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

What is Computer Science?

The Computing Curriculum 1991 (ACM/IEEE)

- Algorithms and Data Structures
- Architecture
- Artificial Intelligence and Robotics
- Database and Information Retrieval
- Human-Computer Communication
- Numerical and Symbolic Computation
- Operating Systems
- Programming Languages
- Software Engineering
- Social and Professional Context

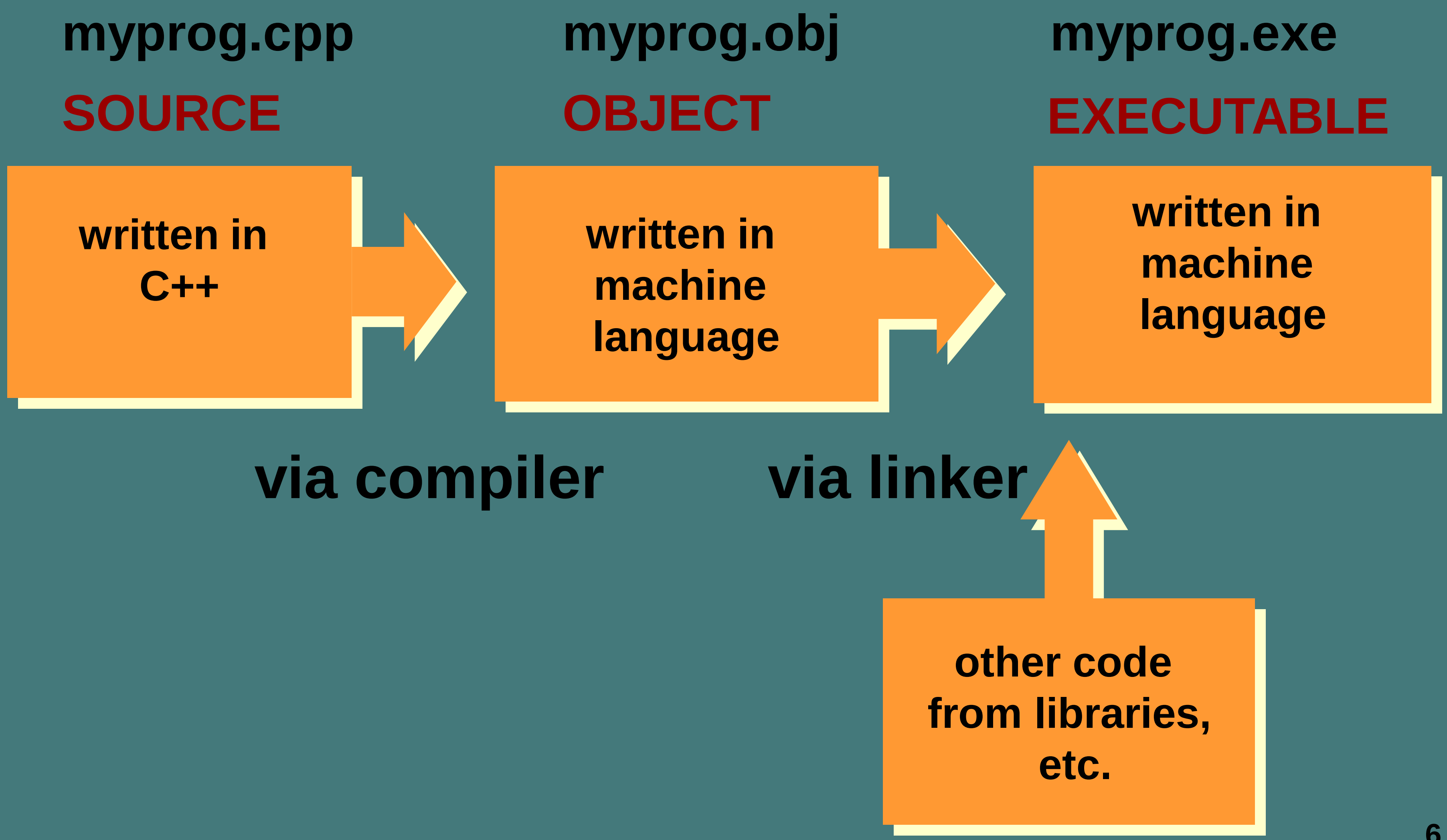
Computing Profession Ethics

- copy software only with permission from the copyright holder
- give credit to another programmer by name whenever using his/her code
- use computer resources only with permission
- guard the privacy of confidential data
- use software engineering principles to develop software free from errors

