CS 152
Computer Programming Fundamentals
Expressions and Operators

Brooke Chenoweth

University of New Mexico

Spring 2024

Expressions

An *expression* piece of program code that represents or computes a value.

- literal
- variable
- function call
- several expressions combined with operators

Using Expressions

Value of an expression can be used many ways

- Assign to variable
- Use as parameter to method call
- Combine with other values into more complicated expression
- Ignore it

Arithmetic Operators

```
int x = 5;
int y = 2;

int sum = x + y;        // 7 addition
int diff = x - y;       // 3 subtraction
int mult = x * y;       // 10 multiplication
int div = x / y;        // 2 division
int remainder = x % y;  // 1 modulus
```

Integer and Floating Point Math

- When two numeric values are combined (after converting type of one, if necessary), the answer will be the same type.
  - Multiply two `ints`, get a `int`
  - Multiply two `doubles`, get a `double`
- Must be careful with division.
  - Divide two `ints`, result is `int` also. Any fractional part of answer is lost!
  - If you want a floating point answer, cast operand before computing.

Relational Operators

We often want to be able to compare numeric types.

```
A == B    // Is A equal to B?
A != B    // Is A not equal to B?
A < B     // Is A less than B?
A > B     // Is A greater than B?
A <= B    // Is A less than or equal to B?
A >= B    // Is A greater than or equal to B?
```

These expressions all compute `boolean` values.

Boolean Operators

```
A && B  // AND: Are both A and B true?
A || B  // OR: Are either A or B true?
!A      // NOT: Negate value of A
```

A	B	A and B	A or B
F	F	F	F
F	T	F	T
T	F	F	T
T	T	T	T

A	not A
F	T
T	F

Assignment

Value on right hand side of assignment operator must be same type as variable on the left. In some cases, computer will automatically convert the value to match the type. You can force a conversion that won't automatically happen by using a *type cast*.

```
int a = 5;
double b = 4.2;

// int converted to double with same value
double c = a;

// explicitly type cast, losing fractional part
int d = (int)b;
```

Assignment operator variations

```
x  -= y;  // same as: x = x - y;
x  *= y;  // same as: x = x * y;
x  /= y;  // same as: x = x / y;
x  %= y;  // same as: x = x % y;  (for integers x, y)
q &&= p;  // same as: q = q && p;  (for booleans q, p)
```

Every operator that applies to two operands has a
similar assignment operator.

Increment and Decrement

Adding 1 to a variable is an extremely common operation in programming. Could use assignment statement like:

```
x = x + 1;
```

This takes old value of x, computes the result of adding 1 to that value, and stores the answer as the new value of x.

Shorter way to do same thing would be:

```
x++;
```

++ is the increment operator. -- is the decrement operator, for subtracting 1 from a value.