Students:
Section 3.5 is a part of 2 assignments: CSC108 CH03 1-3-10 C3A

Instructor: CA
Due: 03/26/2025, 11:59 PM EST

Please do not complete before the due date. Assignment but will not count towards this assignment (unless the due date is changed). See syllabus for more info.

3.5 Detecting ranges using logical operators

Logical AND, OR, and NOT (general)

A **logical operator** treats operands as being true or false, and evaluates to true or false. Logical operators include AND, OR, and NOT. Programming language operators typically use extra symbols for logical operators, but before the words AND, OR, and NOT are used for programming languages.

INSTRUCTION ACTIVITY 3.5.1: Logical operators: AND, OR, and NOT.

Start In speed

x	y	x AND y
false	false	false
false	true	false
true	false	false
true	true	true

y	z	y OR z
false	false	false
false	true	true
true	false	true
true	true	true

z	NOT z
false	true
true	false

z = 7 y = 9

(x > 5) AND (y < 10) false
(x > 5) AND (y > 7) true
(x < 10) AND (y < 7) false
(x < 10) OR (y < 7) true
(x > 5) OR (y > 7) true
NOT (x < 5) true
NOT (x > 7) false

Captions ^

1. AND evaluates to true only if BOTH operands are true
2. OR evaluates to true if AND operand is true (one, the other, or both)
3. NOT evaluates to the opposite of the operand
4. Each operand is commonly an expression itself. If x = 7, y = 9, then (x > 5) AND (y < 10) is true AND true, so evaluates to true (both operands are true).

Feedback?

Table 3.5.1: Logical operators.

Logical operator	Description
x AND b	**Logical AND**: true when both of its operands are true.
x OR b	**Logical OR**: true when at least one of its two operands are true.
NOT a	**Logical NOT**: true when its one operand is false, and vice-versa.

Feedback?

INSTRUCTION ACTIVITY 3.5.2: Evaluating expressions with logical operators.

Indicate whether the expression evaluates to true or false:
x = 7, y = 9

1) x > 5
- True
- False

2) (x < 5) AND (y < 20)
- True
- False

3) (x < 10) AND (y > 20)
- True
- False

4) (x > 10) OR (y > 20)
- True
- False

5) (x > 10) OR (y < 20)
- True
- False

6) NOT (x > 5)
- True
- False

7) NOT ((x < 5) AND (y > 20))
- True
- False

Feedback?

Detecting ranges with logical operators (general)

A common use of logical operators is to detect if a value is within a range.

INSTRUCTION ACTIVITY 3.5.3: Using AND to detect if a value is within a range.

Start In speed

10 x = 15

10 15 16 17 18 19

10 < x AND x < 15

Captions ^

1. The range 10 < x < 15 means that x may be 11, 12, 13, 14.
2. Specifying that range in a program can be done using two + operators along with an AND operator. 10 < x defines the range 11 and higher.
3. x < 15 defines the range 14 and lower. AND'ing yields the overlapping range. Only when x is 11, 12, 13, or 14 will both expressions be true.

Feedback?

INSTRUCTION ACTIVITY 3.5.4: Using AND to detect if a value is within a range.

1) Which approach uses a logical operator to detect if x is in the range 1 to 10?
- (x > 1) AND (x < 10)
- (1 < x) AND (x < 10)
- (1 < x) AND (x > 10)

2) Which detects if x is in the range -4 to +4?
- (x < 3) AND (x > 3)
- (x < 3) OR (x > 3)
- (x > 3) AND (x < 3)

3) Which detects if x is either less than -5, or greater than 10?
- (x < -5) AND (x > 10)
- (x < -5) OR (x > 10)

Feedback?

Logical operators

Special symbols are used to represent the AND, OR, and NOT logical operators. Logical operators are commonly used in expressions of if-else statements.

Table 3.5.2: Logical operators.

Logical operator	Description
a && b	**Logical AND**: (&&) true when both of its operands are true
a || b	**Logical OR**: (||) true when at least one of its two operands are true
!a	**Logical NOT**: (!) true when its one operand is false, and vice-versa.

Feedback?

INSTRUCTION ACTIVITY 3.5.5: Logical operators.

Match the symbol with the logical operator.

Hint: To use this tool

&&	AND
||	OR
!	NOT
	No such operator

Reset

Feedback?

INSTRUCTION ACTIVITY 3.5.6: Logical operators: Complete the expressions to detect the desired range.

1) daysLogged is greater than 30 and less than 90

```
if (daysLogged > 30):        :daysLogged < 90) {
}
```

Check Show answer

2) 0 < maxCars < 100

```
if (maxCars > 0):        :maxCars < 100) {
}
```

Check Show answer

3) numStones is between 10 and 20, inclusive.

```
if (:numStones >= 10):        :numStones <= 20) {
}
```

Check Show answer

4) myValue is either less than 15, or greater than 75

```
if (:myValue < 15):        :myValue > 75) {
}
```

Check Show answer

Feedback?

INSTRUCTION ACTIVITY 3.5.7: Creating expressions with logical operators.

1) numDogs has a minimum of 2 and a maximum of 5.

```
if (:numDogs >= 2):        :) {
}
```

Check Show answer

2) wage is greater than 15 and less than 18. Use > and < (not >= and <=). Use parentheses around sub-expressions.

```
if (:                    :) {
}
```

Check Show answer

3) num is a 3-digit positive integer. Ex: 100, 989, and 523, are 3-digit positive integers, but 99, 1000, and -4 are not.