Some C++ History

- 1972 : Dennis Ritchie & Brian Kernighan at Bell Labs designs C and 90% of UNIX is then written in C
- Late 70's : OOP becomes popular
- Bjarne Stroustrup at Bell Labs adds features to C to form "C with Classes"
- 1983 : Name C++ first used
- 1998 : ISO/ANSI standardization of C++

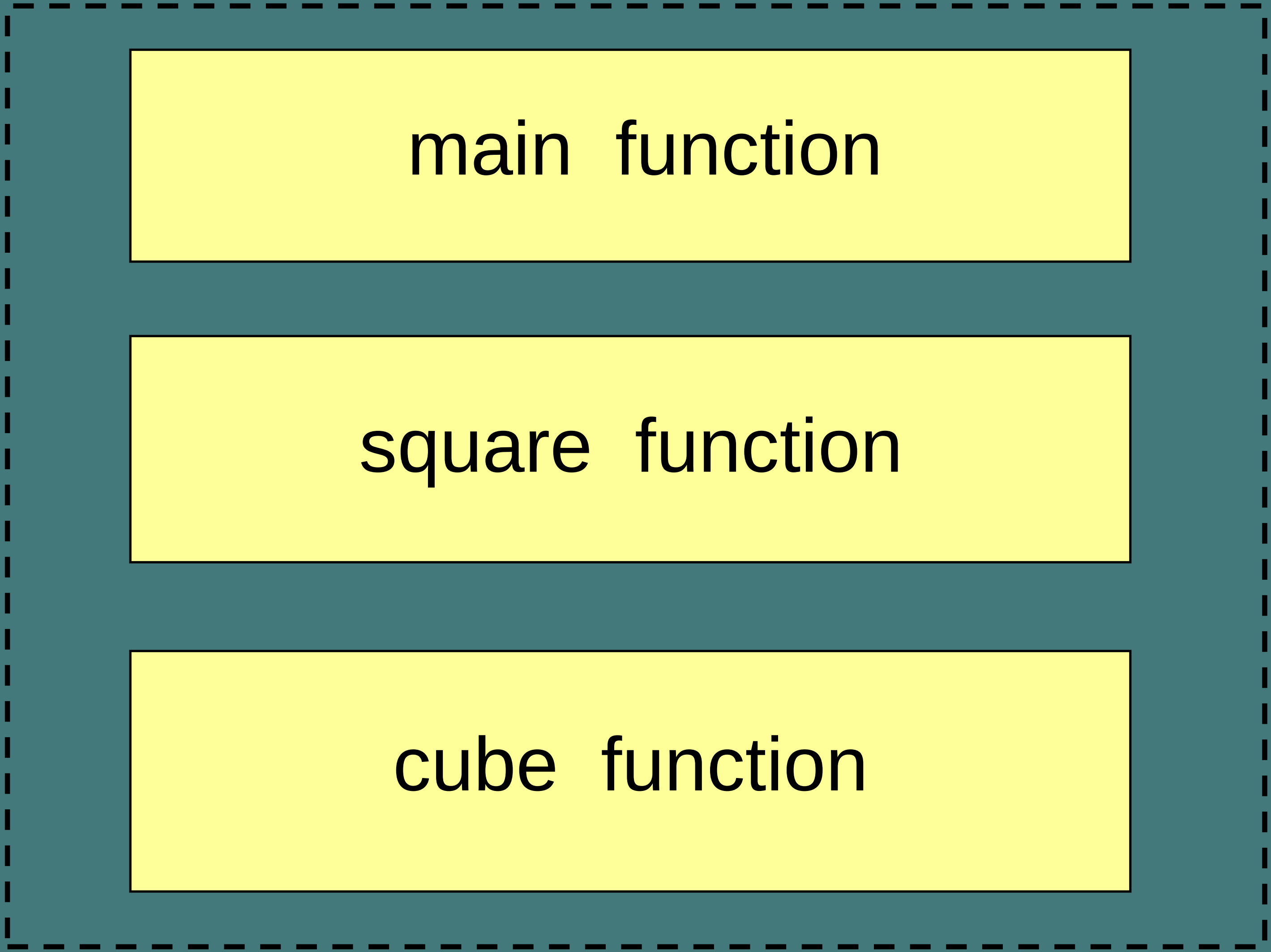
Three C++ Program Stages



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;
}
```

Local variables

↓

n $\boxed{27}$

Square $\boxed{729}$

n $\boxed{27}$

Cube $\boxed{19683}$

Output of program

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

Shortest C++ Program

type of returned value

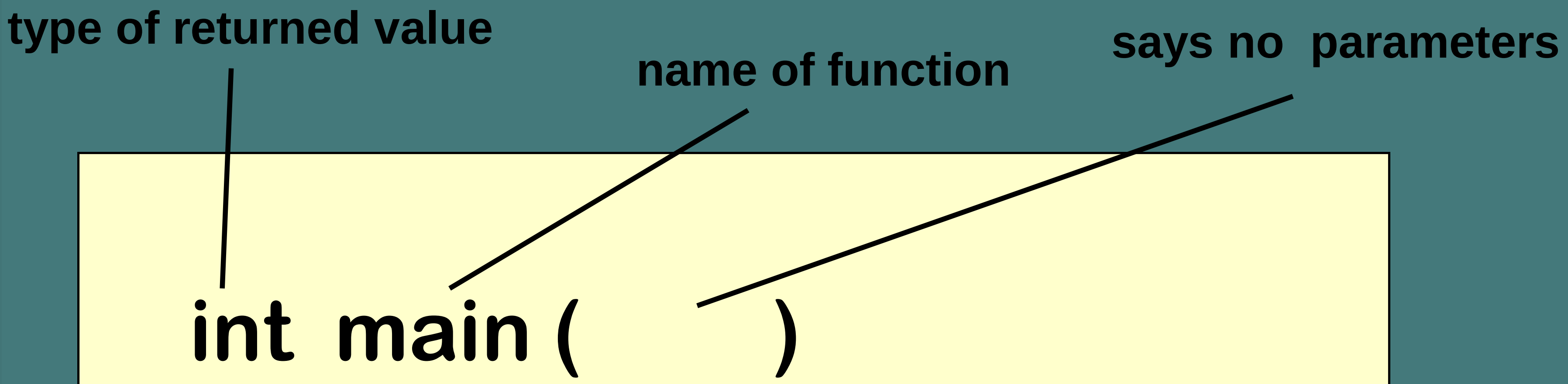
name of function

```
int  main (  )
{

    return 0;

}
```

What is in a heading?



```
int  main (      )
```

Block (Compound Statement)

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

SYNTAX

```
{
        Statement   (optional)
              .
              .
              .
}
```

Every C++ function has 2 parts

```
int main ( )                    ——— heading
{
                                    body block
    return 0;
}
```

演算法

```
function larger( a,b; rst );
//find the larger of a and b, store the result in rst
  [  if (a >=b)
          rst ← a;
     else rst ← b ;   ]

main( )
  Begin
   step 1: input a1 ; L ← a1 ; j←2;
   step 2: while (j<=10 )
            [ input aj ;
              larger(L, aj; rst); //叫用函數,得到較大者
              L ← rst ;           //更新目前最大的值
              j ← j+1 ;
            ]
   step 3: output L ;
 End
```

C++ Programming

```
#include <iostream>
// function larger 的表頭與內容


using namespace std ;

int main( )
{
     //main()的內容
     return 0;
}
```

演習課實作:

請同學參考上面的演算法，完成函數larger()的程式碼並在main() 程式中測試-- 找出十個整數的最大值。

Euclidean Algorithm :

Function findGCD(m, n; d)

Input : m, n positive integers
Returned : d, the Greatest Common Divisor of m and n .

