

COMP 110

Fall 2022

Class 02 - Functions Practice

Today's Goals

1. Announcements
2. Practice function writing
3. Practice function diagramming

Announcements

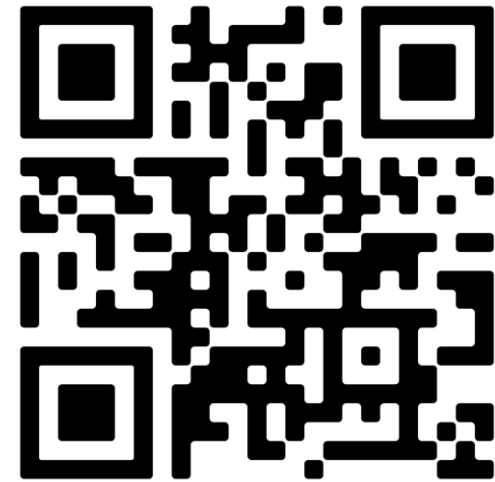
- EX03 - Released: Structured Wordle - Due Wednesday 9/14
- Tuesday 9/13: Watch the video on lists (LS14) before lecture
 - Optional Q & A regarding LS14 or earlier content held in-class on Tuesday



Pearl Hacks Committee Apps

Deadline:
Saturday, Sept. 10 by
11:59 PM

- Hacker Experience
- Beginner Experience
- Graphic Design
- External Marketing
- Sponsorship
- Promotions



<https://bit.ly/ph23committee>

Function #1

- Create a file in your `lessons` directory named `love_functions.py`
- Define a function with the following signature expectations:
 1. Function Name: `love`
 2. Parameters
 - `subject: str`
 3. Return Type: `str`
 4. Docstring: `"""Given a subject as a parameter, returns a loving string."""`
- In the function body, have a single return statement:
 - `return f"I love you {subject}!"`

Expected implementation:

```
def love(subject: str) -> str:
    """Given a subject as a parameter, returns a loving string."""
    return f"I love you {subject}!"
```

How to use in the Python REPL:

- In the terminal, begin a Python REPL:
`python`
- Import the function:
`>>> from lessons.love_functions import love`
- Call it:
`>>> love("Mom")`
`>>> love(", my dear friend")`

Function #2

- Still in the same file `lessons/love_functions.py`, declare a function named `spread_love`, with the following signature expectations:
 1. Function name: `spread_love`
 2. Two parameters:
 - `to: str`
 - `n: int`
 3. Return type: `str`
 4. Docstring: `"""Generates a str repeating a loving message n times."""`
- Implementation:
 1. Declare a string variable named `love_note` and assign it the empty string.
 2. Declare a counter variable that is initialized to zero.
 3. Write a `while` loop that will iterate while your counter variable is less than your parameter `n`. Don't forget to increment your counter variable!
 4. Inside the `while` loop, reassign `love_note` to be its current value concatenated with the result of calling the `love` function with `to` as the argument, then concatenate `"\n"` for a line break.
 5. After the while loop completes, return the generated `love_note`

Expected implementation:

```
def spread_love(to: str, n: int) -> str:
    """Generates a string that repeats a loving message n times."""
    love_note: str = ""
    i: int = 0
    while i < n:
        love_note += love(to) + "\n"
        i += 1
    return love_note
```

How to use in the Python REPL:

- In the terminal, begin a Python REPL:
`python`
- Import the function:
`>>> from lessons.love_functions import spread_love`
- Call it:
`>>> spread_love("Mom", 100)`
`>>> print(spread_love("Mom", 100))`

Challenge Question #1: What returned when the following function definition is called with...

mystery(4)

```
def mystery(n: int) -> str:
    """A useless function."""
    i: int = 0
    while i < n:
        if i % 2 == 1:
            return "ooh"
        i += 1
    return "ahh"
```

Diagramming Practice

```
1 """CQ A main Function."""
2
3
4 def main() -> None:
5     """The program's entrypoint."""
6     print("main()")
7     y: float = double(2.0)
8     print(halve(y))
9
10
11 def halve(x: float) -> float:
12     """Halve a value."""
13     print(f"halve({x})")
14     return x / 2.0
15
16
17 def double(x: float) -> float:
18     """Double a value."""
19     print(f"double({x})")
20     return x * 2.0
21
22
23 if __name__ == "__main__":
24     print("__name__ is '__main__'")
25     main()
```

```
1 """CQ A main Function."""
2
3
4 def main() -> None:
5     """The program's entrypoint."""
6     print("main()")
7     y: float = double(2.0)
8     print(halve(y))
9
10
11 def halve(x: float) -> float:
12     """Hlave a value."""
13     print(f"halve({x})")
14     return x / 2.0
15
16
17 def double(x: float) -> float:
18     """Double a value."""
19     print(f"double({x})")
20     return x * 2.0
21
22
23 if __name__ == "__main__":
24     print("__name__ is '__main__'")
25     main()
```