

Assignment

Oliver W. Layton

CS151: Computational Thinking: Visual Media

Lecture 06, Spring 2021

Monday February 23

Warm-up: For loop review

`lecture06_for_loop_warmup.py`

Big topic for today: Assignment

Consider the code:

```
x = 10
```

In Python (and programming more generally), `x` is called a **variable**.

- It is a name of the value 10 that is **stored in memory** so we can refer to at other points in the program
- Let's look at examples in `lecture06_assignment.py`.

Variable naming rules (1/2)

1. Variables may only contain upper case letter (A through Z), lower case (a through z), decimal digits (0 through 9), and the underscore character (_).
 - Spaces aren't allowed (`my variable = 10` is NOT valid).
 - Other punctuation is not allowed (`myVariable!` and `my$var` are NOT valid).
2. A variable must start with an upper or lower case letter.
 - While upper case first letters are valid, we use by convention (readability) a lowercase starting letter for variables (e.g. `image01`, `myString`, `theHexagonShape`).

Variable naming rules (2/2)

1. There are certain **reserved words** (also called **keywords**) by Python that can't be used as variable names. These are identifiers used within Python itself and you can't use them otherwise things might get ambiguous.
 - For example, you CAN'T name a variable `for` because Python wouldn't know if you meant the variable `for` or wanted to define a for loop.
 - Other examples of reserved words include `False`, `None`, `True`, `for`, `break`, `import`, and `return`. A full list is available in [the official online Python documentation](#).

Assignment rules

When writing `numberOfGems = 10`,
the `=` sign is called the **assignment operator**.

Important: this is different than 'equals' in math!

- *Variable names always go on the left of the assignment operator, values go on the right. Let's see what happens...*
- **When you see a statement like `numberOfGems = 10`, you should say to yourself: "this is an *assignment statement*!" The variable `numberOfGems` is assigned the value **10**.**

Variables can change value

Let's look at examples in our Python file.

Variables in loops

What would the value of `gemsCollected` be after running the following:

```
gemsCollected = 0
for i in range(10):
    gemsCollected = gemsCollected + 1
```

- It's 10 because the variable starts out with a value of 0, our for loop runs for 10 iterations and within each iteration it adds 1 before replacing the existing value each time.

The loop index is a variable

- The loop index (`i`) is actually a variable.
- It takes on values from `range(10)` that happen to correspond to the loop iteration number.

- What do you think the following would print?

```
for i in range(5):
```

```
    print(i**2)
```

- You might think it prints 1, 4, 9, 16, 25, BUT it actually prints 0, 1, 4, 9, 16.
- Tip: Use lots of `print()` statements when in doubt to check your understanding!

Data types

- What are the 4 basic data types we've talked about?
- Which are assigned to the following variables?
 - `myBadVarName = True`
 - `pi = 3.1415`
 - `ten = 10`
 - `soFriendly = 'hello!'`
- A variable can only be associated with ONE type at once (unless it is reassigned).
- The operations rules that we learned apply. Let's see a few examples.

Functions and variables: Let's step thru this program

```
1 '''Convert temperature from Fahrenheit to Celsius
2 Oliver W. Layton
3 CS151 Visual Media, Spring 2021
4 '''
5
6 def convertFtoC(temp):
7     convertedTemp = (5/9)*(temp - 32)
8     print(temp, 'degrees in F is', convertedTemp, 'degrees in C.')
9
10
11 # Main code
12 tempInF1 = 32
13 convertFtoC(tempInF1)
14
15 tempInF2 = 212
16 convertFtoC(tempInF2)
17 tempInF1 = 100
```