Code Academy

Chapter Two

PROGRAMMING WITH NUMBERS AND STRINGS

Sections 2.2 – 2.3
Arithmetic and Problem Solving

Code Academy

Lecture Goals

- To write arithmetic expressions and assignment statements

Code Academy

2.2 Arithmetic

Basic Arithmetic Operations

- Python supports all of the basic **arithmetic operations**:
  - Addition +
  - Subtraction -
  - Multiplication *
  - Division /
- The symbols + − * / for the arithmetic operations are called **operators**.
- The combination of variables, literals, operators, and parentheses is called **expression**. For example, `(a + b)/2` is an expression.
- You write your expressions a bit differently when coding

$$\frac{a+b}{2}$$

s

```
(a + b) / 2
```

Code Academy

Precedence

- **Precedence** is similar to Algebra:
  - PEMDAS
    - Parenthesis, Exponent, Multiply/Divide, Add/Subtract

Precedence	Operator	Description
Higher ↓	( )	Parenthesis, Function Call
	+ -	Positive Negative
	**	Exponent
	* / // %	Multiplication Real Division Floor Division Modulus (remainder)
	+ -	Addition Subtraction
Lower	=	Assignment

Precedence

- Higher **precedence** determines which operator is applied first in an expression having several operators.
  - For example: `3 + 2 * 5` means `3 + (2 * 5) = 3 + 10 = 13`
- Operators with the same precedence are executed left-to-right.
  - For example: `9 - 5 - 1` means `( 9 - 5 ) - 1 = 4 - 1 = 3`
- What are the values of the following expressions?
  - `3 + 4 * 2 / 8`

    4
  - `5 * 3 - 4 / 2`

    13

Code Academy

Mixing numeric types

- If you mix **integer** and **floating-point** values in an arithmetic expression, the result is a floating-point value.

```
7 + 4.0    # Yields the floating value 11.0
```

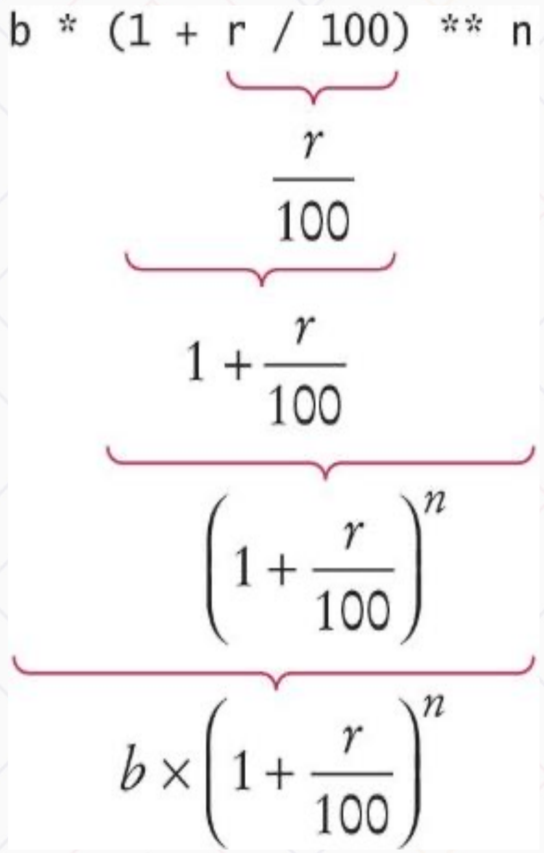
- If you mix strings with floating point values the result is an error

Powers

- Double stars ** are used to calculate an exponent
- Analyzing the expression:

$$b \times \left(1 + \frac{r}{100}\right)^n$$

- Becomes:
  - b * ((1 + r / 100) ** n)



```
b * (1 + r / 100) ** n
```

$$\frac{r}{100}$$

$$1 + \frac{r}{100}$$

$$\left(1 + \frac{r}{100}\right)^n$$

$$b \times \left(1 + \frac{r}{100}\right)^n$$

Code Academy

Floor division

- When you divide two integers with the / operator, you get a floating-point value.
  - For example,

    7 / 4

  yields 1.75

- We can also perform **floor division** using the // operator.
  - The // operator computes the quotient and discards the fractional part

    7 // 4

    - Evaluates to 1 because 7 divided by 4 is 1.75 with a fractional part of 0.75, which is discarded.