For most direct readability, your expression should compare directly with the smallest and largest 3-digit number.

```
if (:num >= 100):                    :) {
}
```

Check Show answer

Feedback?

INSTRUCTION ACTIVITY 3.5.8: Logical expression simulator.

Try trying different expressions and/or values to determine whether the expression evaluates to true for different values of x, y.
Ex: Test the expression (x > 10) && (y > 3) when x is assigned with 1, 2, 5, and 20.

```
int x = 1;
int y = 2;
int z = 3;
```

Run code

Output to:
Waiting your code...

Feedback?

Example: TV channels

A cable TV provider may have regular channels numbered 2-499, and high-definition channels numbered 1002-1499. A program may set a character variable to 'r' for standard, 'h' for high-definition, and 'e' for error.

Figure 3.5.1: Detecting ranges: Cable TV channels.

```
if (channelType == 'r' && (:userChannel >= 2 &&
    :userChannel <= 499)):
    channelStatus = 'r';
else if (:channelType == 'h' && (:userChannel >= 1002 &&
    :userChannel <= 1499)):
    channelStatus = 'h';
else:
    channelStatus = 'e';
```

Feedback?

zyDE 3.5.1: Detecting ranges: Cable TV channels.

Run the program and observe the output. Change the input time value from 3 to another number, and run again.

Load default template...

```
#include <iostream>
using namespace std;

int main() {
    int userChannel;
    char channelType;
    char channelStatus;

    cin >> userChannel;

    if (:channelType == 'r' && (:userChannel >= 2 &&
        :userChannel <= 499)):
        channelStatus = 'r';
    else if (:channelType == 'h' && (:userChannel >= 1002 &&
        :userChannel <= 1499)):
        channelStatus = 'h';
    else:
        channelStatus = 'e';
```

Run

Feedback?

INSTRUCTION ACTIVITY 3.5.9: TV channel example: Detecting ranges

Consider the above example.

1) If userChannel is 300, is what does the if statement's expression, (:userChannel >= 2) && (:userChannel <= 499), evaluate?
- True
- False

2) If userChannel is 300, does the else if's expression (:userChannel >= 1002 && (:userChannel <= 1499) get checked?
- Yes
- No

3) Did the expressions use logical AND or logical OR?
- AND
- OR

4) Channels 500-999 are past channels. Does this expression detect that range? (:userChannel >= 500) || (:userChannel <= 1499)
- Yes
- No

Feedback?

Detecting ranges implicitly vs. explicitly

A programmer often uses logical operators to detect a range by explicitly specifying the high-end and low-end of the range. However, if a program should detect increasing ranges without gaps, a multi-branch if-else statement can be used without logical operators, the low-end of the range is implicitly known upon reaching an expression. Likewise, a decreasing range without gaps has implicitly known high-ends.

INSTRUCTION ACTIVITY 3.5.10: Detecting ranges implicitly vs. explicitly.

Start In speed

```
if (x < 0):
    x = 0;
else if (x < 5):
    x = x + 5;
else if (x < 10):
    x = x + 10;
else:
    x = 10;
```

Explicitly defined ranges

```
if (x < 0):
    x = 0;
else:
    x = x + 5;
else if (x < 10):
    x = x + 10;
else:
    x = 10;
```

Implicitly defined ranges

Captions ^

1. This code detects ranges explicitly using the AND operator. The first branch executes when x < 0, the second when (x >=0) && (x < 5).
2. But, if the first branch doesn't execute, x must be >= 0. So the second branch's expression can just be x <= 5. The x >= 0 is implied.
3. Implicit ranges can simplify a multi-branch if statement for ranges without gaps.

Feedback?

INSTRUCTION ACTIVITY 3.5.11: Detecting ranges implicitly vs explicitly.

For each pair of statements, does the second if-else statement detect the same ranges as the first if-else statement?

1)

```
if (temp >= 0):
  x = (temp / 2) + 10;
else if (temp < 20):
  x = (temp / 2) + 10;

if (temp < 0):
  x = (temp / 2) + 10;
else if (temp < 20):
  x = (temp / 2) + 10;
```

- Yes
- No

2)

```
if (systolic < 120):
  x = 0;
else if (systolic >= 120):
  x = 1;

if (systolic < 120):
  x = 0;
else if (systolic >= 120):
  x = 1;
```

- Yes
- No

3)

```
if (year >= 1901) && (year <= 2000):
  x = 1;
else if (year >= 2001) && (year <= 2100):
  x = 2;

if (year >= 1901):
  x = 1;
else if (year <= 2100):
  x = 2;
```

- Yes
- No

Feedback?

INSTRUCTION ACTIVITY 3.5.1: Detecting ranges using logical operators.

Start

Modify the given if statement so that "Not currently a teenager" is outputted if age is outside the range 13 - 18 inclusive. Otherwise, "Currently a teenager" is outputted.

Click here for examples

```
#include <iostream>
using namespace std;

int main() {
  int age;

  int i = age%10;

  // Details for declaring item
  if (age >= 13 && age <= 18) {
    cout << "Currently a teenager" << endl;
  }
  else {
    cout << "Currently a teenager" << endl;
  }

  return 0;
}
```

1 2 3 4

Check Next level

Feedback?

How was this section?

Provide section feedback

Activity summary for assignment: CSC108 CH03 1-3-10 C3A

Due: 03/26/2025, 11:59 PM EST
This assignment's due date has passed. Activity will still be recorded, but will not count towards this assignment (unless the due date is changed). See syllabus for more info.

Completion details

48 / 48 points
48 / 48 points submitted to Blackboard.com