cs3157: c++ lecture #2 (mon-11-apr-2005)

• today:

- language basics: identifiers, data types, operators, type conversions, branching and looping, program structure
- data structures: arrays, structures
- pointers and references
- I/O: writing to the screen, reading from the keyboard, iostream library
- functions: defining, overloading, inlining, overriding
- classes: defining, scope, ctors and dtors
- listing of keywords

chronology of some programming languages...

- (1958) Algol created the first high-level structured language with a systematic syntax
- (1969) UNIX created using BCPL (Basic Combined Programming Language)
- (1969) B created by Ken Thompson, as a replacement for BCPL
- (1970) Pascal established as the successor to Algol
- (1973) C completed, and released as the successor to B, giving the user control of data types
- (1979) Bjarne Stroustrup begins work on C-with-Classes, an Object Oriented version of C
- (1983) C-with-Classes redesigned and released as C++
- (1985) First mass release of C++ compilers

cs3157-spring2005-sklar-cpp

cs3157-spring2005-sklar-cpp

C++ vs Java

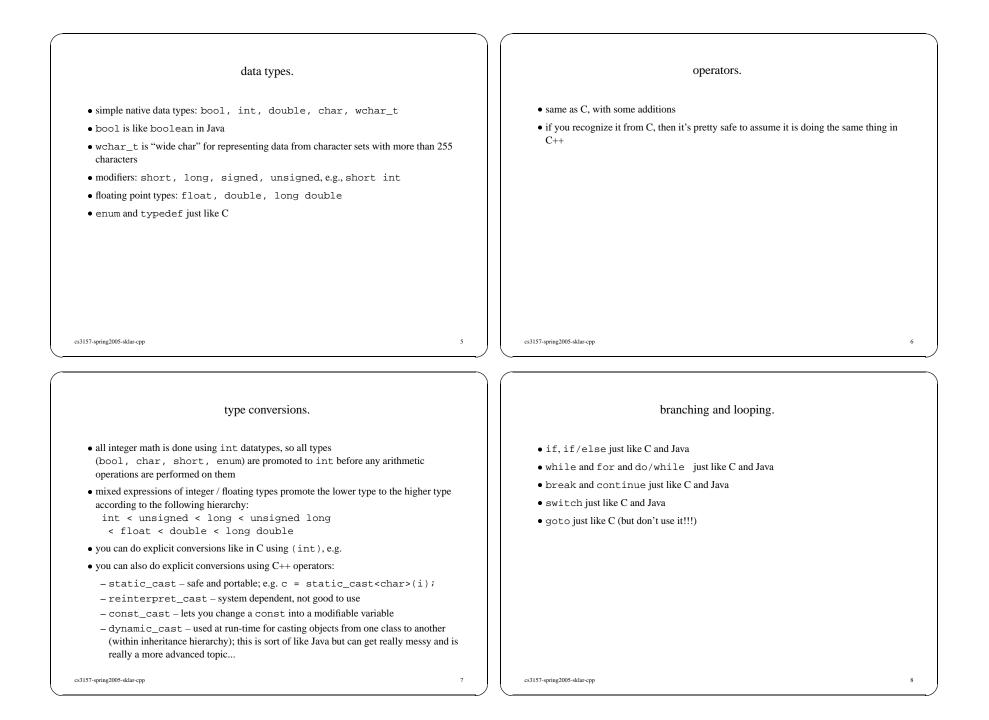
- advantages of C++ over Java:
 - C++ is very powerful
 - C++ is very fast
 - C++ is much more efficient in terms of memory
 - compiled directly for specific machines (instead of bytecode layer, which could also be seen as a portability advantage of Java over C++...)
- disadvantages of C++ over Java:
 - Java protects you from making mistakes that C/C++ don't, as you've learned now from working with C
 - C++ has many concepts and possibilities so it has a steep learning curve
 - extensive use of operator overloading, function overloading and virtual functions can very quickly make C++ programs very complicated
 - shortcuts offered in C++ can often make it completely unreadable, just like in C

identifi ers.

