

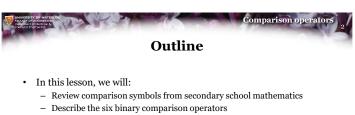


- Previously, we saw that the literals true and false actually evaluate to the values 1 and 0, respectively
- We will now look at six comparison operators that compare integers and floating-point numbers
 - From your secondary school mathematics, given two integers or real numbers, you can always compare their values

$$= \neq < \leq > \geq$$

For example,

$$1.\overline{9} = 2 \qquad \sin(x) \le 1 \qquad \pi \ne \frac{22}{7} \\ x^2 \ge 0 \qquad \pi^e < e^{\pi} \qquad \frac{10(e^{\pi} - \ln(3))}{\ln(2)} < 318$$



- Understand how they differ in purpose
 They evaluate to true or false (1 or 0)
- Look at some common errors
- Upcasting of the operands

000



 In your mathematics courses, you used comparison operators as statements:

If x > 2, then $x^2 > 4$.

- In general, you made statements like "x < y is false."
 Instead, you would now write x ≮ y or x ≥ y
- You may have even defined the absolute-value as:

$ x = \langle$	x	$x \ge 0$
$ \lambda = 1$	$\left -x\right $	x < 0

000

0000

ece150

1



Comparison Operator	Description
==	equal to
! =	not equal to
<	less than
<=	less than or equal to
>=	greater than or equal to
>	greater than

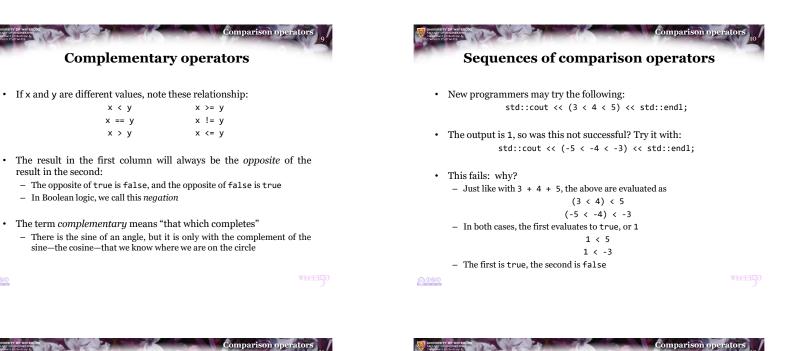
Remember, we are restricted to symbols on the keyboard
 In C++, enunciate '!' as "not" when you say it or think it

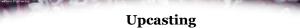
000

 The = operator is res 	it as you say it:
Another very commo mean <= or >=, respec – Remember to write i	on mistake is to write =< or => when you tively it as you say it:
mean <= or >=, respec – Remember to write i	tively it as you say it:
or 3 <= 4	3 is less than or equal to 4
or 3 <= 4	3 < = 4

Comparison P	rry of writing of the second sec
and a	Comparison operators
•	Each takes two operands, and evaluates to either true or false depending on whether or not the operands satisfy the condition
	<pre>std::cout << (3 < 4) << std::endl; // prints 1 std::cout << (4 < 4) << std::endl; // prints 0 std::cout << (3 != 4) << std::endl; // prints 1 std::cout << (3.1 != 3.100) << std::endl; // prints 0 std::cout << (-42.3 == -42.3) << std::endl; // prints 1 std::cout << (-7 == 7) << std::endl; // prints 0</pre>
<u>@@@@</u>	ΨΕκ
UNIVERSI FACULTY O	rromanike
UNIVERSE PARAMETER Converse P	Comparison operator Comparison operator Common mistakes
Marry esta Source of Source of	
UNIVERSI PACINE O PARAGE	Common mistakes Consider these two mistakes:
ANNUESS ANNUES	Consider these two mistakes: #include <iostream></iostream>
Participant State	Consider these two mistakes: #include <iostream> int main(); int main() { std::cout << (3 =< 4) << std::endl; std::cout << (3 => 4) << std::endl; return 0; Write it as you say it: 3 is less than or equal to 4</iostream>
•	Consider these two mistakes: #include <iostream> int main(); int main() { std::cout << (3 =< 4) << std::endl; std::cout << (3 => 4) << std::endl;</iostream>
∙	Consider these two mistakes: #include <iostream> int main(); int main() { std::cout << (3 =< 4) << std::endl; std::cout << (3 => 4) << std::endl; return 0; Write it as you say it: 3 is less than or equal to 4 } The error messages can be a little opaque, but you can see the issue: example.cpp: In function 'int main()': example.cpp: 6:22: error: expected primary-expression before '<' token</iostream>

2





• If one operand is an integer and the other is a floating-point number, the integer is converted to a floating-point number first:

- Remember that integers are stored differently from floating-point numbers—more on this later
- All integers between -2⁵³ + 1 and 2⁵³ 1 can be perfectly represented using a double-precision floating-point number (double)
 - The value 2^{53} is approximately 9 quadrillion

7ECE150

0000

Summary

Understand the six comparison operators: == != < <= >
 Based on the operands, these evaluate to either true or false (1 or 0)

- If one operand is an integer and the other is a floating-point number,

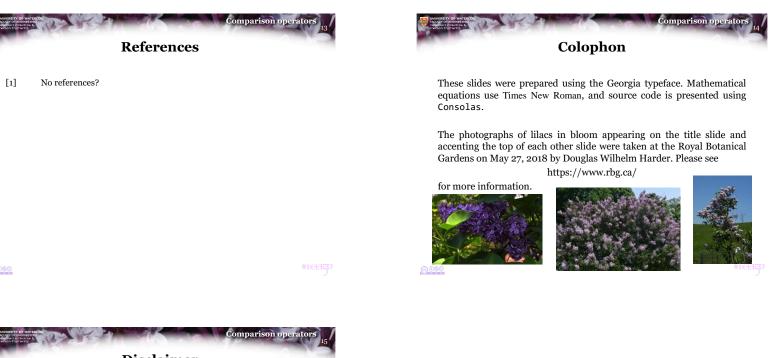
- You must avoid using = when you want to compare the operands

- You cannot use =< or => when you mean <= or >=

the integer is cast as a floating-point number

· Following this lesson, you now:





Disclaimer

These slides are provided for the ECE 150 *Fundamentals of Programming* course taught at the University of Waterloo. The material in it reflects the authors' best judgment in light of the information available to them at the time of preparation. Any reliance on these course slides by any party for any other purpose are the responsibility of such parties. The authors accept no responsibility for damages, if any, suffered by any party as a result of decisions made or actions based on these course slides for any other purpose than that for which it was intended.