Chapter 4

If statements

Quite often in programs we only want to do something provided something else is true. Python's `if` statement is what we need.

4.1 A Simple Example

Let's try a guess-a-number program. The computer picks a random number, the player tries to guess, and the program tells them if they are correct. To see if the player's guess is correct, we need something new, called an *if statement*.

```
from random import randint

num = randint(1,10)
guess = eval(input('Enter your guess: '))
if guess==num:
    print('You got it!')
```

The syntax of the if statement is a lot like the `for` statement in that there is a colon at the end of the if condition and the following line or lines are indented. The lines that are indented will be executed only if the condition is true. Once the indentation is done with, the if block is concluded.

The guess-a-number game works, but it is pretty simple. If the player guesses wrong, nothing happens. We can add to the if statement as follows:

```
if guess==num:
    print('You got it!')
else:
    print('Sorry.  The number is ', num)
```

We have added an `else` statement, which is like an "otherwise."

4.2 Conditional operators

The comparison operators are ==, >, <, >=, <=, and !=. That last one is for *not equals*. Here are a few examples:

Expression	Description
`if x>3:`	if x is greater than 3
`if x>=3:`	if x is greater than or equal to 3
`if x==3:`	if x is 3
`if x!=3:`	if x is not 3

There are three additional operators used to construct more complicated conditions: `and`, `or`, and `not`. Here are some examples:

```
if grade>=80 and grade<90:
    print('Your grade is a B.')

if score>1000 or time>20:
    print('Game over.')

if not (score>1000 or time>20):
    print('Game continues.')
```

Order of operations In terms of order of operations, `and` is done before `or`, so if you have a complicated condition that contains both, you may need parentheses around the `or` condition. Think of `and` as being like multiplication and `or` as being like addition. Here is an example:

```
if (score<1000 or time>20) and turns_remaining==0:
    print('Game over.')
```

4.3 Common Mistakes

Mistake 1 The operator for equality consists of two equals signs. It is a really common error to forget one of the equals signs.

Incorrect	Correct
`if x=1:`	`if x==1:`

Mistake 2 A common mistake is to use `and` where `or` is needed or vice-versa. Consider the following if statements:

```
if x>1 and x<100:
if x>1 or x<100:
```

The first statement is the correct one. If x is any value between 1 and 100, then the statement will be true. The idea is that x has to be *both* greater than 1 *and* less than 100. On the other hand, the second statement is not what we want because for it to be true, *either* x has to be greater than 1 *or* x has to be less than 100. But every number satisfies this. The lesson here is if your program is not working correctly, check your **and**'s and **or**'s.

Mistake 3 Another very common mistake is to write something like below:

```
if grade>=80 and <90:
```

This will lead to a syntax error. We have to be explicit. The correct statement is

```
if grade>=80 and grade<90:
```

On the other hand, there is a nice shortcut that does work in Python (though not in many other programming languages):

```
if 80<=grade<90:
```

4.4 elif

A simple use of an if statement is to assign letter grades. Suppose that scores 90 and above are A's, scores in the 80s are B's, 70s are C's, 60s are D's, and anything below 60 is an F. Here is one way to do this:

```
grade = eval(input('Enter your score: '))

if grade>=90:
    print('A')
if grade>=80 and grade<90:
    print('B')
if grade>=70 and grade<80:
    print('C')
if grade>=60 and grade<70:
    print('D')
if grade<60:
    print('F')
```

The code above is pretty straightforward and it works. However, a more elegant way to do it is shown below.

```
grade = eval(input('Enter your score: '))

if grade>=90:
    print('A')
elif grade>=80:
    print('B')
elif grade>=70:
    print('C'):
```

```
elif grade>=60:
    print('D'):
else:
    print('F')
```

With the separate if statements, each condition is checked regardless of whether it really needs to be. That is, if the score is a 95, the first program will print an A but then continue on and check to see if the score is a B, C, etc., which is a bit of a waste. Using `elif`, as soon as we find where the score matches, we stop checking conditions and skip all the way to the end of the whole block of statements. An added benefit of this is that the conditions we use in the `elif` statements are simpler than in their `if` counterparts. For instance, when using `elif`, the second part of the second if statement condition, `grade<90`, becomes unnecessary because the corresponding `elif` does not have to worry about a score of 90 or above, as such a score would have already been caught by the first if statement.

