New today! Go to **Course.Care** > **COMP110** > **Join Discussion**
If you want to ask a question in lecture from 11am EST to 12:15am EST!

Fall 2021

LDOC - Code Writing Practice

Announcements

- Final deadline on all deliverables: Tomorrow, 12/1 at 11:59pm

- Office Hours through Weds at 5pm
- Tutoring Tonight 5-7pm

- Kaki's Review Session this afternoon at 4pm in SN014

- Final Review Session Thursday at 3pm in SN014

- Apply to be a COMP110 UTA in the Spring:
  - https://bit.ly/comp110-uta-sp22

- Final Exam - In-person Friday at 8am
  - Section 001 - Chapman 211
  - Section 002 - Hamilton 100
  - Seat assignments will go on Sakai's Postem Tool by Thursday (and I will post announcement)
  - https://21f.comp110.com/resources/final.html

Code Writing Practice #1

- Write a function with the following characteristics:

- The function's name is sum_values.
- Called with an argument of type dictionary whose keys are strings and values are floats.
- Returns a float that is the sum of all values in the argument.
- You should explicitly type any variables, parameters, and return types.
- The following REPL example demonstrates usage of sum_values:

```
>>> print(sum_values({"a": 1.0, "b": 2.0, "c": 3.0}))
6.0
>>> print(sum_values({"co": 100.0, "mp": 10.0})
110.0
```

- Write a function with the following characteristics:
  - The function's name is sum_values.
  - Called with an argument of type dictionary whose keys are strings and values are floats.
  - Returns a float that is the sum of all values in the argument.
  - You should explicitly type any variables, parameters, and return types.
  - The following REPL example demonstrates usage of sum_values:

```
>>> print(sum_values({"a": 1.0, "b": 2.0, "c": 3.0}))
6.0
>>> print(sum_values({"co": 100.0, "mp": 10.0})
110.0
```

Code Writing Practice #2

- Write a class with the following characteristics:

- The class' name is Staff.
- Every Staff object has three attributes: name (string), pid (int), and is_cs (bool).
- You should be able to construct a Staff object with a constructor that has parameters to initialize each attribute
- Every Staff object should have a method named greet that takes no parameters and returns a string with the following format for Staff members whose is_cs attribute is False "Hello, I'm NAME" or "Hello, I'm NAME in CS" otherwise. Substitute NAME with the name attribute of the Staff member.
- Example usage:

```
>>> prof: Staff = Staff("Kris", 700000000, True)
>>> print(prof.greet())
Hello, I'm Kris in CS
```

Write a class with the following characteristics:

The class' name is **Staff**.

Every **Staff** object has three attributes: **name** (string), **pid** (int), and **is_cs** (bool).

You should be able to construct a Staff object with a constructor that has parameters to initialize each attribute

Every Staff object should have a method named **greet** that takes **no parameters** and **returns a string** with the following format for Staff members whose is_cs attribute is False "Hello, I'm NAME" or "Hello, I'm NAME in CS" otherwise. Substitute NAME with the name attribute of the Staff object.

- Example usage:

```
>>> prof: Staff = Staff("Kris", 700000000, True)
>>> print(prof.greet())
Hello, I'm Kris in CS
```