Arithmetic Operators

- C++ arithmetic operators:

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus

- +, -, *, and / can be used with integral and floating-point data types
- These are *binary operators*
  - They operate on 2 *operands*

Arithmetic Expressions

- Combinations of arithmetic operators and numbers

```
23 + 4
5 - 6 * 20
56.882 - (34 / 23)
```

Order of Precedence

- All operations inside of () are evaluated first
- *, /, and % are at the same level of precedence and are evaluated next
- + and – have the same level of precedence and are evaluated last
- When operators are on the same level
  - Performed from left to right (associativity)
- `3 * 7 - 6 + 2 * 5 / 4 + 6` means

```
(((3 * 7) - 6) + ((2 * 5) / 4 )) + 6
```

Integer vs. Floating Point

- For all arithmetic operators
  - If both operands are integers, returns an integer
  - If either operand is floating-point, returns floating-point
- For integer division, this means truncating the result

  7.0 / 2.0 = 3.5

  7.0 / 2 = 3.5

  7 / 2 = 3

Exercise 1

- Evaluate the following expressions:

A. 13 / 4

B. 3 – 7 % 5

C. 8 + 5 * 2.0

D. 17.0 / 4

Exercise 2

- After these statements are executed, what are the values of variables `a`, `b`, `c` and `d`?

```
a = 3;
b = 4;
c = (a % b) * 6;
d = c / b;
```

Exercise 3

Write a function to compute an average and print the result

```
void avg()
{
   // statements go here
}
```

1. Write C++ statements that *declare* the following variables of type *int*: `num1`, `num2`, `num3` and `average`
2. Write C++ statements that *assign* `125` to `num1`, `28` to `num2` and `-25` to `num3`
3. Write a C++ statement that *assign* the average of `num1`, `num2` and `num3` into `average`
4. Write C++ statements that print the values of `num1`, `num2`, `num3` and `average`