```
STEP 1.  Input m, n
STEP 2.  Divide m by n,  let r be the remainder .
STEP 3.  If r = 0,  let  d ← n and STOP ;
             otherwise
             let  m ← n , n ← r  and GOTO STEP 2.
STEP 4.  return d
```

演習課實作:

請同學參考上面的演算法，完成函數findGCD () 並在main() 程式中測試-- 找出三個正整數的最大公約數。

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.
- 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

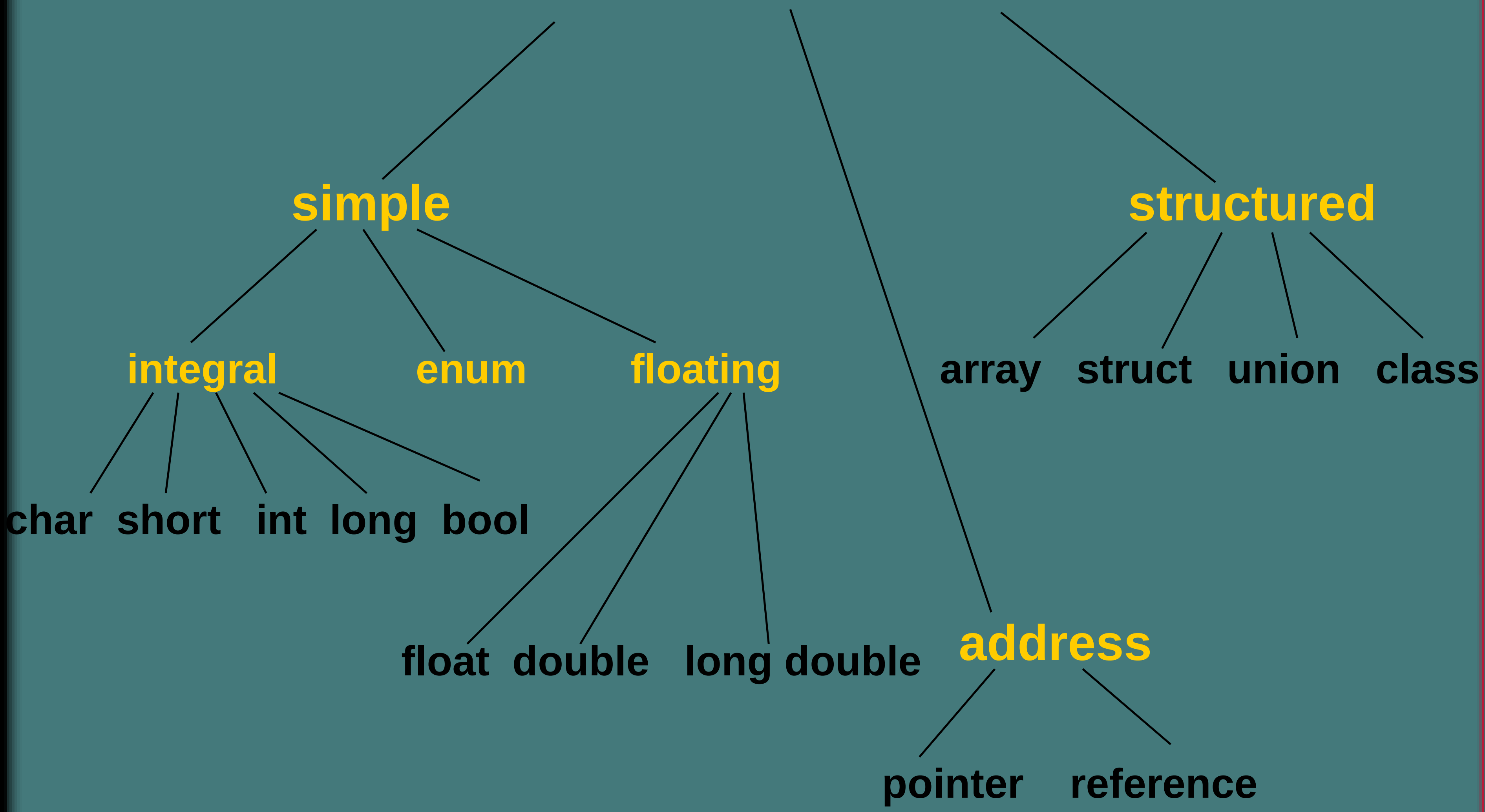
  age_Of_This_Old_Rhinoceros_At_My_Safari

- consider these:

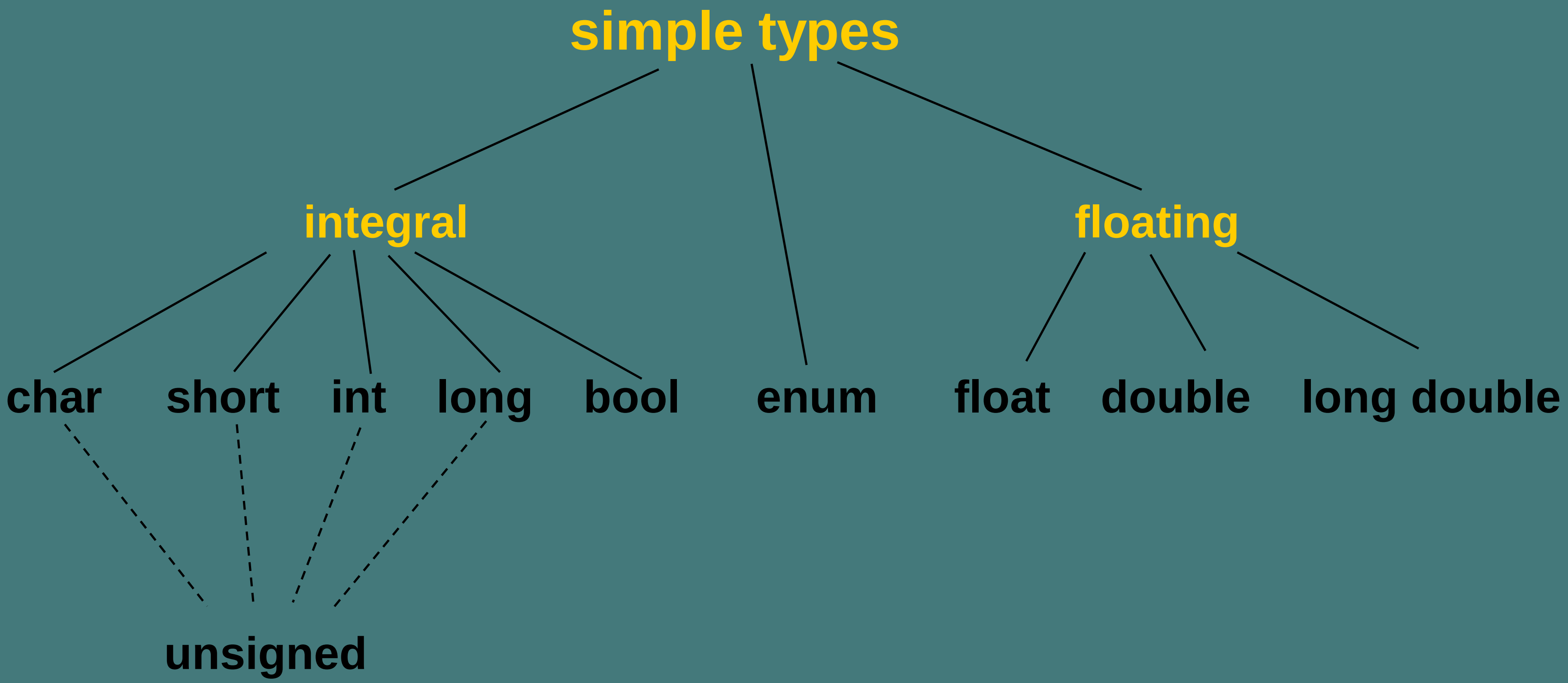
  Age_Of_This_Old_Rhinoceros_At_My_Zoo

  age_Of_This_Old_Rhinoceros_At_My_Zoo

C++ Data Types


simple
structured
integral
enum
floating
array
struct
union
class
char
short
int
long
bool
float
double
long double
address
pointer
reference

C++ Simple Data Types



Type (Integral)	Size in Bytes	Minimum* value	Maximum* value
char	1	-128	127
unsigned char	1	0	255
short	2/1*	-32768/-128	32767/127
unsigned short	2/1*	0	65535/255