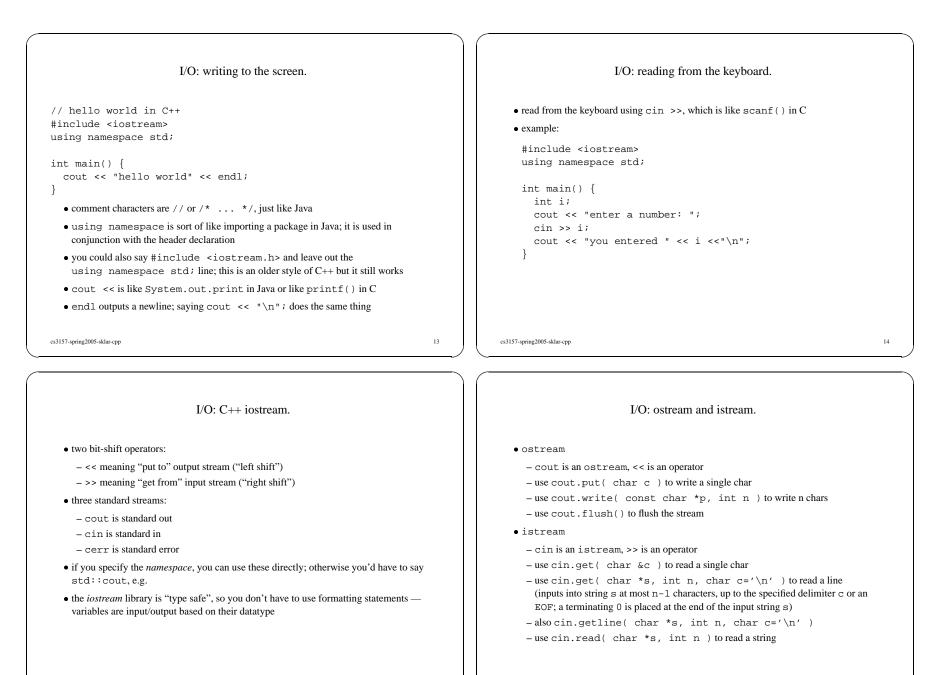
- i.e., valid names for variables, methods, classes, etc
- just like C:
 - names consist of letters, digits and underscores
 - names cannot begin with a digit
 - names cannot be a C++ keyword
- literals are just like in C with a few extras:
 - numbers, e.g.: 5, 5u, 5L, 0x5, true
 - characters, e.g., 'A'
 - strings, e.g., "you" which is stored in 4 bytes as 'y', 'o', 'u', '0'

