

Chapter 7

Arithmetic

Arithmetic in C++

Arithmetic expressions are made up of constants, variables, operators and parentheses. The *arithmetic operators* in C++ are as follows

- + (addition)
- (subtraction)
- * (multiplication)
- / (division)
- % (modulus - the remainder from integer division)

NOTE: The % operator may appear ONLY with integer operands.

When expressions are evaluated, they are evaluated left to right according to the following *precedence rules*:

- ()
- * / %
- + -

The expression $4 + 8 / 2$ is evaluated as follows:

$$4 + 8 / 2 = \quad (\text{division has the highest precedence})$$
$$4 + 4 = 8$$

The expression $(4 + 8) / 2$ is evaluated as follows:

$$(4 + 8) / 2 = \quad (\text{parentheses have the highest precedence})$$
$$12 / 2 = 6$$

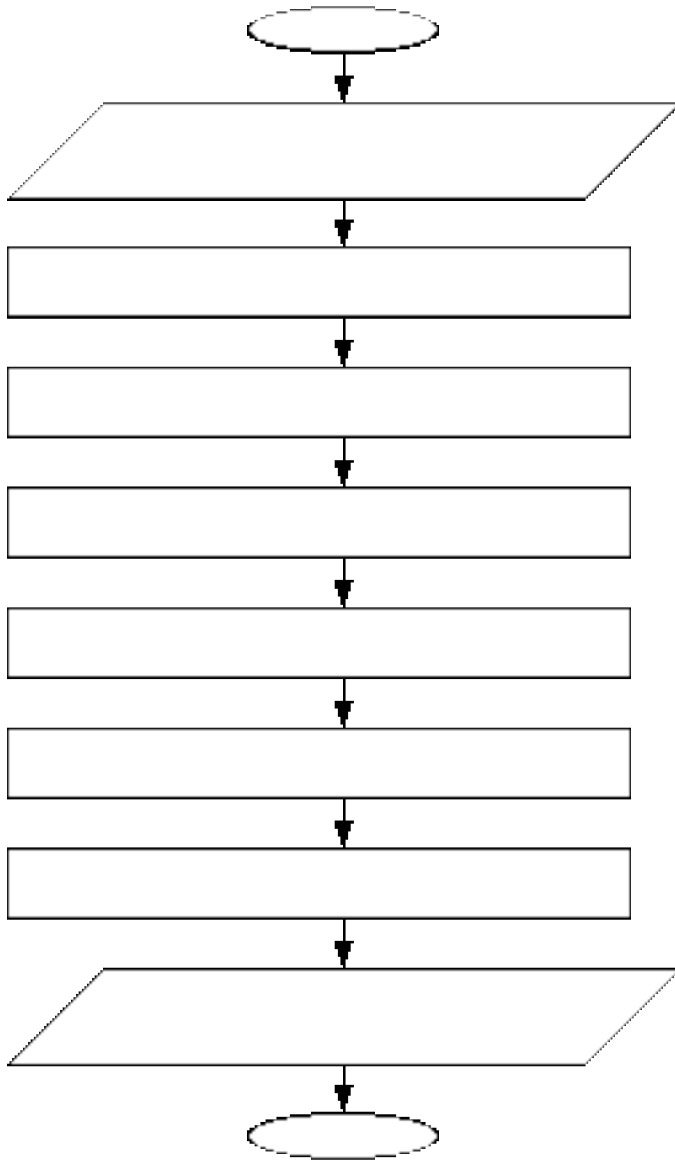
The expression $10 \% 3$ is evaluated as follows:

$10 \% 3 = 1$ (the integer remainder), and
 $10 / 3 = 3$ (the integer quotient)

$$\begin{array}{r} 3 \quad (10 / 3) \\ 3 \overline{) 10} \\ \underline{9} \\ 1 \quad (10 \% 3) \end{array}$$

The expression $8 \% 3.0$ would be invalid (3.0 is a floating point value and therefore not valid), as would the expression $8.0 \% 3$.

A program to calculate the smallest number of quarters, dimes, nickels, and pennies to give for a desired amount of change would be an appropriate application of the modulus operator. Fill in the flowchart for this algorithm using variables called amount (input), quarters, dimes, nickels, and pennies (calculated and output). Perform a desk check with the following input values: 87 and 68



When both of the operands are integers, the expression is said to be an integer expression, integer arithmetic is performed and the result is an integer. When both of the operands are floats, the expression is said to be a float expression, floating point arithmetic is performed and the result is a floating point number. If an expression contains both integers and floating point values, the expression is said to be a mixed expression, floating point arithmetic is performed and the result is a floating point number.

