COMP10001 Foundations of Computing
Semester 1, 2019

Tutorial Questions: Week 3

Discussion

1. What is a "data type"? Can the data type of an object change?

   A: *A data type is a classification of data which tells python how to store and process it and defines what we can do with it. Data types are permanent when you create the object, but you can convert a value into another type with the functions in the table below.*

2. As a class, fill in the below table with the data types we have studied so far. What is the difference between the second and third type, both being numerical?

Type	Example	What does it store	What can we do with it (functions, operations...)	How do we convert to it?
str	"Hello"	A sequence of characters	`len()`, `input()`, `print()`, (in future: slicing, indexing, `.lower()`)	`str()`
int	123	A whole number (integer)	Arithmetic operations, counting & numbering (in future: indexing and slicing)	`int()`
float	3.1415	A number containing a fractional part	Arithmetic operations, mathematics & real world measurements	`float()`
bool	True	A truth value (T/F)	(in future: result of truth tests, used in conditional statements)	`bool()`

   A: *Integer is a whole number with no fraction, float has a fractional part (can be .0 though)*

   Now try exercises 1 & 2

3. Why does `0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1` equal `0.999999999999999` and not `1.0`?

   A: *Floating point numbers are approximations and can store only a finite amount of precision. Sometimes the way the computer stores decimal values can lead to inaccurate results on calculation.*

4. What is an "operator"? Which operators have we learned so far and what do they do?

   A: *An operator is a symbol used between two values which calculates some result. We?ve used + (addition), − (subtraction), ∗ (multiplication), ∕ (division) as well as % (modulo: remainder when first value divided by second), ∕∕ (integer division: converts result into integer - rounds down) and ∗∗ (exponential). Also = for assignment. Note that order of operations applies in Python.*

5. What is "operator overloading"? What is the difference between using + with numerical types and strings/sequences?

   A: *Overloading is where the same operator works in slightly different ways for different data types. In this case, when used with numerical types, + is arithmetic addition and when used with strings/ sequences it is string concatenation, which means joining together.*

   Now try exercises 3 & 4

6. What is a "variable"? How do we use variables and why are they helpful?
   Bonus question: what is a literal?

   A: *A variable is a place in the computer's internal memory where a value can be stored. A name or identifier is used to access a variable. Created by declaration, value is controlled with the assignment operator (=). Useful because we can refer back to the same value more easily, using a name which we can associate with its meaning. Easy to update and use in calculations. A literal is a value typed directly into a program, rather than one referenced by a variable.*

7. How does the `input()` function work?

   A: *Takes a string prompt as argument. When run, displays that prompt to the user and lets them enter some text. That text becomes the return value of the input() function, as a string.*

   Now try Problems 1, 2 & 3

Exercises

1. Look at the following form and decide which data types (string, integer, float or boolean) should be used to store the customer's data.

```
Name: str
Customer ID: int ... or possibly str if ID not numeric
Address: str
Postcode: int or str: in some countries a postcode is alphanumeric; with a str we can test the len()
```
of the input to ensure the correct amount of characters are inserted.
```
Do you own or rent? str? But you may get "rent", "r", "Ren" and other answer variations. If
```
question were "Do you rent?" a `bool` could be appropriate as there are only two options: yes/no.
```
Length of bench top: float
Width of bench top: float
Are you interested in further offers? bool
```

2. Evaluate the following:

   (a) `str(3 + 4) + "cakes"`

   A: `'7cakes'` *(Note lack of space before "cakes" required)*

   (b) `int(5 / 2)`

   A: *2 (2.5 without int conversion, this floors the number)*

   (c) `float("357"+ "."+ "23")`

   A: *357.23*

   (d) `bool("anything")`

   A: *True (Note: from Worksheet 3. Any non-empty string will convert to True)*

3. What is the output of the following? Why?

   A: *Note: this question is a demonstration of operator overloading*

   (a) `123 + 123`

   A: *246 (Arithmetic addition)*

   (b) `"123"+ "123"`

   A: *'123123' (String concatenation)*

   (c) `"123"+ 123`

   A: *TypeError: must be `str, not int`*
   (Can't mix non-numeric types)

   (d) `3 * 4`

   A: *12 (Arithmetic multiplication)*

   (e) `"3"* 4`

   A: *'3333' (String multiplication/repetition)*

   (f) `"3"* "4"`

   A: *TypeError: can't multiply sequence by non-`int` of `type` 'str'*
   (Can't multiply two strings)

4. Evaluate the following given the assignments `a = 1, b = 2, c = 2.0`:

   A: *Note: these variable names are bad and you shouldn't use single-letter names in your code.*

   (a) `a / a`

   A: *1.0 (Note type conversion because of divison)*

   (b) `b + b`

   A: *4*

   (c) `b + c`

   A: *4.0 (Note type conversion because of float operand)*

   (d) `a / b`

   A: *0.5 (Type conversion: division)*

   (e) `a // b`

   A: *0 (Floor division converts to integer; rounds down)*

   (f) `a % b`

   A: *1*

   (g) `a + b / c`

   A: *2.0 (Order of operations: division evaluated first)*

   (h) `(a + b) / c`

   A: *1.5 (Order of operations: brackets indicate priority)*

Problems

1. Write a program which asks the user for their age and calculates the year in which they were born. There will be two possibilities since you haven't taken their birth date, so print both.

   A:

   ```python
   age = int(input("Enter␣your␣age:␣"))
   option_1 = 2019 - age
   option_2 = option_1 - 1
   print("You␣were␣born␣in␣either", option_1, "or", option_2)
   ```

2. Write a program which asks the user for two floats and multiplies them together, printing the equation in the form `1.5 * 2.0 = 3.0` for the case of 1.5 and 2.0

   A:

   ```python
   num_1 = float(input("Enter␣the␣first␣number:␣"))
   num_2 = float(input("Enter␣the␣second␣number:␣"))
   result = num_1 * num_2
   print(num_1, "*", num_2, "=", result)
   ```

3. Write a program which asks the user for a temperature in degrees Fahrenheit and prints the conversion into Celsius. The formula is below:
   $$C = \frac{F - 32}{1.8}$$

   A:

   ```python
   fahrenheit = float(input("Enter␣the␣temperature␣in␣Fahrenheit:␣"))
   celsius = (fahrenheit - 32) / 1.8
   print(fahrenheit, "Fahrenheit␣converts␣to", celsius, "Celsius")
   ```