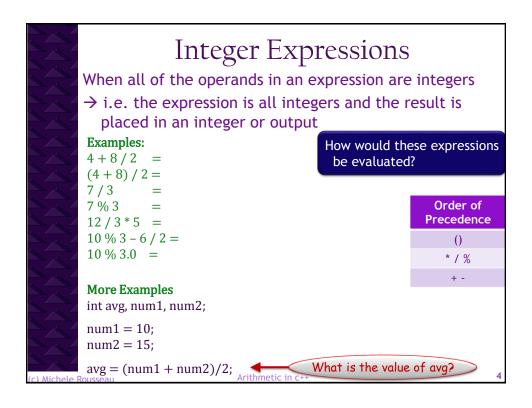
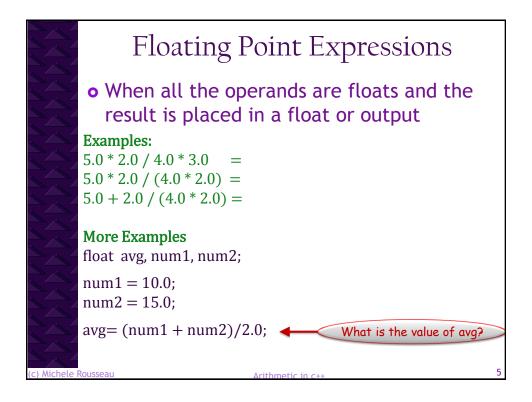
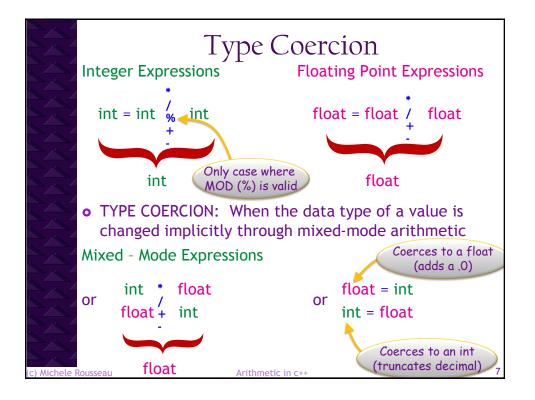


	Arithmetic Operators in C++	
	Symbol	Function
	*	Multiplication
	/	Division
	%	Modulus (remainder from integer division)
	+	Addition
	-	Subtraction
	Order of Precedence	· · · · · · · · · · · · · · · · · · ·
	* / %	according to the order of precedence.
	+ -	of procedence.
(c) Michele Rousseau Arithmetic in c++		





Mixed Mode Arithmetic • Two types of data types to represent numeric values • int & float • They store data differently • Allocate memory differently • i.e. int 6 is stored differently than float 6.0 • Mixed mode arithmetic • → when we combine different data types • e.g. float & int



```
Mixed Mode Arithmetic (2)
o RECALL: We store values in a variable using an
  assignment statement
      variable = expression;
Example;
  given the declarations:
                         Note: this value WILL
   int num1:
                           NOT be rounded
   int num2;
                              THIS IS
   int avg;
                           TYPE COERSION
   num1 = 2: // stores the value 2 in num1
   num2 = 7.75; // truncates the value and stores 7 in num2
   avg = (num1 + num2) / 2;
          // adds 7 + 2 divides by 2 stores 4 in result
```

```
More Examples
given the declarations:
                                The introduction of any float will
int num1
                               cause the expression to convert
int num2:
                               when the float is evaluated
float avg;
             This is called type coercion
                                stores the value 2 in num1
num1 = 2;
num2 = 3.25
                                truncates the value and stores 3 in num2
avg = (num1 + num2) / 2; Converts to float here
// adds 3 + 2 divides by 2 stores the float 2.0 in avg
num1 = 2;
                              Converts to float here
num2 = 3.25;
avg = (num1 + num2) / 2.0;
// adds 3 + 2 divides by 2.0 \leftarrow converts to
      the float then stores 2.5 in avg
```

```
Exercise

GIVEN:
int num;
1. num = 17 % 3;

2. num = 8 / 3 + 2;

3 . num = 6.0 / 12.0 + 5.25;
```

```
Type Casting
Assume:
int age1, age2, ageCount;
float avgAge;
age1
         = 2;
         = 9;
age2
ageCount = 2;
    avgAge = float(age1 + age2) / ageCount;
    • If would add the values age1 and age2, convert them to the floating
      point value 11.0
    • then perform the division producing the desired result 5.5
Which of these would produce an accurate result?
    avgAge = float(age1 + age2) / ageCount;
    avgAge = (age1 + age2) / float(ageCount);
    avgAge = (age1 + age2) / 2.0;
    avgAge = float((age1 + age2) / ageCount );
```

```
Extra Examples
int inum1, inum2;
float fnum3, average;
inum1 = 3;
Inum2 = 7.75;
average = (inum1 + inum2) / 20;
In this case the result will be 0.0 because 20 is an integer
→ the compiler will evaluate these all as integers then
   store as a float so it will store 0.0
How will this differ from?
average = (inum1 + inum2) / 20.0;
In this case the result will be 0.5 because 20.0 is a float
→ the compiler will evaluate the addition as integer then it will
  convert it to float when it divides by 20.0 resulting in 0.5
    This is all referred to as mixed mode arithmetic
    → WARNING: be careful if you are doing this.
```