Evaluate the following expressions.

1) $14 / 3 * 5 =$

2) $10 \% 3 - 6 / 2 =$

3) $5.0 * 2.0 / 4.0 * 2.0 =$

4) $5.0 * 2.0 / (4.0 * 2.0) =$

5) $5.0 + 2.0 / (4.0 * 2.0) =$

6) $4 + 4 / 8 =$

Assignment of Mixed Expressions

Recall that we are using two different data types to represent numeric values, `int` and `float`. These two data types require different amounts of memory and the values are stored in different manners. In other words, the integer 6 is stored differently than the floating point value 6.0. When we combine integer and floating point values in the same expression this is called *mixed mode* arithmetic.

One way to place a value into a variable is by using the *assignment statement*. Recall the general form for an assignment statement

```
variable = expression;
```

Given the declarations

```
int num1;  
int num2;  
float result;
```

`num1` and `num2` may hold integer values only and `result` will hold a floating point value. What happens if the following assignment statements appear in a program?

```
num1 = 3;  
num2 = 7.75;  
result = (num1 + num2) / 2.0;
```

The value 3 is a valid integer and is stored in location `num1`. The value 7.75 is a floating point value so C++ simply truncates (cuts off) the fractional part and stores the integer value 7 in location `num2`.

NOTE: The value is not rounded. The calculation for the average adds the two integer values 3 and 7 and divides the result by the floating point value 2.0. The result is the floating point value 5.0. This automatic conversion performed by the compiler (storing 7 in `num2` rather than 7.75) is called *type coercion*.

Consider the following:

```
num1 = 3;  
num2 = 7.75;  
result = (num1 + num2) / 20;
```

The result is the integer value 0. The integer values 3 and 7 are added then divided by the integer value 20. The integer quotient is 0 and is stored in the variable result (the 0 is *coerced* to 0.0). To obtain the desired value the expression must contain a floating point value. Write the expression as follows

```
result = (num1 + num2) / 20.0;
```

to store the floating point value 0.5 in the variable result. Be very careful with mixed mode arithmetic. We will examine another method for dealing with this problem later.

Exercises in C++ Arithmetic Expressions

What value is stored into the integer variable (num) after each of the following expressions has been evaluated?

- 1) $\text{num} = 17 \% 3;$
- 2) $\text{num} = 8 / 3 + 2;$
- 3) $\text{num} = 6.0 / 12.0 + 5.25;$
4. $\text{num} = 3.75 + 5 * 2;$

For each of the following indicate whether the expression is valid or invalid. If the expression is valid, indicate whether the result is integer or floating point.

- 1) $10.0 / 3.0 + 5 * 2$
- 2) $10 / 4 + 6 / 3$
- 3) $10 \% 4 + 6 \% 3$
- 4) $(10.0 / 3.0 \% 2) / 3$
5. $13.25 + (5.0 (3.0 / 3.5))$
6. $-4 * (-5 + 6)$

Assignment Statements and Type Coercion

Assignment statements become a little tricky. When examining assignment statements, we will break the process into two parts

- 1) evaluate the expression on the right side of the assignment operator
- 2) apply any type coercion based on the data type of the variable on the left side of the assignment operator (=)

```
float n1;  
float n2;  
float n3;  
float n4;  
int n5;  
int n6;
```

```
n1 = 3.2;  
n2 = 2.0;  
n3 = 0.4;
```

Evaluate each of the following (show the final value assigned):

a) $n5 = n1 / n2 * 3 + n3$;

b) $n4 = 5 / 10 * n1$;

c) $n6 = n1 * n3$;

Assignment Statements and Type Casting

We defined type coercion as the implicit conversion from one data type to another done by the compiler. Consider the following

```
int age1;  
int age2;  
float avgAge;  
  
age1 = 5;  
age2 = 6;  
avgAge = (age1 + age2) / 2;
```

In the above statement, all of the operands are of type int. The expression would evaluate as follows

$$(5 + 6) / 2 = 11 / 2 = 5$$

Type coercion would take place and the value 5.0 stored in variable avgAge.

One solution is to divide by the floating point literal constant 2.0. Another approach is to use type casting.

Type Casting - the explicit conversion from one data type to another done by the programmer.