int	2	-32768	32767
unsigned int	2	0	65535
long	4	-2147483648	2147483647
unsigned long	4	0	4294967295

*: depend on machine

Type (floating-point)	Size in Bytes*	Minimum* positive value	Maximum* positive value
float	4	3.4E-38	3.4E+38
double	8	1.7E-308	1.7E+308
long double	10	3.4E-4932	3.4E+4932

*: depend on machine

Standard Data Types in C++

- **Integral Types**
  - represent whole numbers and their negatives
  - declared as int, short, or long
- **Floating Types**
  - represent real numbers with a decimal point
  - declared as float, or double
- **Character Types**
  - represent single characters
  - 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?

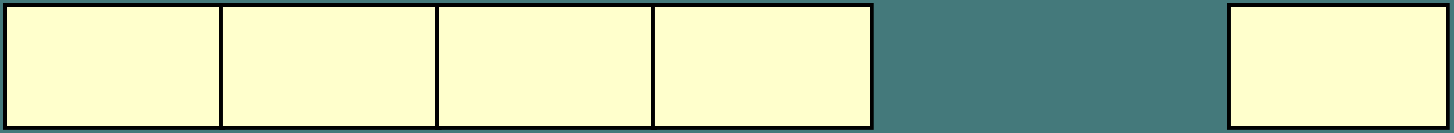
- A variable is a location in memory which we can refer to by an identifier, and in which a data value that can be changed is stored.

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

What Does a Variable Declaration Do?

