# CSC 211: Computer Programming
## Expressions and Selection Statements

Michael Conti

Department of Computer Science and Statistics
University of Rhode Island

Fall 2024

**THINK BIG WE DO**℠

Original design and development by Dr. Marco Alvarez

---

# Expressions
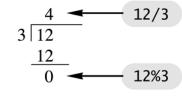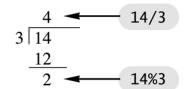
---

# Common arithmetic operators

**+   −   *   /   %**

- Can be used with any numeric type (integers and floating point numbers)

- Result of the **operator** depends on the type of the **operands**

- Be aware of the **integer division** (fractional part discarded)
  - ✓ 22/4 is 5

---

**Integer Division**



$$3\overline{)12} \quad 12/3 = 4$$
$$\phantom{3)}12$$
$$\phantom{3)}\;0 \quad 12\%3 = 0$$

$$3\overline{)14} \quad 14/3 = 4$$
$$\phantom{3)}12$$
$$\phantom{3)}\;2 \quad 14\%3 = 2$$

## "Rules"

- Use parentheses !
  - ✓ even when redundant

- Use whitespaces !

(b*b−4*a*c)/(2*a)  👎

((b * b) − (4 * a * c)) / (2 * a)  👍

## Boolean expressions

- Expressions that evaluate to either `true` or `false`

- Can use comparison operators

**== &lt; &gt; &gt;= &lt;= !=**

- Can use logical operators

**! && ||**

## Truth Tables

### AND

| Exp_1 | Exp_2 | Exp_1 && Exp_2 |
|-------|-------|----------------|
| true  | true  | true  |
| true  | false | false |
| false | true  | false |
| false | false | false |

### NOT

| Exp | !(Exp) |
|-----|--------|
| true  | false |
| false | true  |

### OR

| Exp_1 | Exp_2 | Exp_1 \|\| Exp_2 |
|-------|-------|------------------|
| true  | true  | true  |
| true  | false | true  |
| false | true  | true  |
| false | false | false |

from: Problem Solving with C++, 10th Edition, Walter Savitch

## Comparison Operators

| Math Symbol | English | C++ Notation | C++ Sample | Math Equivalent |
|-------------|---------|--------------|------------|-----------------|
| $=$ | equal to | == | x + 7 == 2*y | $x + 7 = 2y$ |
| $\neq$ | not equal to | != | ans != 'n' | $ans \neq 'n'$ |
| $<$ | less than | < | count < m + 3 | $count < m + 3$ |
| $\leq$ | less than or equal to | <= | time <= limit | $time \leq limit$ |
| $>$ | greater than | > | time > limit | $time > limit$ |
| $\geq$ | greater than or equal to | >= | age >= 21 | $age \geq 21$ |

from: Problem Solving with C++, 10th Edition, Walter Savitch

## Slide 9

**Precedence Rules**

The unary operators +, −, ++, −−, and !. — *Highest precedence (done first)*
The binary arithmetic operations *, /, %
The binary arithmetic operations +, −
The Boolean operations <, >, <=, >=
The Boolean operations ==, !=
The Boolean operations &&
The Boolean operations || — *Lowest precedence (done last)*

from: Problem Solving with C++, 10th Edition, Walter Savitch

## Slide 10

What is the value of this expression?

```
x = 5
(x + 1) > 2 || (x + 1) < −3
```

Recommended style

```
((x + 1) > 2) || ((x + 1) < −3)
```

## Slide 11

In C++ any non-zero value is **true** and zero is **false**

## Slide 12

What is the value of this expression?

```
        false
( ! 32 > 64 )

    ( 0 > 64 )

        false
```

## What is the value of this expression?

```
        true
( ! 0 > 64 )

  (1 > 64)

    false
```

## What is the value of this expression?

```
a = 0;    b = 1;    c = 15;    d = 5;    e = 20;

(!b && !!c) || (d == e) || (!a && ((d + e) % 10 == 0));
```

## Selection Statements
## if and switch

## if statements

‣ Allow conditional execution of code

‣ General idea:

```
if (expression)
    true statement
else
    false statement
```

## The `if` statement (basic syntax)

```
if (expression)
    statementA



if (expression)
    statementA
else
    statementB
```

```
if (expressionA)
    statementA
else if (expressionB)
    statementB
        .
        .
        .
else
    statementN
```

## Example

```cpp
int value;

std::cout << "Enter a number: ";
std::cin >> value;

if (value > 0) {
    std::cout << "positive number" << std::endl;
} else if (value < 0) {
    std::cout << "negative number" << std::endl;
} else {
    std::cout << "zero" << std::endl;
}
```

## Compound statements

```cpp
if (expression) {
    statementA
    statementB
    statementC
    ...
} else {
    statementL
    statementM
    statementN
    ...
}
```

