CSI33 Data Structures

Sharon Persinger

Fall 2019

Day 16 October 30

C++ development

- C was developed 1970's to be a cross-platform programming language, in contrast to assembly languages that are specific to a particular computer architecture.
- Developed by Kernigan and Ritchie at Bell Labs
- OOP developed in late 1970's and 1980's, and Bjorne Stroustrup of AT&T added OOP features to C to create C++
- C++ is largely backward compatible with C
- Both are lower-level languages than Python. Fewer built-in sophisticated data types.
- Many data structures and methods are available for C++ in Standard Template Library (STL).
- Terse and somewhat fussy syntax. && for and, etc.

Compiling C/C++

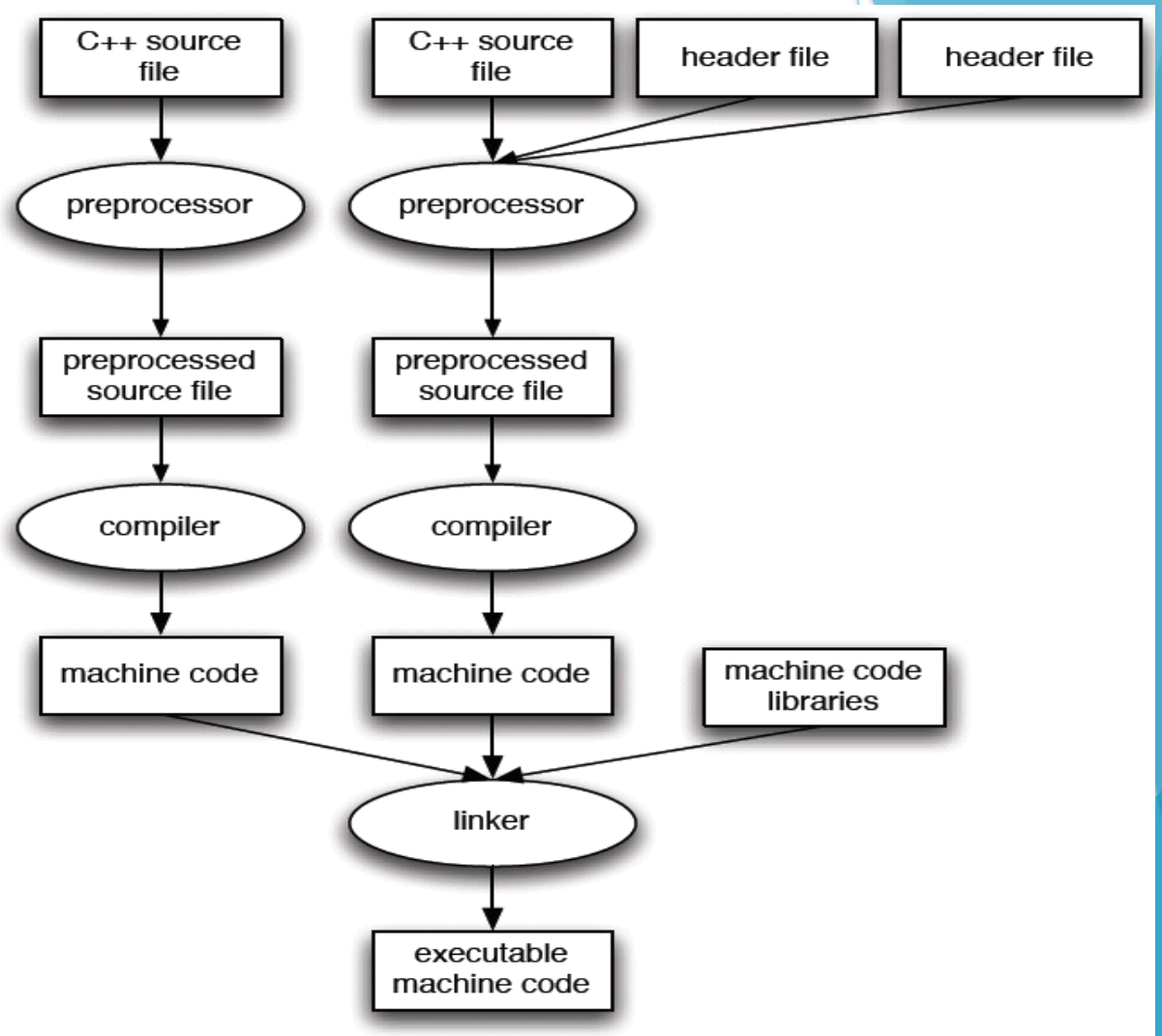

Figure 8.1: Compile-and-link process for C++ code

Hello World!

- Comments: // for one-line comment
- /* */ encloses multiline comments
- Blocks of code: enclosed in braces { } Python uses indentation. Indentation in C++ doesn't have syntactical meaning, but programmers use it to make code more readable
- C++ statements end with ; . The most common beginner error.
- C strings are enclosed in double quotes. " " Used commonly in C++, although there is a string class for an OOP approach.
- C characters are enclosed in single quotes 'x'. Special characters '\n' newline, '\t' tab
- Iostream – C++ OOP simple console input and output, cout for output
- Create some errors

More from Hello World

Include statement - copy the definitions from another file into your program .
Called header files
C+ has namespaces, important when using certain built-in classes. Usually use std namespace, using allows program to use cout instead of std::cout
Input/output classes are in iostream.

Keywords

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

Figure 8.3: C++ Keywords

C++ data types

- 5 basic data types
- int
- char
- float
- double
- bool

Details about C++ data types

Data type	Typical range of values	Typical # of bytes	Comments
`int`	-2,147,483,648 to 2,147,483,647	4	integer values only
`unsigned int`	0 to 4,294,967,295	4	integer values only
`short int`	-32,768 to 32,767	2	integer values only
`unsigned short int`	0 to 65,535	2	integer values only
`char`	-128 to 127	1	integer values only
`unsigned char`	0 to 255	1	integer values only
`float`	approximately $+/-10^{38}$	4	real numbers with 6 or 7 significant digits
`double`	approximately $+/-10^{308}$	8	real numbers with 15 or 16 significant digits
`bool`	`true` or `false`	1	`true` and `false` are constants

Figure 8.4: C++ built-in data types

C++ identifiers

C++ identifiers must start with letter or underscore and can contain letters, digits and underscores.
In C++, keywords cannot be used as identifiers.

C++ variables

- Must be declared with data type before use
- Why? C++ code is compiled to machine code. Actual mechanism of addition, for instance, depends on whether the data is integer or floating point. Compiler generates different instructions for the different types.
- (For Python the difference is handled at the interpreter level once the data type is determined.)

C++ variables

- A variable is a
- Name that
- Refers to a memory location
- That holds a value

Another example program

- ctof.cpp - converts Celsius to Fahrenheit
- Input with cin in skips whitespace. Convenient but cannot enter a space to be stored
- C++ main method returns an int value, return 0 means normal execution
- Function definition example

Example program character input

input2.cpp ctof.cpp
Input with cin in skips whitespace. Convenient but cannot enter a space to be stored

C++ operators and precedence

https://en.cppreference.com/w/cpp/language/operator_precedence

If
If/else

Variable scope