Removing duplication is particularly important when programs are maintained for a long time. When there are two sets of statements with the same effect, it can easily happen that a programmer modifies one set but not the other.

Special Topic 3.1

Use relational

and strings.

(< <= > >= == !=) to compare numbers

operators

Conditional Expressions

Python has a conditional operator of the form

value1 if condition else value2

The value of that expression is either $value_1$ if the condition is true or $value_2$ if it is false. For example, we can compute the actual floor number as

actualFloor = floor - 1 if floor > 13 else floor which is equivalent to

if floor > 13 :
 actualFloor = floor - 1
else :
 actualFloor = floor

Note that a conditional expression is a single statement that must be contained on a single line or continued to the next line (see Special Topic 2.3). Also note that a colon is not needed because a conditional expression is not a compound statement.

You can use a conditional expression anywhere that a value is expected, for example:

print("Actual floor:", floor - 1 if floor > 13 else floor)

We don't use the conditional expression in this book, but it is a convenient construct that you will find in some Python programs.

3.2 Relational Operators

In this section, you will learn how to compare numbers and strings in Python.

Every if statement contains a condition. In many cases, the condition involves comparing two values. For example, in the previous examples we tested floor > 13. The comparison > is called a relational operator. Python has six relational operators (see Table 1).

As you can see, only two Python relational operators (> and <) look as you would expect from the mathematical notation. Computer keyboards do not have keys for \geq , \leq , or \neq , but the >=, <=, and != operators are easy to

In Python, you use a relational operator to check whether one value is greater than another.

remember because they look similar. The == operator is initially confusing to most newcomers to Python.

')

ore

on th on

of of er ne es

non,

t.

1

Table 1 Relational Operators		
Python	Math Notation	Description
>	>	Greater than
>=	≥	Greater than or equal
****** *		Less than
<=	≤	Less than or equal
osen Attacke	=	Equal
!=	≠	Not equal

In Python, = already has a meaning, namely assignment. The == operator denotes equality testing:

floor = 13 # Assign 13 to floor if floor == 13 : # Test whether floor equals 13

You must remember to use == inside tests and to use = outside tests.

Strings can also be compared using Python's relational operators. For example, to test whether two strings are equal, use the == operator

if name1 == name2 :
 print("The strings are identical.")

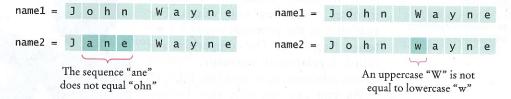
or to test if they are not equal, use the != operator

if name1 != name2 :
 print("The strings are not identical.")

For two strings to be equal, they must be of the same length and contain the same sequence of characters:

name1 = J o h n W a y n e name2 = J o h n W a y n e

If even one character is different, the two strings will not be equal:



The relational operators in Table 1 have a lower precedence than the arithmetic operators. That means you can write arithmetic expressions on either side of the relational operator without using parentheses. For example, in the expression

floor - 1 < 13

both sides (floor - 1 and 13) of the < operator are evaluated, and the results are compared. Appendix B shows a table of the Python operators and their precedences.

Table 2 Relational Operator Examples		
Expression	Value	Comment
3 <= 4	True	3 is less than 4; <= tests for "less than or equal".
3 =< 4	Error	The "less than or equal" operator is <=, not =<. The "less than" symbol comes first.
3 > 4	False	> is the opposite of <=.
4 < 4	False	The left-hand side must be strictly smaller than the right-hand side.
4 <= 4	True	Both sides are equal; <= tests for "less than or equal".
3 == 5 - 2	True	== tests for equality.
3 != 5 - 1	True	!= tests for inequality. It is true that 3 is not $5-1$.
3 = 6 / 2	Error	Use == to test for equality.
1.0 / 3.0 == 0.333333333	False	Although the values are very close to one another, they are not exactly equal. See Common Error 3.2 on page 101.
\(\sigma\) "10" > 5	Error	You cannot compare a string to a number.

Table 2 summarizes how to compare values in Python. The following program demonstrates comparisons using logical expressions.

ch03/compare.py

```
# This program demonstrates comparisons of numbers and strings.
3
5
    from math import sqrt
 6
 7
    # Comparing integers
 8
9
10
11
    if m * m == n :
12
      print("2 times 2 is four.")
13
    # Comparing floating-point numbers
15
    x = sqrt(2)
16 y = 2.0
17
18 if x * x == y:
       print("sqrt(2) times sqrt(2) is 2")
19
20
21
       print("sqrt(2) times sqrt(2) is not four but %.18f" % (x * x))
22
23 EPSILON = 1E-14
```

Sp

```
if abs(x * x - y) < EPSILON:
25
       print("sqrt(2) times sqrt(2) is approximately 2")
26
27
    # Comparing strings
    s = "120"
t = "20"
28
29
31
    if s == t :
32
       comparison = "is the same as"
33
    else:
34
       comparison = "is not the same as"
35
36
    print("The string '%s' %s the string '%s'." % (s, comparison, t))
37
38
    u = "1" + t
39
    if s != u :
40
       comparison = "not "
41
    else:
42
       comparison = ""
43
44 print("The strings '%s' and '%s' are %sidentical." % (s, u, comparison))
```

Program Run

```
2 times 2 is four.
sqrt(2) times sqrt(2) is not four but 2.00000000000000444
sqrt(2) times sqrt(2) is approximately 2
The string '120' is not the same as the string '20'.
The strings '120' and '120' are identical.
```



6. Which of the following conditions are true, provided a is 3 and b is 4?

```
a. a + 1 <= b
b. a + 1 >= b
c. a + 1 != b
```

7. Give the opposite of the condition

```
floor > 13
```

8. What is the error in this statement?

```
if scoreA = scoreB :
    print("Tie")
```

9. Supply a condition in this if statement to test whether the user entered a Y:

```
userInput = input("Enter Y to quit.")
if . . . :
    print("Goodbye")
```

- 10. How do you test that a string userInput is the empty string?
- 11. Consider the two strings

```
"This is a long string."
"This is a long string;"
Why are the two strings not equal?
```

Practice It Now you can try these exercises at the end of the chapter: R3.4, R3.7.