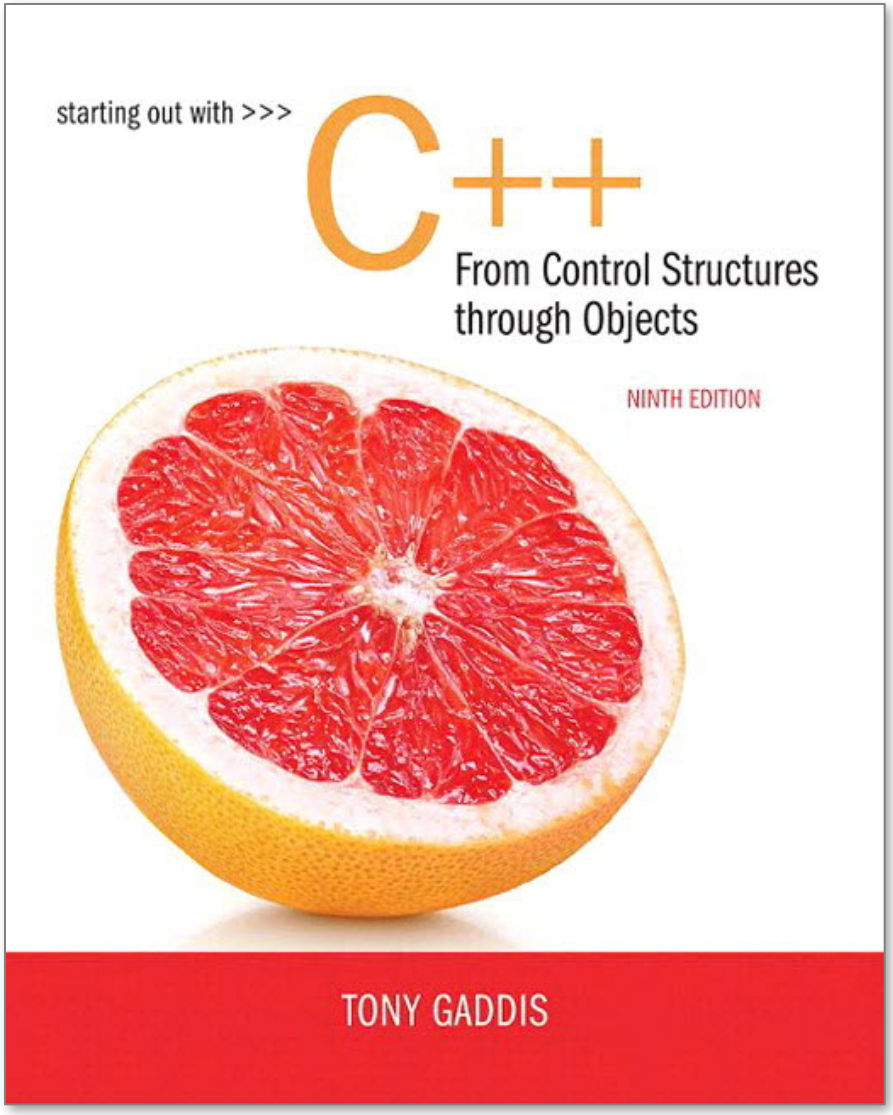
Chapter 2:

Introduction

to

C++



2.1

The Parts of a C++ Program

The Parts of a C++ Program

```
// sample C++ program
#include <iostream>
using namespace std;
int main()
{
    cout << "Hello, there!";
    return 0;
}
```

comment

preprocessor directive

which namespace to use

beginning of function named `main`

beginning of block for `main`

output statement

string literal

Send `0` to operating system

end of block for `main`

Special Characters

Character	Name	Meaning
//	Double slash	Beginning of a comment
#	Pound sign	Beginning of preprocessor directive
< >	Open/close brackets	Enclose filename in #include
( )	Open/close parentheses	Used when naming a function
{ }	Open/close brace	Encloses a group of statements
" "	Open/close quotation marks	Encloses string of characters
;	Semicolon	End of a programming statement

Productivity

Productivity = intensity of focus * time spent.

Focus is the secret of the relaxed high performer. When you work, work. When you play, play.

Replace re-reading with self-quizzing.

Practice.

- Especially in computer science courses
- Learning is like playing the piano: you must put the hours in.

2.2

The `cout` Object

The `cout` Object

- Displays output on the computer screen

- You use the stream insertion operator `<<` to send output to `cout`:

```
cout << "Programming is fun!";
```

The `cout` Object

- Can be used to send more than one item to cout:

```
cout << "Hello " << "there!";
```

Or:

```
cout << "Hello ";
cout << "there!";
```

The `cout` Object

- This produces one line of output:

```
cout << "Programming is ";
cout << "fun!";
```

The `endl` Manipulator

- You can use the **`endl`** manipulator to start a new line of output. This will produce two lines of output:

```
cout << "Programming is" << endl;
cout << "fun!";
```

The `endl` Manipulator

```
cout << "Programming is" << endl;
cout << "fun!";
```

The `endl` Manipulator

- You do NOT put quotation marks around **`endl`**

- The last character in **`endl`** is a lowercase L, not the number 1.

`endl` ← This is a lowercase L

The `\n` Escape Sequence

- You can also use the **`\n`** escape sequence to start a new line of output. This will produce two lines of output:

```
cout << "Programming is\n";
cout << "fun!";
```

Notice that the **`\n`** is INSIDE the string.