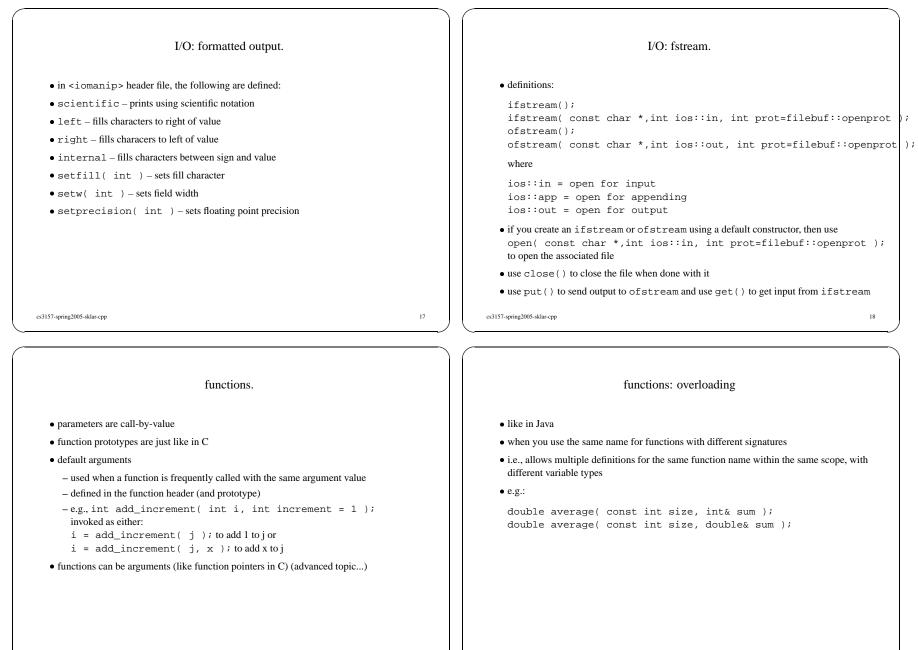
1

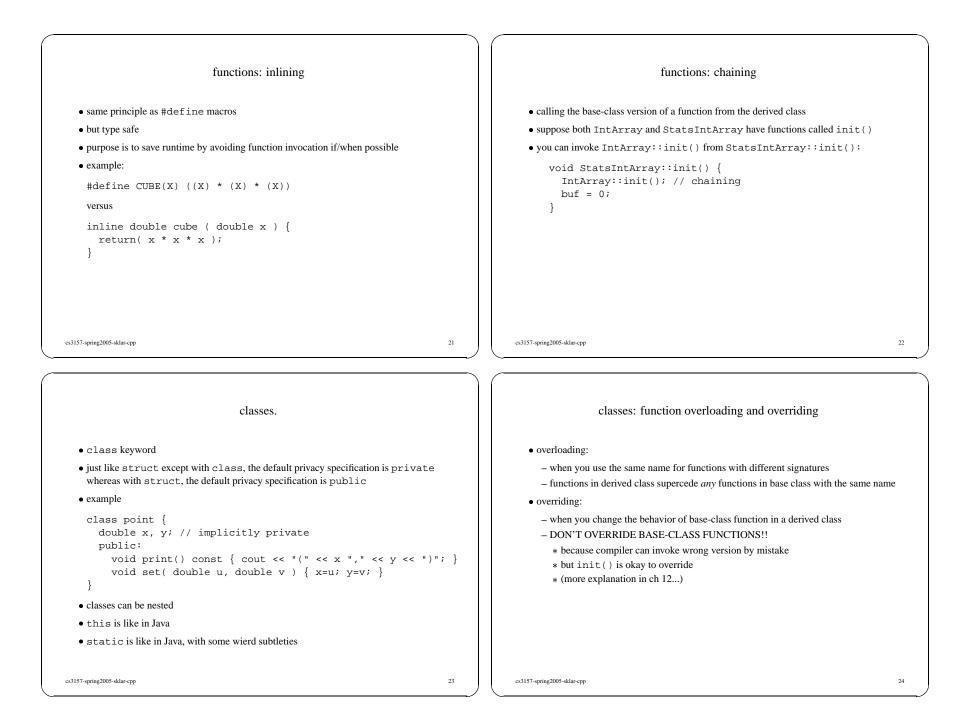
2

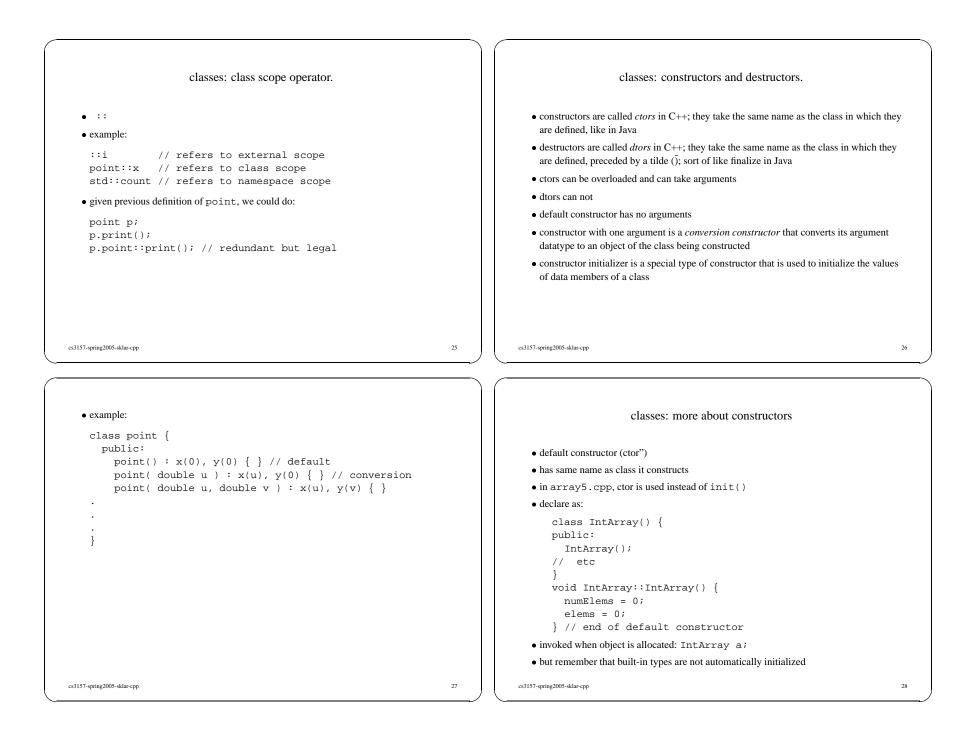


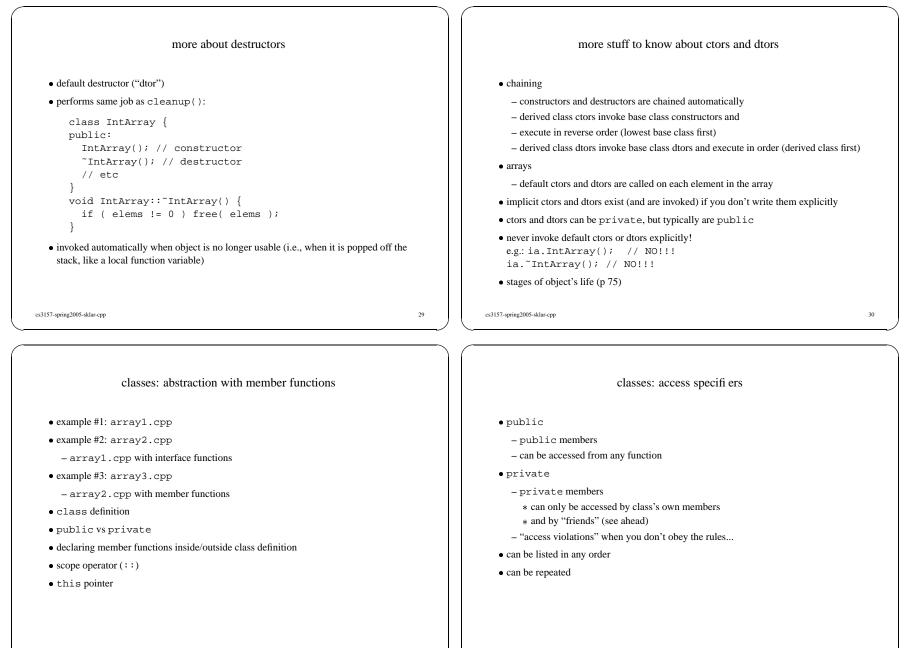
program structure.	arrays.		
• just like in C	• similar to C		
 program is a collection of functions and declarations language is block-structured declarations are made at the beginning of a block; allocated on entry to the block and freed when exiting the block parameters are call-by-value unless otherwise specified 	<pre>• similar to C • dynamic memory allocation handled using new and delete instead of malloc (and family) and free • examples: int a[5]; char b[3] = { 'a', 'b', 'c' }; double c[4][5]; int *p = new int(5); // space allocated and *p set to 5 int **q = new int[10]; // space allocated and q = &q[0] int *r = new int; // space allocated but not initialize </pre>		
3157-spring2005-sklar-cpp 9	cs3157-spring2005-sklar-cpp 10		
structures.	pointers and references.		
• struct keyword like in C (but you don't need typedef)	pointers and references. <i>pointers</i> are like C: 		
• struct keyword like in C (but you don't need typedef)	 <i>pointers</i> are like C: int *p means "pointer to int" p = &i means p gets the address of object i 		
 struct keyword like in C (but you don't need typedef) use dot operator or -> to access members (fields) of a struct or struct * 	 <i>pointers</i> are like C: – int *p means "pointer to int" 		





