Arithmetic Operators

Section 2.15 & 3.2
p 60-63, 81-89

Today

- Arithmetic Operators & Expressions
  - Computation
  - Precedence
  - Algebra vs C++
  - Exponents

Assigning `float`s to `int`s

```
int intVariable;

intVariable = 42.7;

cout << intVariable;
```

- What do you think is the output?

Assigning `double`s to `int`s

- What is the output here?

```
int intVariable;

double doubleVariable = 78.9;

intVariable = doubleVariable;

cout << intVariable;
```

Integer Division

- What is the output?

  -
    ```
    int grade;
    grade = 100 / 20;
    cout << grade;
    ```

  -
    ```
    int grade;
    grade = 100 / 30;
    cout << grade;
    ```

Division

- `grade = 100 / 40;`
    - Check operands of /
        - the data type of grade is not considered, why?
    - We say the integer is *truncated.*

- `grade = 100.0 / 40;`
    - What data type should grade be declared as?

Mathematical Expressions

- Complex mathematical expressions are created by using multiple operators and grouping symbols
  - expression: programming statement that has value
  - $\texttt{sum = }\underbrace{\texttt{21 + 3}}_{\text{expression}}\texttt{;}$
  - $\texttt{number = }\overbrace{\texttt{3}}\texttt{;}$

In these two examples, we assign the value of an *expression* to a variable

Arithmetic Operators

- Operators allow us to manipulate data
  - Unary: `operator operand`
  - Binary: `operand operator operand`

    (left hand side) (right hand side)

Operator	Meaning	Type	Example
-	Negation	Unary	`- 5`
=	Assignment	Binary	`rate = 0.05`
*	Multiplication	Binary	`cost * rate`
/	Division	Binary	`cost / 2`
%	Modulus	Binary	`cost % 2`
+	Addition	Binary	`cost + tax`
-	Subtraction	Binary	`total - tax`

Operator Precedence

- `result = 4 * 2 - 3;`
- `result = 12 + 6 / 3;`
  - `result = ?`
- Rules on how to evaluate an arithmetic expression
  - arithmetic expressions are evaluated left to right
  - do them in order of precedence
  - grouping symbols ( )

Operator Precedence

Precedence of Arithmetic Operators (Highest to Lowest)		
(unary negation) -		
*	/	%
+	-	
(assignment) =		

If two operators have the same precedence, evaluate them from left to right as they appear in the expression

Practice

```
int x = 3;
double y = 2.5;

cout << 5 + 2 * 3;

cout << ( 10 / 2 - y );

cout << 3 + 12 * 2 - 3;

cout << 4 + 17 / 3.0 + 9;

cout << (6 - y) * 9 / x * 4 - 9;
```

If you are unsure, you can always type up and run the code in Visual Studio

Modulus

- Modulus is the remainder after integer division
- `grade = 100 % 20;`
    - `grade` = ?
- `grade = 100 % 30;`
    - `grade` = ?
- `rem = x % n;`
    - What are the possible values for `rem`?

Summary

- Today we have looked at:
  - Arithmetic Operators & Expressions
- Next time we will:
  - Continue looking at mathematic operators
- Completed section 2.15 & started on section 3.2