HIGH-LEVEL PROGRAMMING I

Relational & Logical Operators by Prasanna Ghali

Relational Operators

Relational operators allow you to make comparisons

these 4 operators have similar precedence but *lower* than arithmetic operators

these 2 operators have similar

precedence but lower than 4

operators listed above

Operator		Description	
-	<	Less than	
	>	Greater than	
	<=	Less than or equal to	
_	>=	Greater than or equal to	
	==	Equal to	
	! =	Not equal to	

Left to right associative order

Relational Expressions (1/2)

□ Relational expression evaluates to value 0 if it is false, or 1 if it is true

Expression	Meaning	Value
8 < 15	8 is less than 15	1
6 != 6	6 is not equal to 6	0
2.5 > 5.8	2.5 is greater than 5.8	0
5.9 <= 7.5	5.9 is less than or equal to 7.5	1
'A' >= 'Z'	ASCII value of 'A' is greater than or equal to 'Z'	0

Relational Expressions (2/2)

□ Given int a = 3, b = 8, c = 1; what is value of following expressions?

Expression	Meaning	Value
a==b==c		
a>=b!=b<=a		
a < 10 < b		
(a < 10) != (10 < b)		

Boolean Values in C89

- Older C standards did not have Boolean type
- Hacks involving preprocessor were required to simulate Boolean type and values

```
#define TRUE (1)
#define FALSE (0)
#define BOOL int
```

```
if (flag == TRUE) {
    // do this thing ...
} else {
    // do other thing ...
}
```

Boolean Values in C99 and C11 (1/2)

- □ Boolean type introduced in C99: _Bool
- C++ provides Boolean type bool and keywords true and false to indicate values
 1 and 0, respectively
- C99 provides new header file <stdbool.h> that supplies macros bool, true, and false
- This is handy and we'll use these macros often through the rest of this semester

Boolean Values in C99 and C11 (2/2)

```
int year = 2021;
// does year represent centennial?
bool flag = year % 100 == 0; // flag is false
printf("flag: %d\n", flag); // prints 0 to stdout
```

Compound Conditions

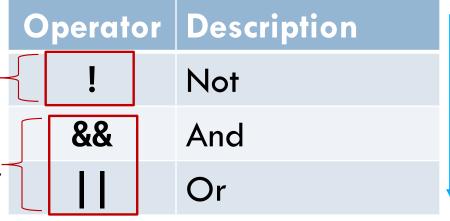
- Often, you need to evaluate more than one expression to determine whether an action should take place
- If you worked more than 40 hours and you're not a manager, you get overtime pay
- If you make less than \$100,000, you pay 10% tax; more than \$100,000 but less than \$1,000,000, you pay 15% tax
- If you were speeding and your speed is less than twice posted speed limit, you get \$100 ticket; otherwise you get \$500 ticket

Logical Operators

Logical operators allow you to combine expressions

Unary operators have higher precedence than binary operators

These 2 operators have *lower* precedence than relational operators which in turn have lower precedence than arithmetic operators



precedence order

Understanding AND Logic (1/3)

- □ Basic monthly cell phone service bill is \$40
- Additional \$20 billed to customers making more than 100 calls that last for total of more than 500 minutes
- Require ability to write following expression required:
 - calls_made > CALLS) AND (call_minutes > MINUTES)
- □ This is done in C/C++ as:
 - calls_made > CALLS) && (call_minutes > MINUTES)

Understanding AND Logic (2/3)

All listed conditions must be met for resulting action to take place!!!

X	у	x && y	Value
true	true	true	1
true	false	false	0
false	true	false	0
false	false	false	0

Understanding AND Logic (3/3)

- Each part of logical expression is evaluated only as far as necessary to determine whether entire expression is true or false
 - Referred to as short-circuit evaluation
- Additional \$20 billed to customers making more than 100 calls that last for total of more than 500 minutes
 - calls_made > CALLS) && (call_minutes > MINUTES)

Not evaluated if customer has not made more than 100 calls

Understanding OR Logic (1/3)

- □ Basic monthly cell phone service bill is \$40
- Additional \$20 billed to customers making more than
 100 calls or send more than 200 text messages
- Require ability to write following expression required:
 - calls_made > CALLS) OR (texts_sent > TEXTS)
- \Box This is done in C/C++ as:
 - calls_made > CALLS) | (texts_sent > TEXTS)

Understanding OR Logic (2/3)

Only one of the listed conditions must be met for resulting action to take place!!!

X	у	x y	Value
true	true	true	1
true	false	true	1
false	true	true	1
false	false	false	0

Understanding OR Logic (3/3)

- Short-circuit evaluation implemented here too
 - Each part of expression is evaluated only as far as necessary to determine whether entire expression is true or false
- Additional \$20 billed to customers making more than 100 calls or send more than 200 text messages
 - calls_made > CALLS) || (texts_sent > TEXTS)

Not evaluated if customer has made more than 100 calls

Understanding NOT Logic (1/2)

- Reverses meaning of logical expression
- \square !(a == b) is same as (a != b)

X	!x	Value
true	false	0
false	true	1

high to low precedence order

Precedence Table: Arithmetic, Relational, and Logical Operators

Operator(s)	Description
! + -	Logical negation, unary plus and minus
* / %	Multiplication, division, remainder
+ -	Addition, subtraction
< <= > >=	Less than, less than or equal to, greater than, greater than or equal to
== !=	Equal to, not equal to
&&	Logical AND
	Logical OR

Relation and Logical Expressions (1/2)

Expression	Value
!('A' > 'B')	
!(6 <= 7)	
!7 > !6	
14 >= 5 && 'A' < 'B'	
24 >= 35 && 'A' < 'B'	
14 >= 5 'A' > 'B'	
24 >= (35 'A') > 'B'	
'A' <= 'a' 7 != 7	

Relation and Logical Expressions (2/2)

□ Given int a = 3, b = 8; what is value of following expressions

Expression	Meaning	Value
!a > b		
!(a > b)		
a < 10 < b		
a < 10 && 10 < b		
!a != !b		