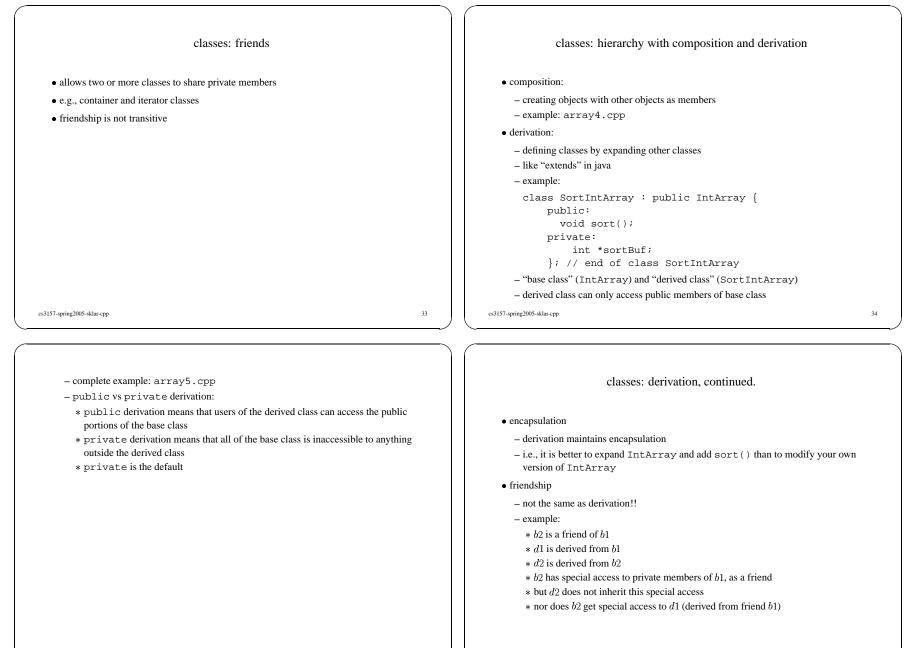


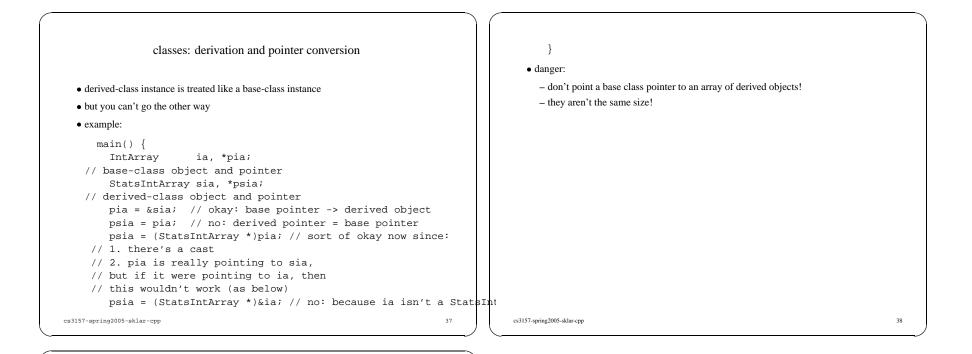




cs3157-spring2005-sklar-cpp

31





C^{++}	keywords.

asm	else	new	this
auto	enum	operator	throw
bool	explicit	private	true
break	export	protected	try
case	extern	public	typedef
catch	false	register	typeid
char	float	reinterpret_cast	typename
class	for	return	union
const	friend	short	unsigned
const_cast	goto	signed	using
continue	if	sizeof	virtual
default	inline	static	void
delete	int	static_cast	volatile
do	long	struct	wchar_t
double	mutable	switch	while
dynamic_cast	namespace	template	

cs3157-spring2005-sklar-cpp

39