- Examples

8 // 3	2
2 / 5	0.4
6 // 7	0

Code Academy

Calculating a remainder

- If you are interested in the remainder of dividing two integers, use the % operator (called modulus):

  remainder = 7 % 4

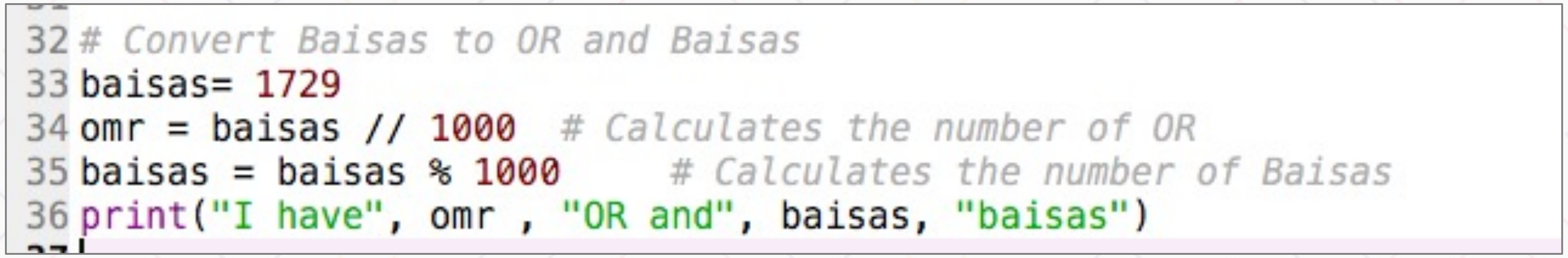
  - The value of remainder will be 3

- Sometimes called modulo divide
- 8 % 3

- Examples

Examples	Value
8 % 3	2
2 % 5	2
10 % 7	3
6 % 2	0

A Simple Example:



```
# Convert Baisas to OR and Baisas
baisas= 1729
omr = baisas // 1000  # Calculates the number of OR
baisas = baisas % 1000      # Calculates the number of Baisas
print("I have", omr , "OR and", baisas, "baisas")
```

