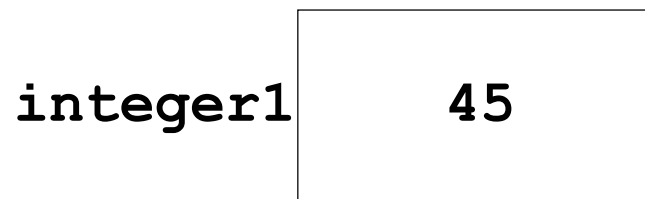


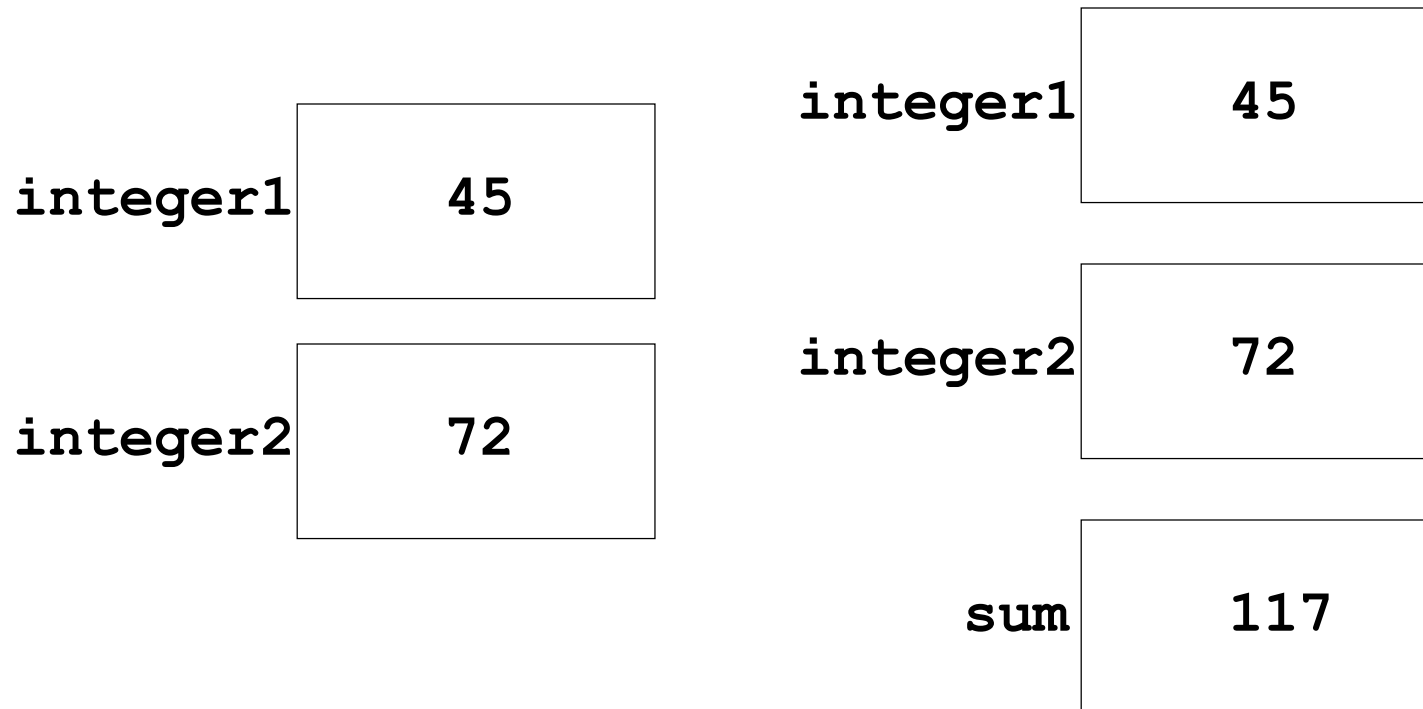
Memory Concepts

- Variables
 - Variable names correspond to locations in the computer's memory; the compiler assigns the actual locations
 - Every variable has a name, a type, a size and a value
 - Whenever a new value is placed into a variable it replaces the previous value
 - Reading variables from memory does not change them
- Declare `int integer1;`, then write `integer1 = 45;`



Memory Concepts

- Declare `int integer2;`
- Declare `int sum;`
- Write `integer2 = 72;`
- Now write `sum = integer1 + integer2;`



Arithmetic

- Arithmetic calculations
 - Use * for multiplication and / for division
 - Integer division truncates remainder
 - $7 / 5$ evaluates to 1
 - Modulus operator(%) returns the remainder
 - $7 \% 5$ evaluates to 2

Division and Remainder

- If both operands to the division operator (/) are integers, the result is an integer (the fractional part is discarded)

14 / 3 equals? 4

8 / 12 equals? 0

- The remainder operator (%) returns the remainder after dividing the second operand into the first

14 % 3 equals? 2

8 % 12 equals? 8

Arithmetic Expressions

- An *expression* is a combination of one or more operands and their operators
- *Arithmetic expressions* compute numeric results and make use of the arithmetic operators:

Addition	+	
Subtraction	-	
Multiplication	*	
Division		/
Remainder	%	

- If either or both operands associated with an arithmetic operator are floating point, the result is a floating point

