

**NOTES: CH4, CH5, CH6**

# CH4: ARITHMETIC OPERATORS

---

- ✘ Most math is the same as QB (+, -, \*, /)
- ✘ Division is different
  - + Integer division uses same sign as decimal division
  - + Ex:  $11 / 7$  or  $a / b$ .
  - + Resulting value depends on data types of variable and storage variable.
- ✘ See Next Slides

# CODE THIS PROGRAM

```
short a, b, c;  
a = 20, b = 6;  
c = a/b;  
cout << c;
```

Don't Need to Write

Note:  $20 / 6 = 3R2$  or  $3.33333$ . The program above yields 3 because we are dealing with just integers.

# CONSIDER THIS CODE

```
int a, b;  
float c;  
a = 20, b = 6;  
c = a/b;  
cout << c;
```

Don't Need to Write

Will you get 3 or 3.333? Now that c is a float, it can store a decimal number, but a and b are still integers, so the result is still an integer.

# CONSIDER THIS CODE

```
int a, c;  
float b;  
a = 20, b = 6;  
c = a/b;  
cout << c;
```

Don't Need to Write

You still get 3 because  $a/b$  give you 3.3333, but  $c$  can't store a float value.

# TO GET A FLOAT RESULT

- ✘ a or b must be a float to do decimal division instead of integer division.
- ✘ c must be a float to store a decimal number

```
int a;
```

```
float b, c;
```

```
a = 20, b = 6;
```

```
c = a/b;
```

```
cout << c;
```

Don't Need to Write

# WHY NOT USE FLOAT FOR EVERYTHING?

1. Sometimes you want to work with integers...there are a lot of things in life that you can't cut into pieces.
2. Floats take up more memory (RAM) than most integer types.
  1. A goal of good programs is to minimize RAM needed.

# MODULUS

---

- ✘ MODULUS division (MOD) used for integer types to get remainder

```
short a = 20
```

```
short b = 6
```

```
cout << a % b
```

(Result is **2**... $20 / 6 = 3R**2**$ )

# CH4: INCREMENTING AND DECREMENTING

- ✘ In QB, this is a counter:  $x = x + 1$ 
  - + Adds one to the value of  $x$ ;
- ✘ In C++, you can use an incrementer:
  - $x++$  does the same thing as  $x=x+1$ ;
  - You can also use  $++x$
- ✘ The location of the  $++$  determines when the variable will be incremented:
  - + Before the statement or after the statment

# COMPARING CODE

---

Task: Add one to x, then print the value

QB:

Let  $x = x + 1$

Print x

C++:

`cout << ++x;`

*(only one line of c++ code needed)*

Task: Print x, then add one

QB:

Print x

Let  $x = x + 1$

C++:

`cout << x++;`

*(The c++ code prints the value of x before adding one)*