The `\n` Escape Sequence

```
cout << "Programming is\n";
cout << "fun!";
```

2.3

The `#include` Directive

The `#include` Directive

- Inserts the contents of another file into the program
- This is a preprocessor directive, not part of C++ language
- `#include` lines not seen by compiler
- Do <u>not</u> place a semicolon at end of `#include` line

2.4

- Variables and Literals

Variables and Literals

- Variable: a storage location in memory
  - Has a name and a type of data it can hold
  - Must be defined before it can be used:

```
int item;
```

Variable Definition in Program 2-7

Program 2-7

```
1   // This program has a variable.
2   #include <iostream>
3   using namespace std;
4
5   int main()
6   {
7     int number;
8
9     number = 5;
10    cout << "The value in number is " << number << endl;
11    return 0;
12  }
```

Variable Definition

Program Output

```
The value in number is 5
```

Literals

- <u>Literal</u>: a value that is written into a program's code.

  `"hello, there"` (string literal)

  `12` (integer literal)

Integer Literal in Program 2-9

Program 2-9

```
// This program has literals and a variable.
#include <iostream>
using namespace std;

int main()
{
   int apples;

   apples = 20;
   cout << "Today we sold " << apples << " bushels of apples.\n";
   return 0;
}
```

20 is an integer literal

Program Output

```
Today we sold 20 bushels of apples.
```

String Literals in Program 2-9

Program 2-9

```
1  // This program has literals and a variable.
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7     int apples;
8
9     apples = 20;
10    cout << "Today we sold " << apples << " bushels of apples.\n";
11    return 0;
12 }
```

These are string literals

Program Output

```
Today we sold 20 bushels of apples.
```

C++ Key Words

Table 2-4 The C++ Key Words

alignas	const	for	private	throw
alignof	constexpr	friend	protected	true
and	const_cast	goto	public	try
and_eq	continue	if	register	typedef
asm	decltype	inline	reinterpret_cast	typeid
auto	default	int	return	typename
bitand	delete	long	short	union
bitor	do	mutable	signed	unsigned
bool	double	namespace	sizeof	using
break	dynamic_cast	new	static	virtual
case	else	noexcept	static_assert	void
catch	enum	not	static_cast	volatile
char	explicit	not_eq	struct	wchar_t
char16_t	export	nullptr	switch	while
char32_t	extern	operator	template	xor
class	false	or	this	xor_eq
compl	float	or_eq	thread_local	

You cannot use any of the C++ key words as an identifier. These words have reserved meaning.

Variable Names

- A variable name should represent the purpose of the variable. For example:

```
itemsOrdered
```

The purpose of this variable is to hold the number of items ordered.

Identifier Rules

- The first character of an identifier must be an alphabetic character or and underscore (_),
- After the first character you may use alphabetic characters, numbers, or underscore characters.
- Upper- and lowercase characters are distinct

Valid and Invalid Identifiers

IDENTIFIER	VALID?	REASON IF INVALID
`totalSales`	Yes	
`total_Sales`	Yes	
`total.Sales`	No	Cannot contain `.`
`4thQtrSales`	No	Cannot begin with digit
`totalSale$`	No	Cannot contain `$`

2.6

- Integer Data Types

Integer Data Types

- Integer variables can hold whole numbers such as 12, 7, and -99.

Table 2-6 Integer Data Types

Data Type	Typical Size	Typical Range
`short int`	2 bytes	−32,768 to +32,767
`unsigned short int`	2 bytes	0 to +65,535
`int`	4 bytes	−2,147,483,648 to +2,147,483,647
`unsigned int`	4 bytes	0 to 4,294,967,295
`long int`	4 bytes	−2,147,483,648 to +2,147,483,647
`unsigned long int`	4 bytes	0 to 4,294,967,295
`long long int`	8 bytes	−9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`unsigned long long int`	8 bytes	0 to 18,446,744,073,709,551,615

2.7

The `char` Data Type

The `char` Data Type

- Used to hold characters or very small integer values
- Usually 1 byte of memory
- Numeric value of character from the character set is stored in memory:

CODE:

```
char letter;
letter = 'C';
```

MEMORY:

letter

67

Character Literals

- Character literals must be enclosed in single quote marks. Example:

```
'A'
```

Character Literals in Program 2-14

Program 2-14

```
// This program uses character literals.
#include <iostream>
using namespace std;

int main()
{
    char letter;

    letter = 'A';
    cout << letter << '\n';
    letter = 'B';
    cout << letter << '\n';
    return 0;
}
```

Program Output

```
A
B
```

2.8

The C++ `string` Class

The C++ `string` Class

- Special data type supports working with strings

```
#include <string>
```

- Can define `string` variables in programs:

```
string firstName, lastName;
```

- Can receive values with assignment operator:

```
firstName = "George";
lastName = "Washington";
```

- Can be displayed via `cout`

```
cout << firstName << " " << lastName;
```

The `string` class in Program 2-15

Program 2-15

```
// This program demonstrates the string class.
#include <iostream>
#include <string> // Required for the string class.
using namespace std;

int main()
{
   string movieTitle;

   movieTitle = "Wheels of Fury";
   cout << "My favorite movie is " << movieTitle << endl;
   return 0;
}
```

Program Output

```
My favorite movie is Wheels of Fury
```

Sample Practice Problems

- Using integer variables
  - Declare 5 integer variables
  - calculate their sum and store the sum in a new variable, total
  - display the total by using cout

- Using char variable
  - Declare two character type variables in a program and initialize their values with 'a' and 'b' respectively
  - Display the two characters by using cout

- Solving an expression
  - Declare three integer variables, a, b, and c. Initialize them with 5, 9, and 3 respectively
  - Now, calculate the value (b*b - 4 * a * c) / (2 * a) and store the result in another integer variable, exp
  - Display the the value of exp by using cout. (the result should be 2)