- What is the result?

```
I have 1 OR and 729 Baisas
```

Integer Division and Remainder Examples

- Hand
  - pe
  - dc
  - ce

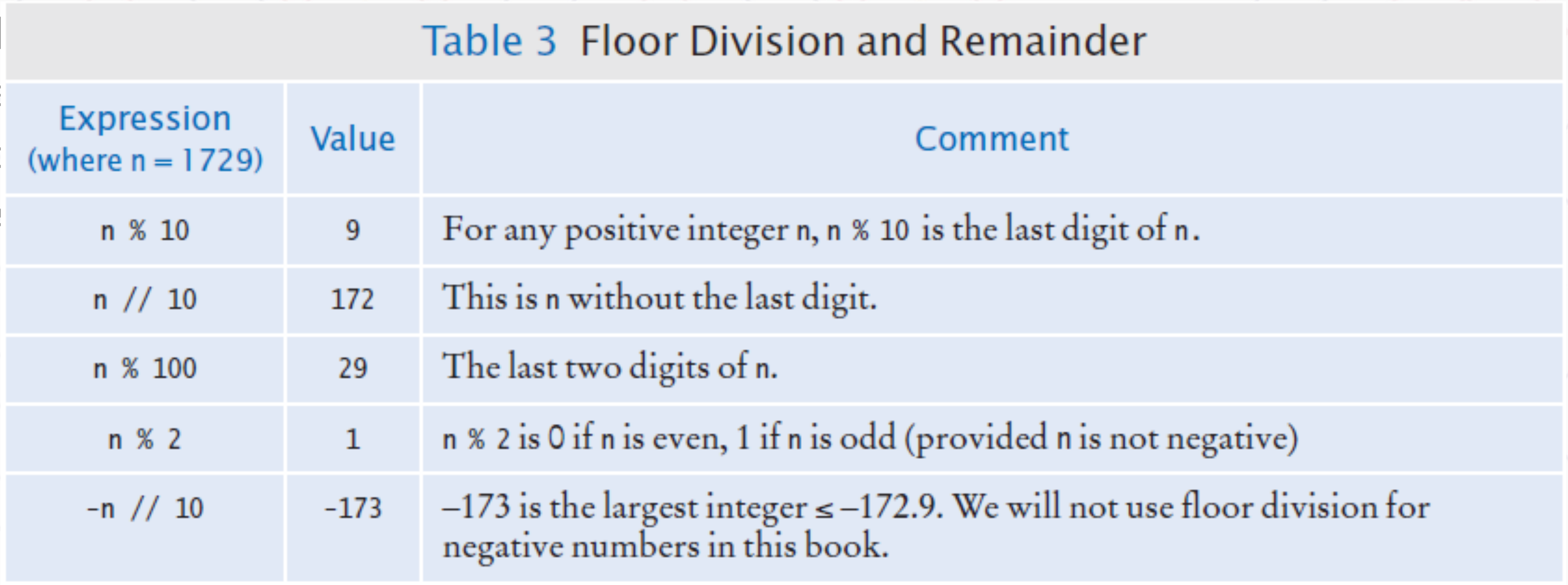


Table 3 Floor Division and Remainder

Expression (where n = 1729)	Value	Comment
`n % 10`	9	For any positive integer n, `n % 10` is the last digit of n.
`n // 10`	172	This is n without the last digit.
`n % 100`	29	The last two digits of n.
`n % 2`	1	`n % 2` is 0 if n is even, 1 if n is odd (provided n is not negative)
`-n // 10`	-173	–173 is the largest integer ≤ –172.9. We will not use floor division for negative numbers in this book.

Code Academy

Calling functions

- Recall that a **function** is a collection of programming instructions that carry out a particular *task*.
- The `print()` function can display information, but there are many other functions available in Python.
- When calling a function you must provide the correct number of **arguments**
  - The program will generate an error message if you don’t
  - For example, if you call

```
abs(-10, 2)
```

or

```
abs()
```

your program will generate an error message because `abs()` takes only one argument.

Code Academy

Calling functions that return a value

- Most functions **return** a value. That is, when the function completes its task, it passes a value back to the point where the function was called.
  - For example:

```
>>> abs(-173) #returns the value 173.
```

- The value returned by a function can be stored in a variable:

```
>>> distance = abs(x)
```

- You can use a function call as an argument to the **print** function
  - What is the result of

```
>>> print(abs(-173))


>>> 173
```

Built in Mathematical Functions

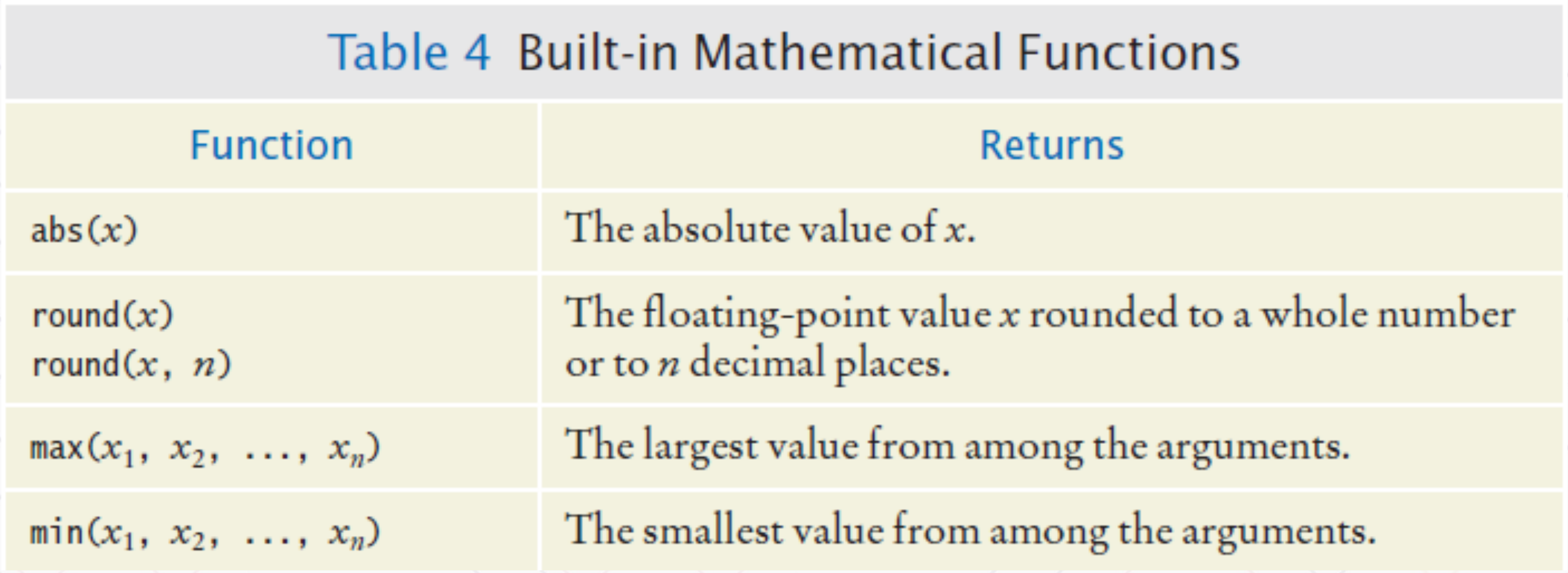


Table 4 Built-in Mathematical Functions

Function	Returns
`abs(`x`)`	The absolute value of x.