✓ Recommended to **always use braces**, even with single statements

✓ Develop a good and consistent programming style

## Compound statements

```cpp
2
3    #include <iostream>
4    int main( )
5    {
6        double fuelGaugeReading;
7
8        std::cout << "Enter fuel gauge reading: ";
9        std::cin >> fuelGaugeReading;
10
11       std::cout << "First with braces:\n";
12       if (fuelGaugeReading < 0.75)
13       {
14           if (fuelGaugeReading < 0.25)
15               std::cout << "Fuel very low. Caution!\n";
16       }
17       else
18       {
19           std::cout << "Fuel over 3/4. Dont stop now!\n";
20       }
21
22       std::cout << "Now without braces:\n";
23       if (fuelGaugeReading < 0.75)
24           if (fuelGaugeReading < 0.25)
25               std::cout << "Fuel very low. Caution!\n";
26       else
27           std::cout << "Fuel over 3/4. Don't stop now!\n";
28
29       return 0;
30   }
```

## Slide 21

**Compound Statements Used with *if-else***

```
if (my_score > your_score)
{
    cout << "I win!\n";
    wager = wager + 100;
}
else
{
    cout << "I wish these were golf scores.\n";
    wager = 0;
}
```

## Slide 22

**An *if-else* Statement within an *if* Statement**

```
if (count > 0)

    if (score > 5)

        cout << "count > 0 and score > 5\n";

    else

        cout << "count > 0 and score <= 5\n";
```

## Slide 23

# switch statements

- Allow conditional execution of code based on the value of an **integer** expression

- Basic syntax:

```
switch (expression) {
    case valueA:
        statementA
    case valueB:
        statementB
        .
        .
        .
    case valueN:
        statementN
    default:
        statement
}
```

> if expression equals to a value, control executes corresponding statement (can be a compound statement), then continue executing statements until **break** is encountered

## Slide 24

# switch statements

```
3    #include <iostream>
4    int main() {
5    int x = 2;
6        switch (x)
7        {
8            case 1:
9                std::cout << "Choice is 1 \n";
10               break;
11           case 2:
12               std::cout << "Choice is 2 \n";
13               break;
14           case 3:
15               std::cout << "Choice is 3 \n";
16               break;
17           default:
18               std::cout << "Choice other than 1, 2 and 3 \n";
19               break;
20       }
21   return 0;
22   }
```

# switch statements

```cpp
 4    int main() {
 5    int x = 2;
 6        switch (x)
 7        {
 8            case 1:
 9                std::cout << "Choice is 1 \n";
10                //break;
11            case 2:
12                std::cout << "Choice is 2 \n";
13                //break;
14            case 3:
15                std::cout << "Choice is 3 \n";
16                //break;
17            default:
18                std::cout << "Choice other than 1, 2 and 3 \n";
19                //break;
20        }
21    return 0;
22    }
```

---

**A** *switch* **Statement (part 1 of 2)**

```cpp
//Program to illustrate the switch statement.
#include <iostream>
using namespace std;

int main()
{
    char grade;

    cout << "Enter your midterm grade and press Return: ";
    cin >> grade;

    switch (grade)
    {
        case 'A':
            cout << "Excellent. "
                 << "You need not take the final.\n";
            break;
        case 'B':
            cout << "Very good. ";
            grade = 'A';
            cout << "Your midterm grade is now "
                 << grade << endl;
            break;
        case 'C':
            cout << "Passing.\n";
            break;
        case 'D':
        case 'F':
            cout << "Not good. "
                 << "Go study.\n";
            break;
        default:
            cout << "That is not a possible grade.\n";
    }

    cout << "End of program.\n";
    return 0;
}
```

> characters (ascii values) can also be used in switch statements

**A** *switch* **Statement (part 2 of 2)**

**Sample Dialogue 1**

> Enter your midterm grade and press Return: **A**
> Excellent. You need not take the final.
> End of program.

**Sample Dialogue 2**

> Enter your midterm grade and press Return: **B**
> Very good. Your midterm grade is now A.
> End of program.

**Sample Dialogue 3**

> Enter your midterm grade and press Return: **D**
> Not good. Go study.
> End of program.

**Sample Dialogue 4**

> Enter your midterm grade and press Return: **E**
> That is not a possible grade.
> End of program.

---

# Exercise

‣ Write a program in C++ (**on paper**) that:

  ✓ reads the number of **hours**

  ✓ calculates payment:

  - if number of hours no greater than 40, **payment** is calculated using the regular hourly rate of $35

  - if overtime, **payment** is calculated using the regular hourly rate for the first 40 hours and the special rate of $50 for the remaining hours

  ✓ prints the calculated **payment**