Business Programming (using Python)

Zhaohu (Jonathan) Fan

Information Technology Management

Scheller College of Business

Georgia Institute of Technology

August 29, 2023

Main topics

- Go over some of the Severance Chapter 2 concepts (plus some)
- Values & types
  - Variables
    - Variable assignment
    - Roles that variables play
  - Type conversions
  - Expressions & order of evaluation
  - Style Guide
  - Debugging
    - Errors & Exceptions
    - Try & Except

Values & (data) types

Variables

- In Python, you must declare a **variable** and then 'bind' a value to it (assignment) for it to **store data**.
- When you declare a variable, you must also give it a name.
- Variable names in Python are **case sensitive** and **cannot start with a number** – but **can contain letters, numbers, and underscores**.
  - `bob`
  - `Bob`
  - `_bob`
  - `_2_bob_`
  - `bob_2`
  - `BoB`

Variable names and naming conventions

- Variable names and naming conventions (with a link)
- - **Lowercase**, with **words separated by underscores**
    - Snake case (stylized as `snake_case`) refers to the style of writing in which each space is replaced by an underscore `(_)` character, and the first letter of each word is written in lowercase.

Built-in simple types

Type	Example	Description
`int`	`x = 1`	integers (i.e., whole numbers)
`float`	`x = 1.0`	floating-point numbers (i.e., real numbers)
`complex`	`x = 1 + 2j`	Complex numbers (i.e., numbers with real and imaginary part)
`bool`	`x = True`	Boolean: True/False values
`str`	`x = 'abc'`	String: characters or text
`NoneType`	`x = None`	Special object indicating nulls

So, how do you go about creating a variable in Python?

Variables: assignment

- Binding a variable: setting a name to hold a reference to some object through assignment.
- When declaring a variable, the name appears on the left-side of an assignment expression.
- **Assignment using the `=`**: `number_of_dogs_I_own=4`
  - `number_of_dogs_I_own=4`
  - `number_of_cats_I_own=2`
  - `number_of_pets_I_own=number_of_dogs_I_own+number_of_cats_I_own`
- First time is called variable initialization.

What is the input?

- Getting Input from the Keyboard
  - input(prompt): To accept input from a user
  - print(): To display output on the console/screen

How input() and print() functions work

Output

```
x = int(input('How many dogs do you own?\n'))
print('I currently own ' + str(x) + ' wonderful dogs.')
```

```
How many dogs do you own?
4
I currently own 4 wonderful dogs.
```

```
type(x)
```

```
int
```

Variables: types

Output

```
x = int(input('How many dogs do you own?\n'))
print('I currently own ' + str(x) + ' wonderful dogs.')
```

```
How many dogs do you own?
4
I currently own 4 wonderful dogs.
```

```
type(x)
```

```
int
```

- e.g.: `+` means `addition` if something is an integer -- but will mean `concatenate` if something is a string

Type conversions (type casting)

- When a variable is first declared (initialized), Python looks at the data you just assigned to the variable and determines the appropriate type.
    - **Interesting question: can the type of an already-declared variable be changed?**
        - e.g. can a variable of type integer change to start holding strings?
        - Hang on.... why would you even think of doing this?

Type conversions (type casting)

Output

```
x = int(input('How many dogs do you own?\n'))
print('I currently own ' + str(x) + ' wonderful dogs.')
```

```
How many dogs do you own?
4
I currently own 4 wonderful dogs.
```

```
type(x)
```

```
int
```

- Use `type( )` to confirm the type.
  - Sometimes working with multiple data types requires type conversion.
  - In this case, let's use the built-in function `str()` for datatype conversion.

Explicit type conversions

Output

```
x = int(input('How many dogs do you own?\n'))
print('I currently own ' + str(x) + ' wonderful dogs.')
```

```
How many dogs do you own?
4
I currently own 4 wonderful dogs.
```

```
type(x)
```

```
int
```

- Converting numbers->strings `str()`
- Converting integer->float `float()`
- Converting floats -> integers `int()`

Numeric expressions

- Order of evaluation

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Power
%	Remainder (Modulo)

```
xx = 2
xx = xx + 2
print(xx)
```

```
4
```

```
yy = 440 * 12
print(yy)
```

```
5280
```

```
zz = yy / 1000
print(zz)
```

```
5.28
```

```
jj = 23
kk = jj % 5
print(kk)
```

```
3
```

```
print(4 ** 3)
```

```
64
```

Errors & Exceptions

- When facing an error message, and the error type is unclear, you should do some research on it.
    - There are two main types of errors: 1) syntax errors, and 2) exceptions.