`round(`x`)`
`round(`x`, `n`)`
	The floating-point value x rounded to a whole number or to n decimal places.

`max(`x_1`, `x_2`, ..., `x_n`)`	The largest value from among the arguments.
`min(`x_1`, `x_2`, ..., `x_n`)`	The smallest value from among the arguments.

- Examples:

```
>>> round(45.347)    #returns the value 45
>>> round(45.347, 2) #returns the value 45.35
>>> min(7.25, -1.95, 5.95, 6.05)    #returns the value -1.95
```

Python libraries (modules)

- A **library** is a collection of code, written and compiled by someone else, that is ready for you to use in your program
- A **standard library** is a library that is considered part of the language and must be included with any Python system.
- Python's standard library is organized into **modules**.
  - Related functions and data types are grouped into the same module.
  - Functions defined in a module must be explicitly loaded into your program before they can be used.
- **Built-in** functions are a small set of functions that are defined as a part of the Python language
  - They can be used without importing any modules

Code Academy

Using functions from the Math Module

- For example, to use the sqrt() function, which computes the square root of its argument:

```
# First include this statement at the top of your
# program file.
from math import sqrt

# Then you can simply call the function as
x = 2
y = round(sqrt(x), 3)

print(y)          # 1.414
```

Functions from the Math Module

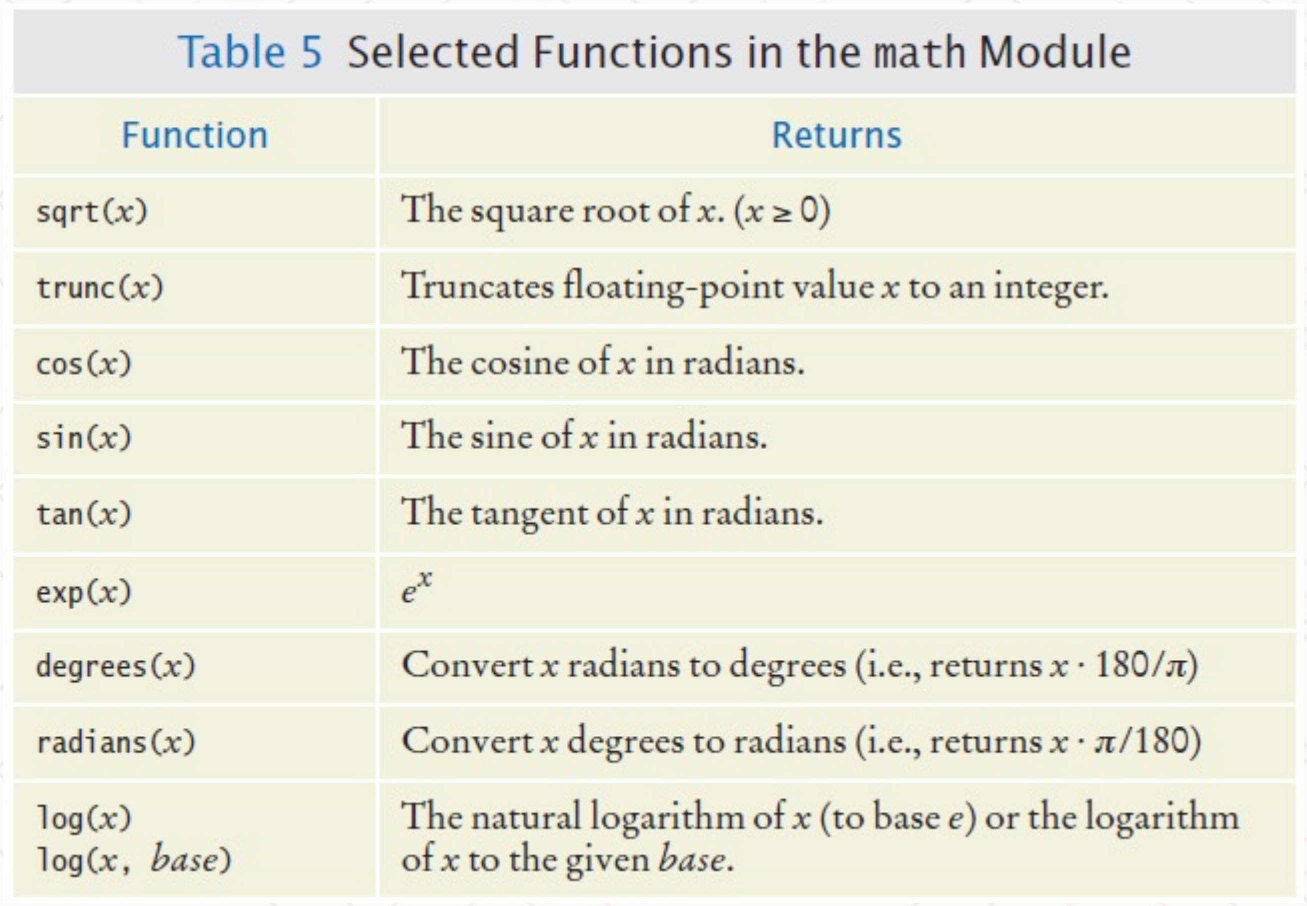


Table 5 Selected Functions in the math Module

Function	Returns
`sqrt(x)`	The square root of x. ($x \geq 0$)
`trunc(x)`	Truncates floating-point value x to an integer.
`cos(x)`	The cosine of x in radians.
`sin(x)`	The sine of x in radians.
`tan(x)`	The tangent of x in radians.
`exp(x)`	e^x
`degrees(x)`	Convert x radians to degrees (i.e., returns $x \cdot 180/\pi$)
`radians(x)`	Convert x degrees to radians (i.e., returns $x \cdot \pi/180$)

`log(x)`
`log(x, base)`
	The natural logarithm of x (to base e) or the logarithm of x to the given *base*.

Code Academy

Floating-point to integer conversion

- You can use the function `int()` and `float()` to convert between integer and floating point values:

```
balance = total + tax   # balance: float
dollars = int (balance) # dollars: integer
```

You lose the fractional part of the floating-point value (no rounding occurs)