```
int   ageOfDog;
float taxRateY2K;
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

- string sample values

  `“Hello”    “Year 2000”    “1234”`

- the empty string (null string) contains no characters and is written as `“”`

More About Type String

- 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 we can refer 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

You can 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

```
Variable  =  Expression
```

First, Expression on right is evaluated.

Then the resulting value is stored in the memory location of Variable on left.

NOTE: An automatic type coercion occurs after evaluation but before the value is stored if the types differ for Expression and Variable

String Concatenation (+)

- concatenation is a binary operation that uses the + operator
- at least one of the operands must be a string variable or named constant--the other operand can be string type or char type

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 2 operands

- the left operand is a stream expression, such as cout. The right operand is an expression of 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?

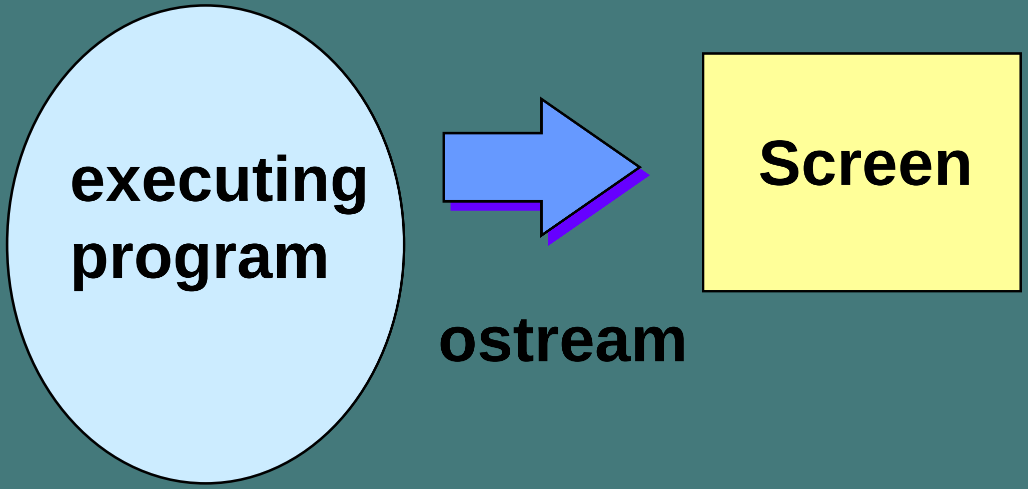
- No. Before your source program is compiled, it is first examined by the preprocessor to
  - remove all comments from source code
  - handle all preprocessor directives--they begin with the # character such as