Arithmetic

- Operator precedence - you should use parentheses to "say what you mean and mean what you say"
- Operator precedence makes behavior well-defined in absence of parentheses
 - Some arithmetic operators act before others (i.e., multiplication before addition)
 - Example: Find the average of three variables a, b and c
 - Do not use: $a + b + c / 3$
 - Use: $(a + b + c) / 3$

Arithmetic

- Arithmetic operators:

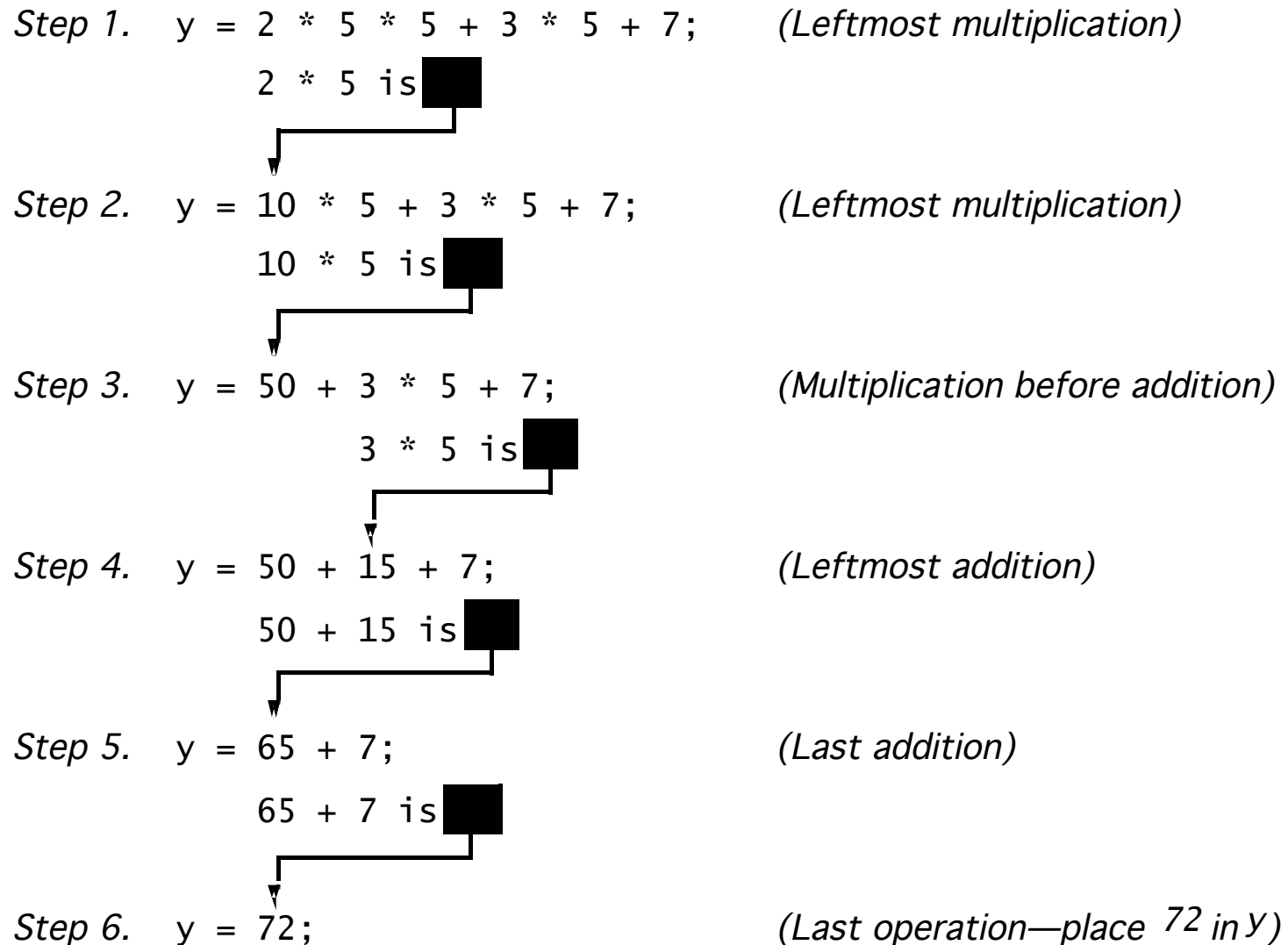
C operation	Arithmetic operator	Algebraic expression	C expression
Addition	+	$f + 7$	$f + 7$
Subtraction	-	$p - c$	$p - c$
Multiplication	*	bm	$b * m$
Division	/	x / y	x / y
Modulus (remainder)	%	$r \text{ mod } s$	$r \% s$

- Rules of operator precedence:

Operator(s)	Operation(s)	Order of evaluation (precedence)
()	Parentheses	Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses “on the same level” (i.e., not nested), they are evaluated left to right.
*, /, or %	Multiplication, Division, Modulus	Evaluated second. If there are several, they are evaluated left to right.
+ or -	Addition Subtraction	Evaluated last. If there are several, they are evaluated left to right.

Operator Precedence

equal precedence: left-to-right



Assignment

- The assignment operator has a lower precedence than the arithmetic operators

First the expression on the right hand side of the = operator is evaluated

```
answer = sum / 4 + MAX * lowest;
```

4 1 3 2



Then the result is stored in the variable on the left hand side