Arithmetic Expressions

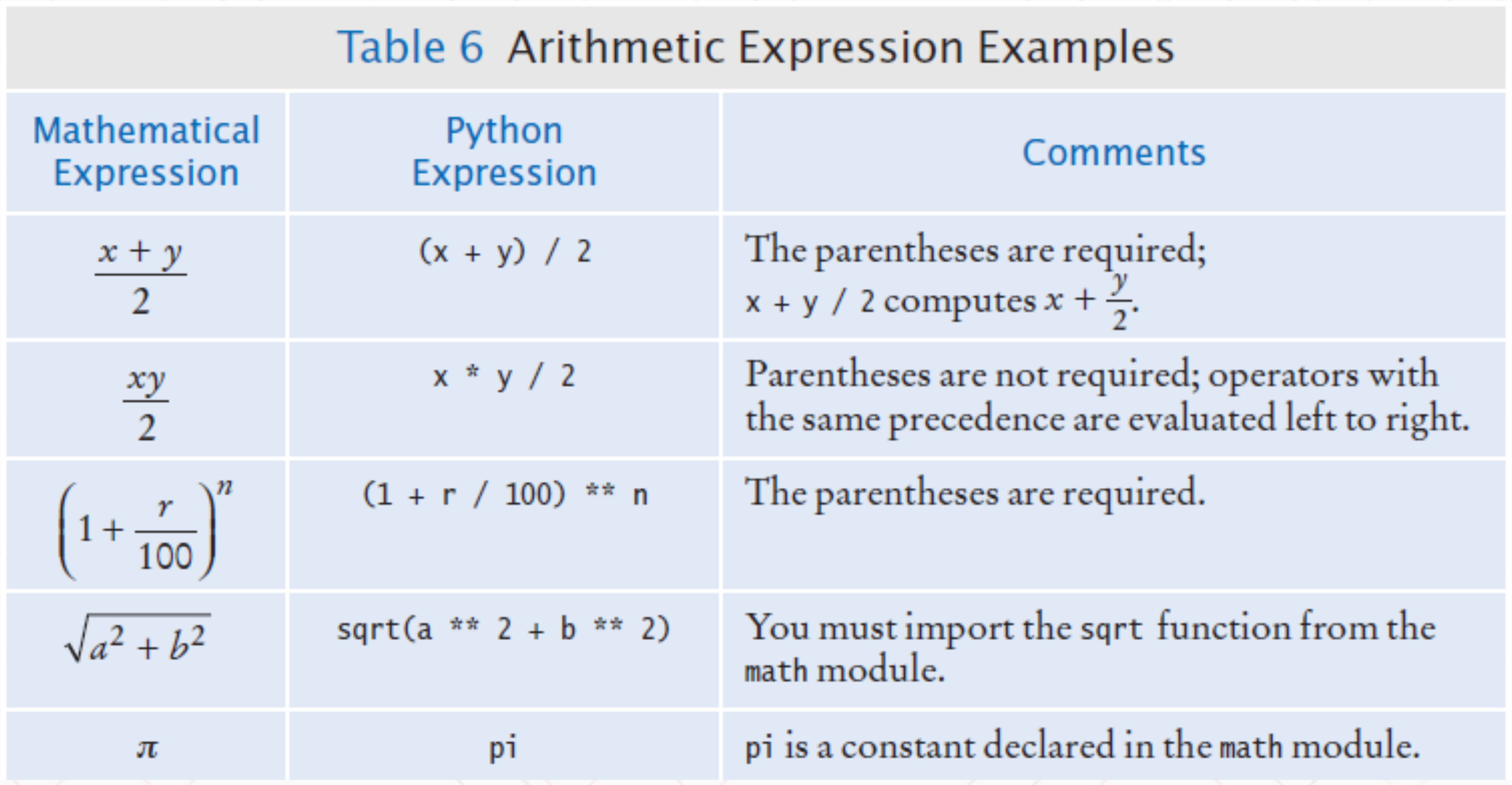


Table 6 Arithmetic Expression Examples

Mathematical Expression	Python Expression	Comments
$\frac{x+y}{2}$	`(x + y) / 2`	The parentheses are required; `x + y / 2` computes $x + \frac{y}{2}$.
$\frac{xy}{2}$	`x * y / 2`	Parentheses are not required; operators with the same precedence are evaluated left to right.
$\left(1+\frac{r}{100}\right)^n$	`(1 + r / 100) ** n`	The parentheses are required.
$\sqrt{a^2+b^2}$	`sqrt(a ** 2 + b ** 2)`	You must import the `sqrt` function from the `math` module.
π	`pi`	`pi` is a constant declared in the `math` module.

Code Academy

Arithmetic Expressions Examples

- Evaluate the following expression:

7 * 10 - 5 % 3 * 4 + 9

(7 * 10) - 5 % 3 * 4 + 9

70 - 5 % 3 * 4 + 9

70 - (5 % 3) * 4 + 9

70 - 2 * 4 + 9

70 - (2 * 4) + 9

70 - 8 + 9