    #include <iostream>

    - tells 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 2 parts

  Interface (stored in a header file) tells what items are in the library and how to use them.

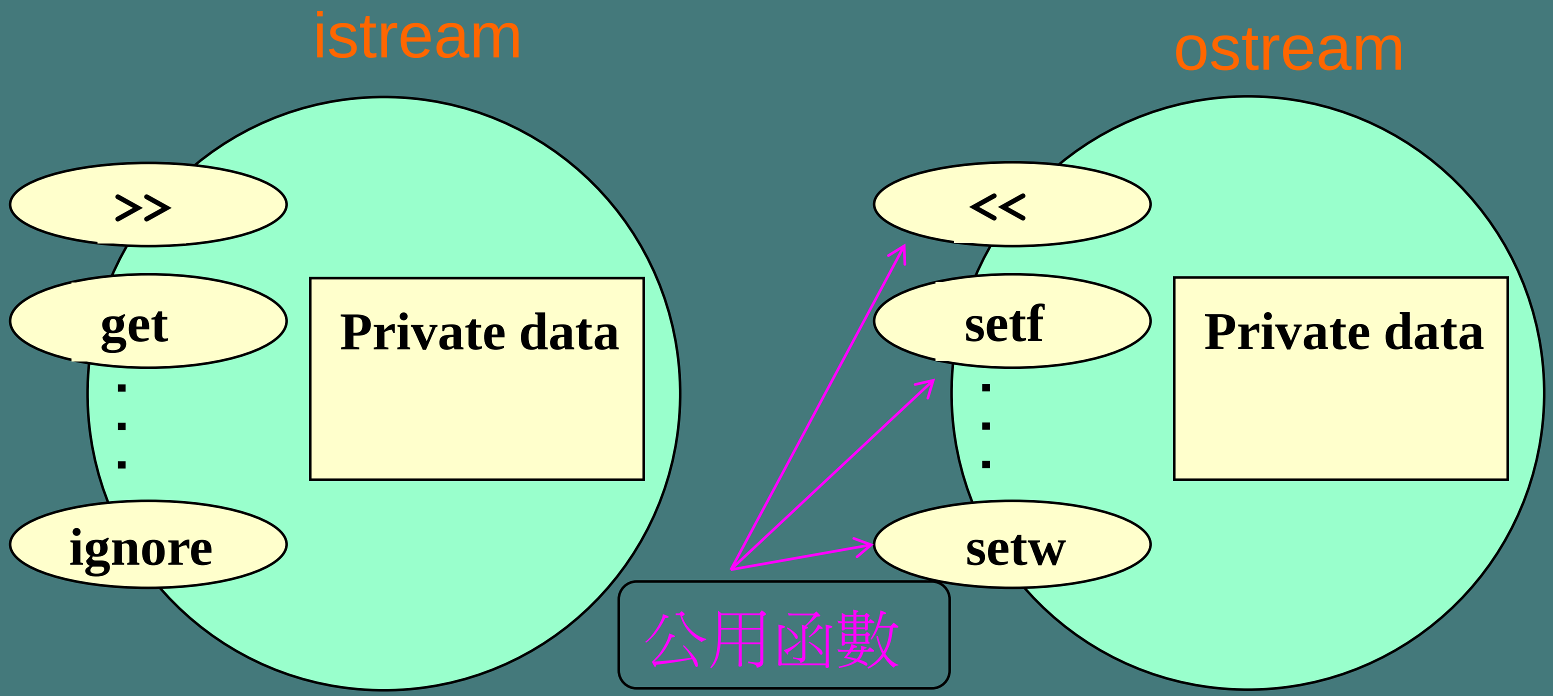
  Implementation (stored in another file) contains the definitions of the items in the library.

