8 Conditionals

Key terms: `if elif else`
Reading: Severance 3
Exercise: Write a program that indicates whether the first command line argument contains any duplicate letters.
The procedural or imperative model of programming is the one most programmers already know, with looping and branching. This approach runs the risk of creating "spaghetti code" — programs where execution jumps around in complicated ways. It was thought that "structured" procedural code would resolve the problems associated with "goto" statements, but stateful nested loops can also be very confusing.
The hierarchical structure of Python code is delimited by full colons and indentation, instead of curly brackets. Code blocks are set off by a trailing full colon followed by additional indentation, and end when their indentation ends.
Consider the object produced by the expression `2>1`; it's `True`. Boolean values like that, either accessed in memory or calculated on the spot, are where programs branch into conditional blocks. Mutually exclusive branches live within the blocks of one `if`, any number of `elif`s, and possibly one `else`.

```
# Determine the speed to play a record based on its diameter:

if vinyl_diameter_inches == 7:
 rpm = 45
else:
 rpm = 100/3 # i.e., "33-1/3"
```

The order of a series of conditions matters because each one excludes the rest. We usually prefer clarity as a criterion for order over performance. However, branching conditions which are not truly mutually exclusive make the order positively matter, which can be difficult to notice. If possible, take care that your conditions be mutually exclusive, because doing so can reduce confusion.
Note that objects which are not exactly `bool`s can nevertheless have Boolean meanings for the purpose of conditional branching. With regard to the length of a string, instead of `if len(name)>0`, we can write `if len(name)`, or indeed even `if name`, to do the same job. A special operator precedence for chained comparisons allows compounds like `2005<year<2015`.

```
# Describe in English a person's age in years:

if age <= 12:
 cohort = 'child'
elif 13 <= age <= 19:
 cohort = 'teenager'
elif 20 <= age <= 69:
 cohort = 'adult'
else:
 cohort = 'senior'

print(age,cohort)
```

These `if`/`elif`/`else` branches written in procedural, block style as above require many lines of code. Inline conditional expressions can be more economical, as well as avoiding "error-prone attempts to achieve the same effect using `and` and `or`"; `PEP-308`[12] describes how the standard was adopted and which variations were considered.

```
print("profit" if roi > 0 else "loss")
```

Next, we will address the implications of Pythonic style to loops and program flow.

[12] `https://peps.python.org/pep-0308/`