The statement

```
avgAge = float(age1 + age2) / 2;
```

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.

```
avgAge = float(age1 + age2) / 2;           // valid  
avgAge = (age1 + age2) / float(2);        // valid  
avgAge = (age1 + age2) / 2.0;             // valid  
avgAge = float( (age1 + age2) / 2 );      // invalid result - why?
```

Formula Conversion Worksheet

Convert each of the following to proper C++ format.

$$\frac{a + b}{cd}$$

$$p = [s(s-a)(s-b)(s-c)]^2$$

$$i - (n(m + c))$$

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Simple Sequence Problems

These problems deal with the control/logic structure simple sequence and require only cin, cout and the assignment statement. Complete the structure chart, variable list and flowchart for each of the following problems.

When it is time to transfer the flowcharts to an actual C++ program be sure to review Chapters 4 and 5. It is important that you adhere to the style required for this course.

- 1.) Design a program to calculate the average gasoline consumption for a trip. Assign the following values to variables: 15.25 gallons for fill up 1, 12.5 gallons for fill up 2 and 18.75 gallons for fill up 3. The odometer read 10,575 when you left on your trip and 11,363 when you arrived at your destination. Output the average miles per gallon for this trip. Note: This program requires only assignment statements and cout. This program is not interactive.
- 2.) Make the necessary changes to the program from question 1 to make it *interactive*.
- 3.) Design a program to obtain from the user, an annual interest rate, the term of the loan (in years) and the amount of money borrowed. Your program is to calculate and output the monthly payment on this loan using the following formula:

$$\text{monthlyPmt} = \frac{\text{rate}(1 + \text{rate})^{\text{term}}}{(1 + \text{rate})^{\text{term}} - 1} \times \text{loanAmt}$$

Use the following values to test your program.

Loan Amount	120000.00
Interest Rate	7.0
Term (years)	30

It is necessary to convert the formula to C++ format. You will recall that there is no exponentiation operator available in C++. We will use a function called *pow()* for this purpose.

GENERAL FORM for pow(x,y) function

pow(x,y) where x and y are integer or float values

This function returns the result **x raised to the power y**. Example:

pow(3,2) returns the value 9
pow(4,0.5) returns the value 2

A value-returning function returns a single value. This value may be returned to a variable or an expression. Examples:

```
float result;
```

```
result = pow(3,2);
```

or

```
cout << "3 to the power of 2 is " << pow(3,2);
```

or

```
result = 10 + pow(4,0.5);
```

Write the C++ formula to calculate the payment here.

Review (Ch. 3 - 7)

Vocabulary:

identifier -

variable -

named constant -

literal constant -

data type -

assignment statement -

assignment operator -

cin -

cout -

statement separator -

list and define the 3 control/logic structures -

Basic C++ Concepts Review

1. Given the following:

```
1. void main(void)
2. {
3.     const float DENOM = 2.0;
4.     int num1;
5.     int num2;
6.     float average;
7.
8.     cout << "Enter first value: ";
9.     cin >> num1;
10.    cout << "Enter second value: ";
11.    cin >> num2;
12.    average = (num1 + num2) / DENOM;
13.    cout << "\nThe average is " << average;
14. }
```

num1, num2, main, int, and DENOM are all examples of _____.

num1 is a _____ of type _____.

DENOM in line 12 is an example of a _____ of type _____.

"Enter first value: " in line 8 is an example of a _____ of type _____ and is used to _____ the user for input.

The statement on line 12 is called a(n) _____.

The ; is used in C++ to _____.

The = in line 12 is called the _____.

>> is called the _____ and is used with a _____ statement and << is called the _____ and is used with a _____ statement.

(num1 + num2) / DENOM in line 12 is called an _____.

Given the following:

```
int n1, n2;  
float n3, n4;
```

Show the value stored in the variable after each of the following:

a) $n4 = 4 / 5 * 1.5;$

b) $n1 = 4 \% 5 * 1.5;$

c) $n2 = 1.5 * 3 + 0.4;$

"a" is an example of a literal constant of type char. T/F

For each data type listed below, give an example of a literal constant, the declaration of a variable of that type, and the declaration of a named constant of that type.

literal constant variable declaration named constant declaration

int

float

char

c-string

Program Design Process Review

Step 1. Make a list of all the necessary variables and named constants for the program.

Step 2: Read the program description and identify the control/logic structures. Add any necessary variables you might need.

Step 3. Draw the flowchart for the program.

Step 4. Perform a desk check looking for any logic problems. Correct these problems before continuing.

Step 5. Choose an appropriate data type for each variable and write the C++ declaration section.

Recall:

```
char someChar;    // someChar is of data type char and may hold one
                  // single alpha-numeric character enclosed in single
                  // quotes ('A', 'x', '! ' etc.)
```

```
char name[15];    // name is of data "type" c-string and may hold multiple
                  // characters enclosed in double quotes. Variable name
                  // may hold 14 characters with the 15th character being
                  // the null terminator (\0). Other examples of cstrings
                  // include "Hello world", "Enter first value", etc.
```

Step 6. Write this program using the following as a guide. FOLLOW YOUR FLOWCHART EXACTLY. Any logic errors should have been found and corrected in step 4.

