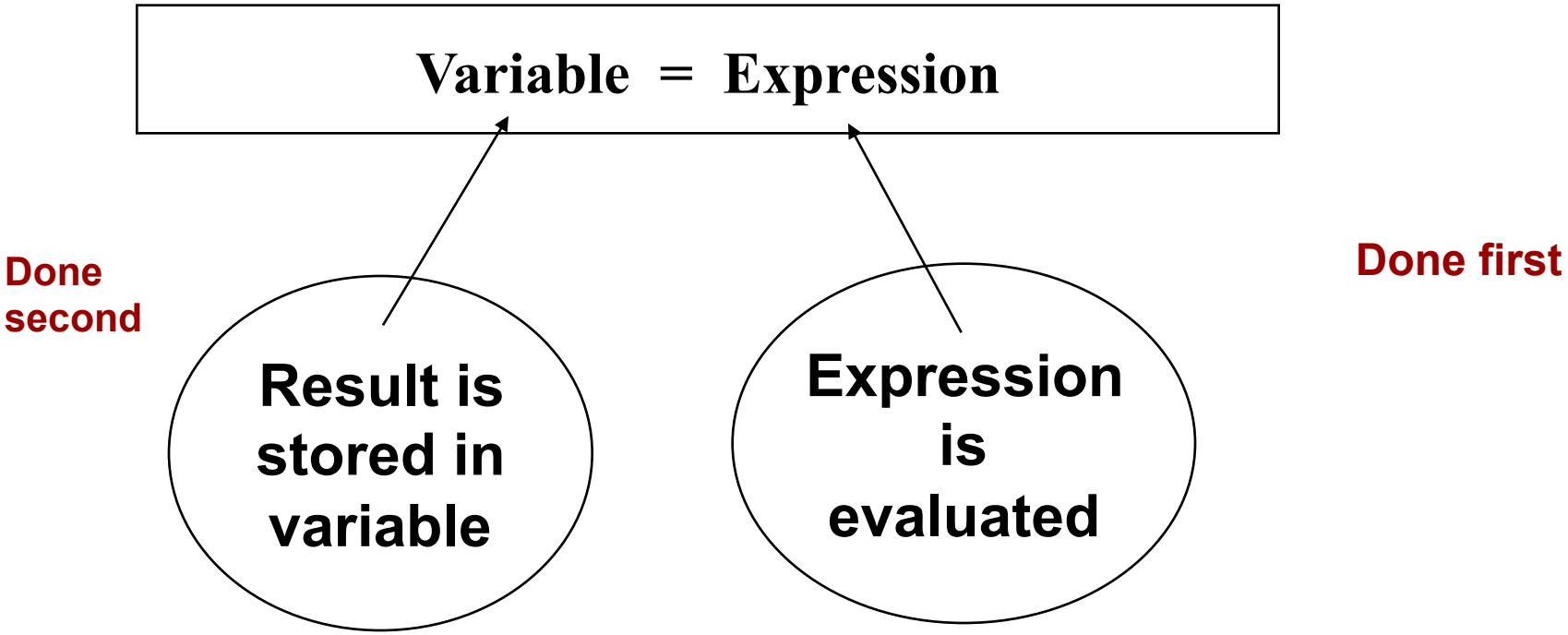
```
firstName = "Fido";
middleInitial = 'X';
letter = middleInitial;
ageOfDog = 12;
```

What is an Expression in C++?

- An **expression** is a valid arrangement of variables, constants, and operators
- In C++ each expression can be evaluated to compute a value of a given type
- The value of the expression

  9 + 5 is 14

Assignment Operator Syntax



String Concatenation(+)

- **Concatenation is a binary operation that uses the + operator**
- **At least one of the operands of the + operator must be a string variable or named string constant--the other operand can be a string literal or a char variable, literal, or constant**

Concatenation Example

```
const   string WHEN = "Tomorrow";
 const   char  EXCLAMATION = '!';
 string   message1;
 string  message2;

 message1 = "Yesterday ";
 message2 = "and ";
 message1 = message1 + message2 +
              WHEN + EXCLAMATION;
```

Insertion Operator(<<)

- Variable cout is predefined to denote an output stream that goes to the standard output device(display screen)
- The insertion operator << called "put to" takes two operands
- The left operand is a stream expression, such as cout
- The right operand is an expression of a simple type or a string constant