You can get along just fine without `elif`, but it can often make your code simpler.

4.5 Exercises

1. Write a program that asks the user to enter a length in centimeters. If the user enters a negative length, the program should tell the user that the entry is invalid. Otherwise, the program should convert the length to inches and print out the result. There are 2.54 centimeters in an inch.

2. Ask the user for a temperature. Then ask them what units, Celsius or Fahrenheit, the temperature is in. Your program should convert the temperature to the other unit. The conversions are $F = \frac{9}{5}C + 32$ and $C = \frac{5}{9}(F - 32)$.

3. Ask the user to enter a temperature in Celsius. The program should print a message based on the temperature:

   - If the temperature is less than -273.15, print that the temperature is invalid because it is below absolute zero.
   - If it is exactly -273.15, print that the temperature is absolute 0.
   - If the temperature is between -273.15 and 0, print that the temperature is below freezing.
   - If it is 0, print that the temperature is at the freezing point.
   - If it is between 0 and 100, print that the temperature is in the normal range.
   - If it is 100, print that the temperature is at the boiling point.
   - If it is above 100, print that the temperature is above the boiling point.

4. Write a program that asks the user how many credits they have taken. If they have taken 23 or less, print that the student is a freshman. If they have taken between 24 and 53, print that they are a sophomore. The range for juniors is 54 to 83, and for seniors it is 84 and over.

5. Generate a random number between 1 and 10. Ask the user to guess the number and print a message based on whether they get it right or not.

6. A store charges $12 per item if you buy less than 10 items. If you buy between 10 and 99 items, the cost is $10 per item. If you buy 100 or more items, the cost is $7 per item. Write a program that asks the user how many items they are buying and prints the total cost.

7. Write a program that asks the user for two numbers and prints `Close` if the numbers are within .001 of each other and `Not close` otherwise.

8. A year is a leap year if it is divisible by 4, except that years divisible by 100 are not leap years unless they are also divisible by 400. Write a program that asks the user for a year and prints out whether it is a leap year or not.

9. Write a program that asks the user to enter a number and prints out all the divisors of that number. [Hint: the `%` operator is used to tell if a number is divisible by something. See Section 3.2.]

10. Write a multiplication game program for kids. The program should give the player ten randomly generated multiplication questions to do. After each, the program should tell them whether they got it right or wrong and what the correct answer is.

```
Question 1: 3 x 4 = 12
Right!
Question 2: 8 x 6 = 44
Wrong.  The answer is 48.
...
...
Question 10: 7 x 7 = 49
Right.
```

11. Write a program that asks the user for an hour between 1 and 12, asks them to enter `am` or `pm`, and asks them how many hours into the future they want to go. Print out what the hour will be that many hours into the future, printing `am` or `pm` as appropriate. An example is shown below.

```
Enter hour: 8
am (1) or pm (2)? 1
How many hours ahead? 5
New hour: 1 pm
```

12. A jar of Halloween candy contains an unknown amount of candy and if you can guess exactly how much candy is in the bowl, then you win all the candy. You ask the person in charge the following: If the candy is divided evenly among 5 people, how many pieces would be left over? The answer is 2 pieces. You then ask about dividing the candy evenly among 6 people, and the amount left over is 3 pieces. Finally, you ask about dividing the candy evenly among 7 people, and the amount left over is 2 pieces. By looking at the bowl, you can tell that there are less than 200 pieces. Write a program to determine how many pieces are in the bowl.

13. Write a program that lets the user play Rock-Paper-Scissors against the computer. There should be five rounds, and after those five rounds, your program should print out who won and lost or that there is a tie.