- #include <iostream>

  Refers to the header file for the *iostream* library needed for use of cout and endl.

<iostream>

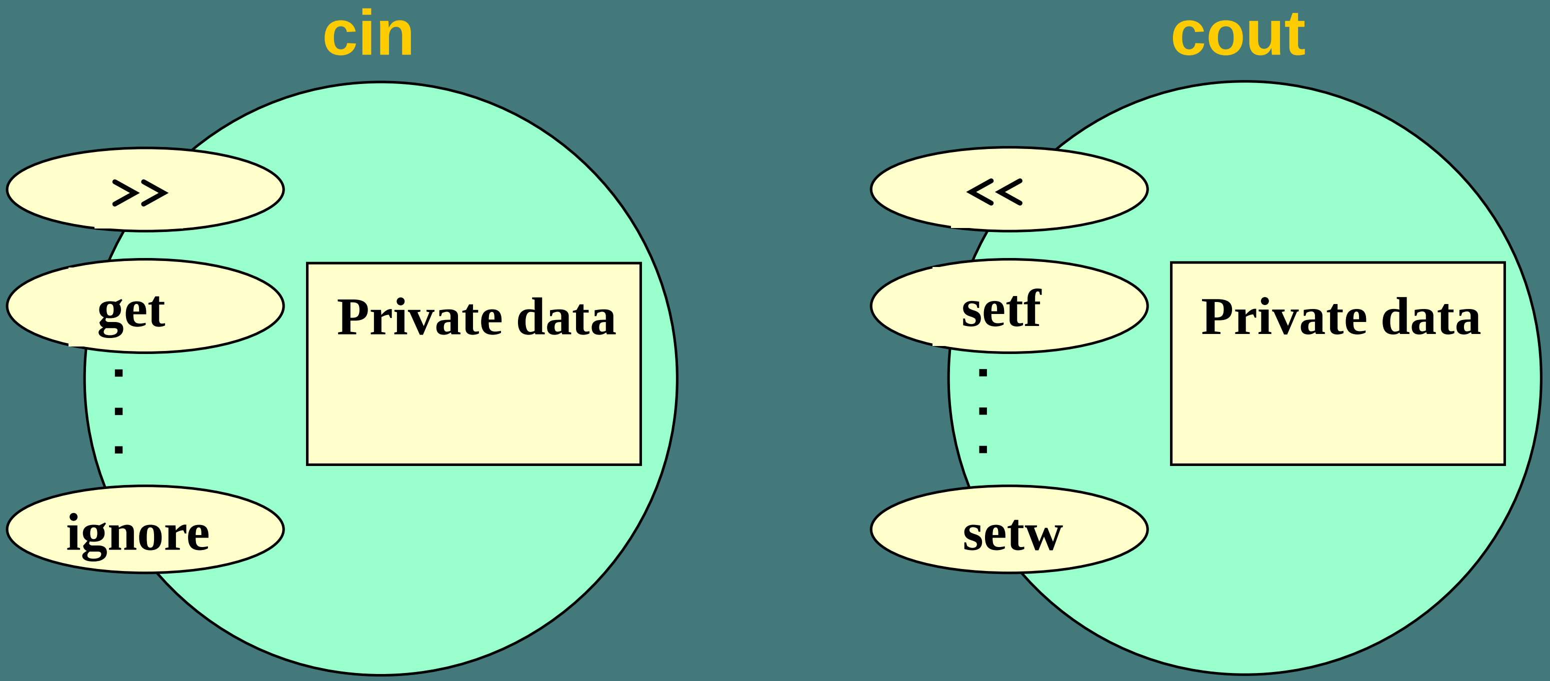
❖ A header file ,contains the declarations of 2 classes: istream, ostream



<iostream>

- C++ standard library declares :宣告兩物件

  istream cin ;   ostream cout ;



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;
}
```

End of line

Output of Program

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