Output Statements

SYNTAX

```
cout << Expression << Expression . . .;
```

These examples yield the same output:

```
cout << "The answer is ";
cout << 3 * 4;
```

```
cout << "The answer is " << 3 * 4;
```

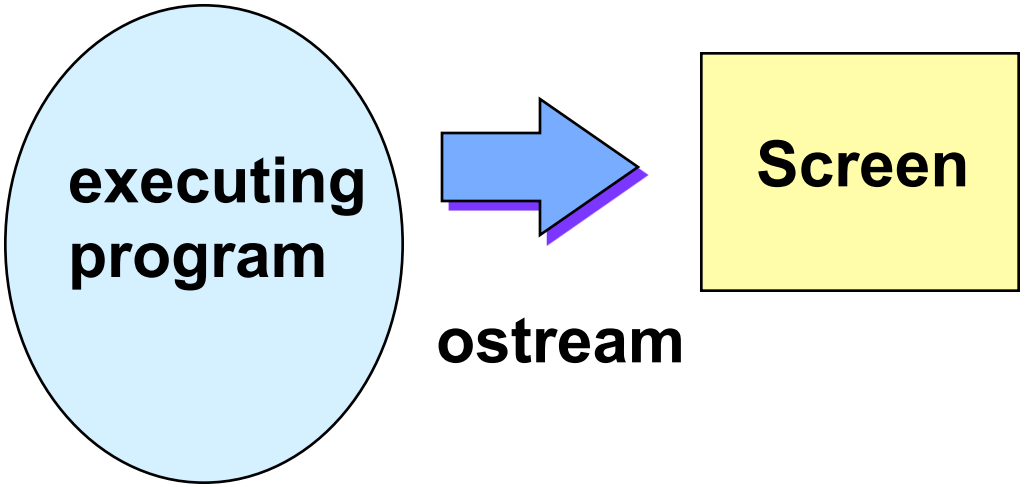
Is compilation the first step?

- **No; before your source program is compiled, it is first examined by the C++ Preprocessor that:**
  - **removes all comments from source code**
  - **handles all preprocessor directives--they begin with the # character such as**

    #include <iostream>
  - **This include tells the preprocessor to look in the standard include directory for the header file called iostream and insert its contents into your source code**

No I/O is built into C++

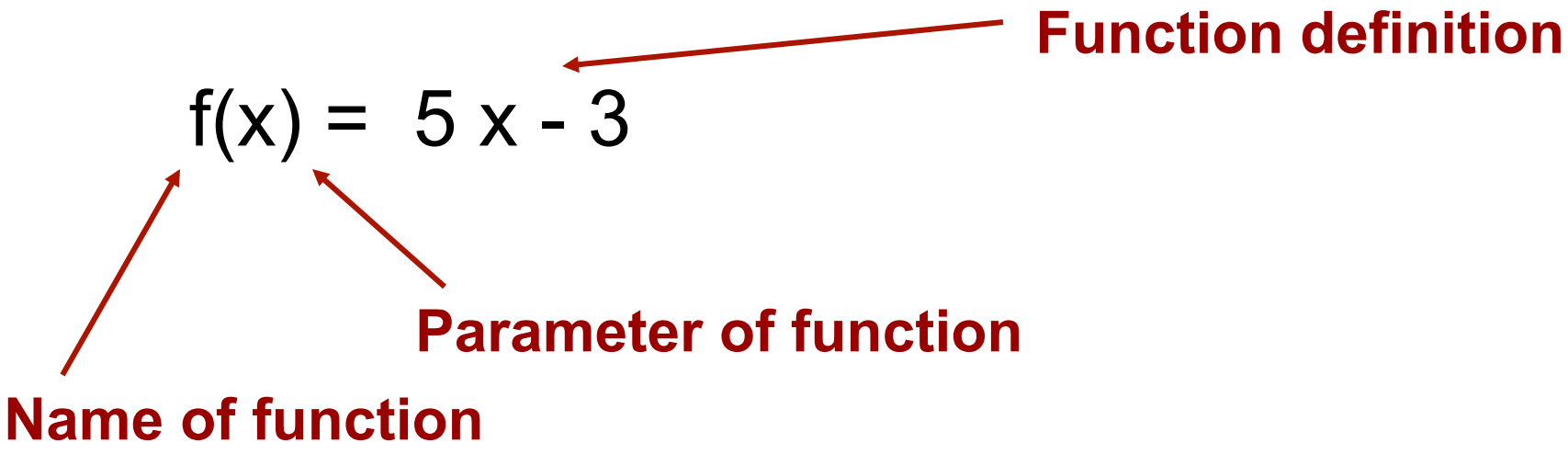
- Instead, a library provides an output stream



Using Libraries

- A library has two parts
  - Interface (stored in a header file) tells what items are in the library and how to use them
  - Implementation (stored in another file) contains the definitions of the items in the library
- #include <iostream>
  - Refers to the header file for the *iostream* library needed for use of cout and endl.

