Logical Operators and if/else statement

If Statement

- We may want to execute some code if an expression is `true`, and execute *some other code* when the expression is `false`.
- This can be done with two if statements…

```
 if (value >= LIMIT)

{
     // do something
}
 if (value < LIMIT)
{
     // do something else
}
```

If/Else (4.3)

- C++ provides a shortcut to combine two **`if`** statements:
- The statements in the else clause are executed only when the expression is false.

```
if (expression)
{
   // do stuff
}
else
{
   // do other stuff
}
```

Example

```
int number;
cout << "Enter a number, I'll tell you";
cout << " if it is odd or even: ";
cin >> number;

// use an if/else statement here
```

If/Else: Commenting

```
// the expression I'm using here
// checks for . . .
// so that I can ...
if (expression)
{

}
else
{

}
```

if/else/if statements (4.4)

- What if there are more than two alternatives?

```
if (RAINY == currentWeather)
{
   cout << "I need a rain jacket";
}
else if (SUNNY == currentWeather)
{
   cout << "I need a my shades";
}
else
{
   cout << "I better look outside!";
}
```

if/else/if statements (4.4)

- What if there are more than two alternatives?

```
if (RAINY == currentWeather)
{
   cout << "I need a rain jacket";
}
else if (SUNNY == currentWeather)
```

Why is `SUNNY == currentWeather`
better than `currentWeather == SUNNY` ?

```
else
{
   cout << "I better look outside!";
}
```

Logical Operators (4.7)

- There are three logical operators

&&	And
||	Or
!	Not

Precedence

Precedence Operators (Highest to Lowest)

`- (negation)      ! (Logical NOT)`
`* / %`
` -  +`

`<= => > <`
` == !=`

`&&` `||`
`= += -= *= /= %=`

Evaluating Expressions: And &&

- **`expr1 && expr2`**
- For the complete expression to be true, both expr1 and expr2 have to be true
- Example:

```
temp > HOT && humidity > STICKY
```

  - These are unbearable heat and humidity conditions
  - Both must be true for the entire expression to be true

Evaluating Expressions: Or ||

- **`expr1 || expr2`**
- The complete expression is true if either expr1 or expr2 is true
- Examples:

```
salary < MIN_SALARY || MARRIED == status
```

  - To qualify for financial aid, salary has to be less than some minimum salary or you must be married
  - Only one condition has to be true

Evaluating Expressions: Not !

- **`!expr`**
- Unary operator: Negation
- Examples:

```
!(salary < MIN_SALARY)
```

  - What makes this true? False?

Expression Evaluation

- According to the operator precedence and associativity rules given on the previous slide, how will the following expressions be evaluated?

```
x < min + max

min <= x && x <= max

!x == y + 2

x = a + b % 7 * 2
```

Practice

- Are these two code snippets equivalent?

```
int x, y;
if(x > y)
{
      x += y;
}
if(y < x)
{
      y += x;
}
```

```
int x, y;
if(x > y)
{
      x += y;
}
else
{
      y += x;
}
```

Problem

- Write a C++ program segment that allows the user the ability to input an integer from the keyboard.
- If the integer is positive, increment a variable **posCount** by 1. If the integer is **negative**, increment a variable **negCount** by 1. If neither, increment **zeroCount** by 1

```
int posCount=0, negCount=0, zeroCount=0;
```

Problem

- Write a program that displays a letter grade corresponding to an exam score

90 - 100	A
80 - 89	B
70 - 79	C
60 - 69	D
0 - 59	F

```
double examGrade;
cin >> exanGrade;
```

Nested if Statements (4.6)

- Note the indentation of the inner if

```
if (actual > expected)
{
   if (MAX == actual)
   {
   }
   else
   {
   }
}
else
{
}
```

Example

- Write nested if statements that set the correct value in the **`wage`** variable:

  If your status is full time, and you worked more than 10 years, your wage is $25. All other full time workers have a wage of $15. If your status is part time, you have a wage of $10.

```
const int FULLTIME=0, PARTTIME=1;
double wage;
int yearsWorked, status;
```

Example

- Your local bookstore has asked you to write a program to help them determine the cost of shipping of customers orders. If the order is $30 or less then shipping will cost $5, if the order is over $30 then shipping will be $3

Problem

- The bookstore has now changed it’s shipping policy so that
  - If the order is $30 or less, shipping is $5
  - If the order is over $30 but less than $50, shipping is $3
  - If the order is over $50 then shipping is $2

`exit()`

- To terminate a program we can use the **`exit(int status)`** function
  - This is a *function*, not part of the language
    - **`#include<stdlib.h>`**
  - The **`status`** is returned to the operating system to denote program success or failure
    - Success: 0
    - Failure: non-zero

Practice

- Write a complete program that will ask the user for two integers. Display both integers to the screen only if they are each greater than 1000 and terminate the program with **`exit()`** otherwise. Use exactly one **`if/else`**

Floating Point and Relational Operators

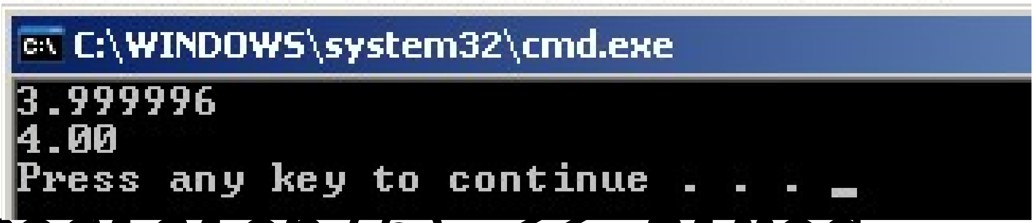
- Floating point math may not work out as you expect because of round off errors.
- In Math
  - 6 * 2/3 = 4
- In C++, where 0.66666 is equivalent to 2/3
  - 6.0 * 0.66666 =
  - 6.0 * 0.66667 =
  - 6.0 * 0.666666 =
  - 6.0 * (2.0 / 3.0) =

Example

```
double result;

result = 6.0 * 0.666666;

if(result == 4.0)
{
  cout << “result == 4.0” << endl;
}

cout << setprecision(6) << fixed
     << result << endl;
cout << setprecision(2) << fixed
    << result << endl;
```

Example

```
double result;

result = 6.0 * 0.666666;

if(result == 4.0)
{
  cout << "result == 4.0" << endl;
}
```



```
C:\WINDOWS\system32\cmd.exe
3.999996
4.00
Press any key to continue . . . _
```

```
cout << setprecision(6) << fixed
     << result << endl;
cout << setprecision(2) << fixed
     << result << endl;
```