- preprocessor directives
- documentation (name, class section, assignment name, due date)
- heading for the main function
- declaration section
- input section
- processing section
- output section

C++ Basics

T/F The statement
 `const int DAY = 5;`
is an example of an assignment statement

T/F The statement
 `average = (val1 + val2) / 2.0;`
is an example of an assignment statement

Given the following:

```
const char NAME[4] = "Joe";  
int num1;  
char theCh;
```

```
char is a _____  
NAME is a _____ of type _____  
num1 is a _____ of type _____  
int is a _____  
"Joe" is a _____ of type _____  
4 is _____ of type _____  
theCh is a _____ of type _____
```

Label each of the following as valid or invalid. If invalid, explain why.

```
theCh = 'x';
```

```
theCh = "a";
```

```
theCh = '5';
```

```
theCh = c;
```

Arithmetic Expressions

Given two int locations called n1 and n2, and two float locations called n3 and n4, show the value stored after each of the following expressions has been evaluated.

n3 = pow(4, 1/2);

n1 = 2.5 + 3.4;

n1 = 5 / 8;

n2 = 5 % 8;

n3 = 3 / 6;

n4 = 3.0 / 6.0;

n3 = 3 % 6.0;

n4 = pow(4, 1/2.0);

_____ is the implicit conversion from one data type to another done by the compiler.

_____ is the explicit conversion from one data type to another done by the programmer.

Name _____

Due Date _____

Remaining Balance Homework

DO NOT WORK ON THIS ASSIGNMENT WITH ANOTHER STUDENT

Suppose you have purchased a car, home etc. You want to keep track of the remaining balance you owe after each monthly payment. Use the calculation for the monthly payment (7-12) in this program. The formula for this remaining balance calculation is as follows:

$$\text{bal}_k = \text{pmt} \left[\frac{1 - (1 + i)^{k-n}}{i} \right]$$

where,

- bal_k = balance remaining after k th payment
- k = payment number (1, 2, 3, ...)
- pmt = amount of the monthly payment
- i = interest rate per month
- n = total number of payments

Use the following user interface:

Original Loan Amount:

Monthly Payment:

Annual Interest Rate:

Term of Loan (Years):

Show Balance After Payment #:

Balance After Payment # \$: (calculated balance displayed here)

Flowchart and code the above program. Be sure to use meaningful identifier names (not i , n , k etc.) and make any necessary modifications to the user input. We know from the payment program we did in class that the monthly payment on a \$120,000 loan at 7% interest for 30 years is \$798.36. Using this information, run your program three times and obtain the balance after one year (12th payment), ten years (120th payment), and twenty years.

A sample run should look as shown on the next page - use this result to check your program. Be sure your user interface and output look EXACTLY as shown below. Your program will be graded accordingly.

Original Loan Amount: 120000.00
Monthly Payment: 798.36
Annual Interest Rate: 7.0
Term of Loan (Years): 30
Show Balance After Payment #: 12

Balance After Payment # 12 \$: 118780.92

Refer to the style guide in Chapter 5 of the shrink wrap. You are responsible for the style shown. Turn in IN THIS ORDER

- this sheet
 - flowchart (template or software)
 - listing of the code
 - 3 runs of your program using data provided
- STAPLE IN THE UPPER LEFT CORNER**

Name _____

Due Date _____

Remaining Balance Homework EXTRA CREDIT
DO NOT WORK ON THIS ASSIGNMENT WITH ANOTHER STUDENT

Add the necessary calculations to produce the following output:

Original Loan Amount: 120000
Monthly Payment: 798.36
Annual Interest Rate: 7.0
Term of Loan (Years): 30
Show Balance After Payment #: 360

Remaining Balance:	\$	0.00
Total Paid to Date:	\$	287409.59
Interest Paid To Date:	\$	167409.59
Amount Applied to Principle:	\$	120000.00

TURN THIS IN AS A SEPARATE ASSIGNMENT - DO NOT COMBINE IT WITH THE BASIC ASSIGNMENT. RUN THIS PROGRAM TWICE. ONCE USING THE DATA PROVIDED ABOVE AND AGAIN USING THE SAME LOAN AMOUNT BUT SHOWING THE BALANCE AFTER PAYMENT #180.

Turn in (IN THOS ORDER):

- this sheet
- flowchart
- listing of the code
- listing of the output (2 sample runs)