Function Concept in Math



When x = 1, f(x)= 2 is the returned value

When x = 4, f(x)= 17 is the returned value

Returned value is determined by the function definition and by the values of any parameters

C++ Program

```
// ********************************************************
//  PrintName program
//  This program prints a name in two different formats
// ********************************************************

#include <iostream>          // for cout and endl
#include <string>            // for data type string

using namespace std;

const  string  FIRST = "Herman";  // Person's first name
const  string  LAST =  "Smith";   // Person's last name
const  char    MIDDLE = 'G';      // Person's middle initial
```

C++ Code Continued

```
int  main()
{
     string    firstLast;    // Name in first-last format
     string    lastFirst;    // Name in last-first format


     firstLast = FIRST + " " + LAST;
     cout  << "Name in first-last format is "  << endl
           << firstLast  <<  endl;

     lastFirst = LAST + ", " + FIRST + ' ';
     cout  << "Name in first-last format is "  << endl
           << lastFirst  <<  MIDDLE  <<  '.'  <<  endl;

     return  0;
}
```

Output of Program

Name in first-last format is
Herman Smith

Name in last-first-initial format is
Smith, Herman G.

Software Maintenance Tips When Modifying Complex Code

- Break a long block of code into smaller chunks that have distinct purposes
- Identify portions of the code that you know you can ignore
- Focus on those code sections that are clearly related to the maintenance task
- Make sure you understand which changes are required including asking questions about unclear matters

Software Maintenance Tips When Modifying Complex Code

- Consider the major steps (e.g., an application whose steps are input, process, and output) you have identified in the existing code
- Then establish how you would solve the maintenance task within the overall approach of the existing code
- Examine and evaluate how your changes affect other parts of the application
- Document your changes to the code

Creating a Chessboard

Problem **Your college is hosting a chess tournament, and the people running the tournament want to record the final positions of the pieces in each game on a sheet of paper with a chessboard preprinted on it. Your job is to write a program to preprint these pieces of paper. The chessboard is an eight-by-eight pattern of squares that alternate between black and white, with the upper left square being white. You need to print out squares of light characters (spaces) and dark characters(such as *) in this pattern to form the chessboard.**

Chessboard

Constants

- ***Name Value*** — ***Function***
  - `BLACK` `'********'` — Characters forming one line of a black square
  - `WHITE` `'        '` — Characters forming one line of a white square

Variables

- ***Name Data Type*** — ***Description***
  - `whiteRow string` — A row beginning with a white square
  - `blackRow string` — A row beginning with a black square

Algorithm

Repeat four times
  Output five whiteRows
  Output five blackRows

C++ Program

```
//*****************************************************
// Chessboard program
// This program prints a chessboard pattern that is
// built up from basic strings of white and black
// characters.
//*****************************************************
#include <iostream>
#include <string>
using namespace std;
const string BLACK = "********"; // Define black square line
const string WHITE = "        "; // Define white square line
```

C++ Program

```
int main()
{
    string whiteRow;   // White square beginning row
    string blackRow;   // Black square beginning row
    // Create a white-black row
    whiteRow = WHITE + BLACK + WHITE + BLACK +
               WHITE + BLACK + WHITE + BLACK;
    // Create a black-white row
    blackRow = BLACK + WHITE + BLACK + WHITE +
               BLACK + WHITE + BLACK + WHITE;
```

C++ Program

```
    // Print five white-black rows
    cout << whiteRow << endl;
    cout << whiteRow << endl;
    cout << whiteRow << endl;
    cout << whiteRow << endl;
    cout << whiteRow << endl;

    // Print five black-white rows
    cout << blackRow << endl;
    cout << blackRow << endl;
    cout << blackRow << endl;
    cout << blackRow << endl;
    cout << blackRow << endl;
   // Print rest of the rows
    ...
    return 0;
}
```