(70 - 8) + 9

62 + 9

71

- What about this: **(7 *(10 - 5) % 3) * 4 + 9**

Code Academy

More Examples

- What is the Python expressions of the following mathematical notation?

$$c = \sqrt{a^2 + b^2 - 2ab\cos t}$$

```
c = sqrt(a ** 2 + b ** 2 - 2 * a * b * cos(t))
```

- What is the mathematical notation of the following Python expression?

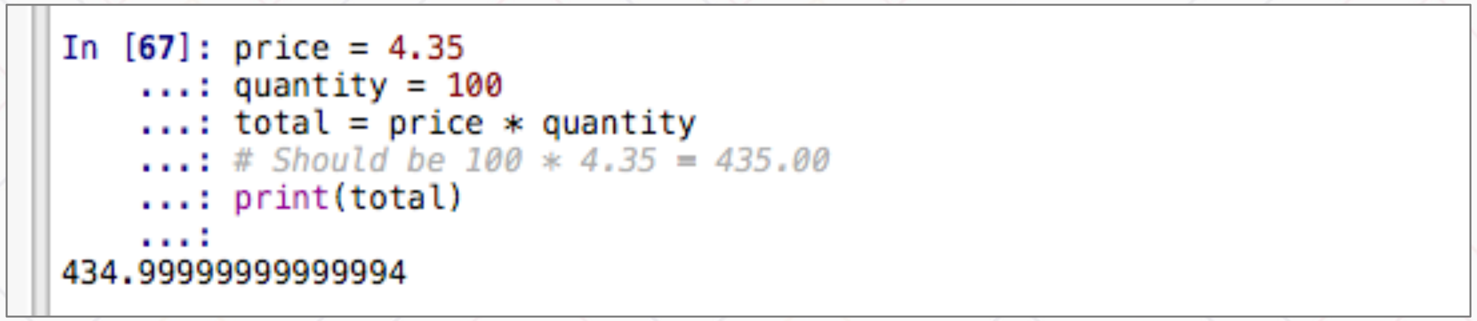
```
g = 4 * n ** 2 * (a ** 3 / (p ** 2 * (x + y)))
```

$$g = 4n^2 \frac{a^3}{p^2(x+y)}$$

Code Academy

Roundoff Errors

- Floating point values are not exact
  - This is a limitation of binary values; not all floating point numbers have an exact representation
- Example:



```
In [67]: price = 4.35
    ...: quantity = 100
    ...: total = price * quantity
    ...: # Should be 100 * 4.35 = 435.00
    ...: print(total)
    ...:
434.99999999999994
```

- You can deal with roundoff errors by
  - rounding to the nearest integer or by
  - displaying a fixed number of digits after the decimal separator

Unbalanced Parentheses

- Consider the expression:

```
((a + b) * t / 2 * (1 – t)
```

  What is wrong with the expression?

- Now consider this expression.

```
(a + b) * t) / (2 * (1 – t)
```

  This expression has three "(" and three ")", but it still is not correct

- At any point in an expression the count of "(" must be greater than or equal to the count of ")"

- At the end of the expression the two counts must be the same

Additional Programming Tips

- Use Spaces in expressions

```
totalCans = fullCans + emptyCans
```

- Is easier to read than

```
totalCans=fullCans+emptyCans
```

- Other ways to import modules:

```
# imports the functions listed
from math import sqrt, sin, cos

# imports all functions from the module
from math import *

# imports all functions from the module
import math
```

- If you use the last style you have to add the module name and a "." before each function call

```
y = math.sqrt(x)
```

Code Academy

Combining Assignment and Arithmetic

In Python, you can combine arithmetic and assignment. For example,

```
 total += cans          #is a shortcut for
total = total + cans

 total *= 2             #is a shortcut for
total = total * 2

#Then an increment by one can be written as
 total += 1
```

Code Academy

2.3 Problem Solving

DEVELOP THE ALGORITHM FIRST, THEN WRITE THE PYTHON

Code Academy

2.3 Problem Solving: First by Hand

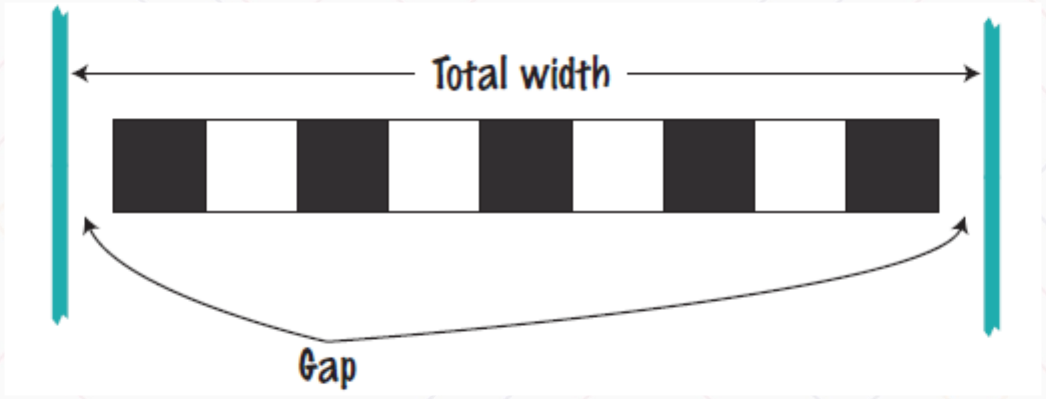
- A very important step for developing an algorithm is to first carry out the computations by hand.

  - If you can't compute a solution by hand, how do you write the program?

- **Problem:**

  A row of black and white tiles needs to be placed along a wall. For aesthetic reasons, the architect has specified that the first and last tile shall be black.

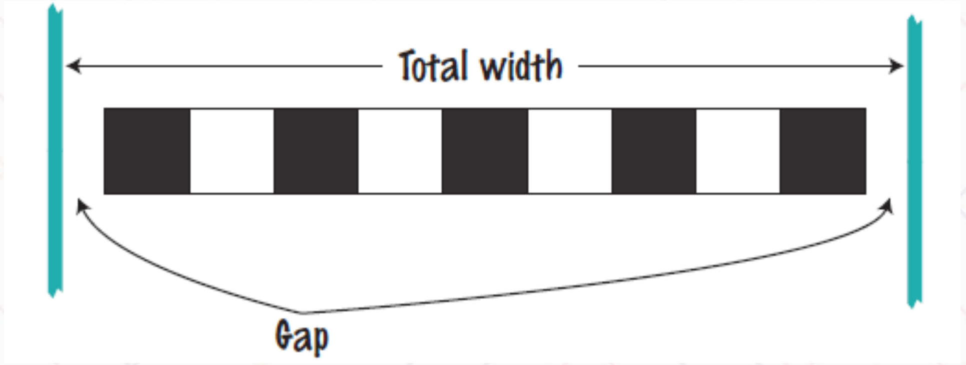
  Your task is to compute the number of tiles needed and the gap at each end, given the space available and the width of each tile.



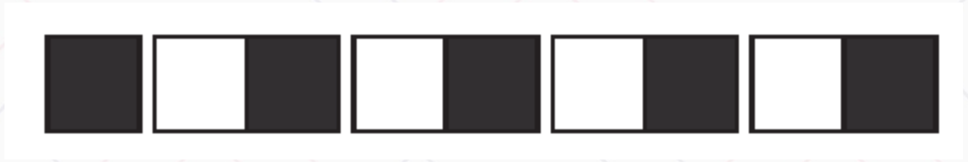
Code Academy

Start with example values

- Givens
  - Total width: 100 inches
  - Tile width: 5 inches



- Test your values
  - Let’s see… 100/5 = 20, perfect! 20 tiles. No gap.
  - But wait… BW…BW “…first and last tile shall be black.”
- Look more carefully at the problem….
  - Start with one black, then some number of WB pairs

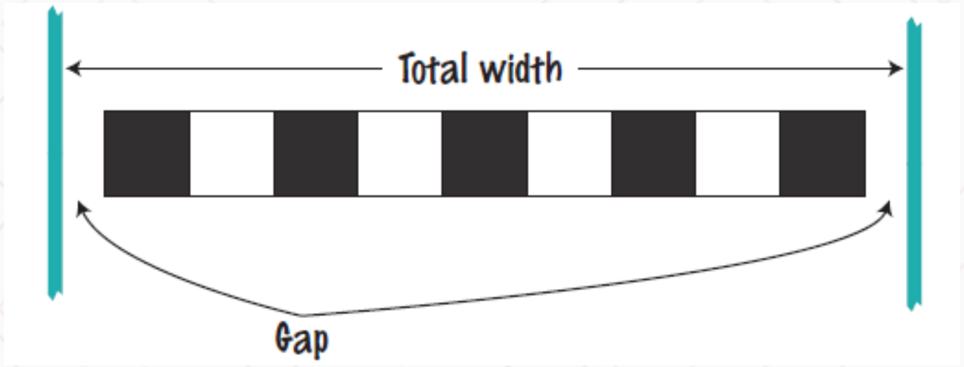


  - Observation: each pair is 2x width of 1 tile
    - In our example, 2 x 5 = 10 inches

Code Academy

Keep applying your solution

- totalWidth = 100
- tileWidth = 5



- Calculate total width of all tiles
  - One black tile: 5
  - 9 pairs of BWs: 90
  - Total tile width: 95
- Calculate gaps (one on each end)
  - 100 – 95 = 5 total gap
  - 5 gap / 2 = 2.5 at each end

Code Academy

Now devise an algorithm

- Use your example to see how you calculated values
- How many pairs?

  Note: must be a whole number

  Integer part of: (totalWidth – tileWidth) /(2 x tileWidth)

- How many tiles?

  1 + 2 x the numberOfPairs

- Gap at each end

  (totalWidth – numberOfTiles x tileWidth) / 2

Code Academy

The algorithm

1) Calculate the number of pairs of tiles
   `NumberOfPairs = int((totalWidth – tileWidth) / (2 * tileWidth))`
2) Calculate the number of tiles
   `NumberOfTiles = 1 + (2 * numberOfPairs)`
3) Calculate the gap
   `GapAtEachEnd = (totalWidth – numberOfTiles * tileWidth) / 2`
4) Print the `NumberOfPairs`
5) Print the total number of tiles in the row `NumberOfTiles`
6) Print the gap `GapAtEachEnd`

Code Academy

Summary: operators

- The assignment operator = does not denote mathematical equality.
- The / operator performs a division yielding a value that may have a fractional value.
- The // operator performs a division, the remainder is discarded.
- The % operator computes the remainder of a floor division.

Summary: python overview

- The Python library declares many mathematical functions, such as `sqrt()` and `abs()`
- Python libraries are grouped into modules. Use the `import` statement to use methods from a module.