Functions II

Warm-up

Last class, we practiced with functional decomposition and composition. Test your knowledge by answering the following questions:

Questions (15 min)

Start time:

1. Design a program that prints a fish. Make sure to define appropriate utility functions for the distinctive features (such as the tail, body and head).

```
   vvv
    v
   vvv
  vvvvv
 vvvvvvv
vvvvvvvvv
 vvv vvv
  v   v
```

```python
def draw_tail() -> None:
    print('   vvv')
    print('    v')
def draw_body() -> None:
    print('   vvv')
    print('  vvvvv')
    print(' vvvvvvv')
    print('vvvvvvvvv')
def draw_head() -> None:
    print(' vvv vvv')
    print('  v   v')
def main() -> None:
    draw_tail()
    draw_body()
    draw_head()
main()
```

Model 1 Flow of Execution

As we saw last class, in addition to using Python's built-in functions (e.g., `print`, `abs`) and functions defined in other modules (e.g., `math.sqrt`), you can write your own functions. Consider the following program:

```python
def model_one() -> None:
    word = input("Enter a word: ")
    L = len(word)
    ans = word * L
    print(ans)

def main() -> None:
    print("Starting main...")
    model_one()
    print("All done!")

main()
```

2. What is the output of the code above if the user enters `Hi`?

Starting main...
Enter a word: Hi
HiHi
All done!

3. STOP HERE and wait for further instructions. The professor will trace the code using `Thonny`. As the professor steps through the program, pay attention to what is happening in the visualization.

a) Which button stops the execution of the code? The stop button

b) Which button starts the tracing of the code? The bug-looking one

c) Which button allows the step by step tracing of the code?

The Step in - second arrow from the left

d) Which button stops the tracing of the code and resumes the execution of the code?

The resume button to the left of stop

4. What happens when a function is called?

A new box with the function name pops up.

5. What happens when a function finishes executing?

When a function finishes executing, the pop-up box disappears and the function call is replaced with the return value.

6. Draw the tracing diagram corresponding to the code here:

main's Frame	model_one's Frame
	word -> "Hi"
	L -> 2
	ans -> "HiHi"

7. Notice that the variable `ans` is printed from within the `model_one` function. What happens if you try to `print(ans)` inside the main function?

NameError: name 'ans' is not defined

Model 2 Passing Arguments

Instead of using `input` inside a function to get data, we can define a function to take a ***parameter*** (variable). When we call the function, we need to provide an ***argument*** (value). Consider the following code:

```
1  def model_two(word: str) -> None:
2      ans = word * len(word)
3      print(ans)
4
5  def main() -> None:
6      print("Starting main...")
7      w = input("Enter a word: ")
8      model_two(w)
9      print("All done!")
10
11 main()
```

8. Draw the tracing diagram corresponding to the code here assuming the user enters `Hi`:

main's Frame	model_two's Frame
w -> "Hi"	word -> "Hi"
	ans -> "HiHi"

9. Underline the parameter in the `model_two` function definition, then circle each use of the parameter inside the function. two uses of the parameter `word` should be circled in line 2

10. Find the `model_two` function call in `main`, and underline the argument being passed by the function call. word should be underlined on line 1, and w should be underlined on line 8

11. When a variable is used as an argument, does the name of the variable need to be the same as the parameter variable name? No

12. Assume that `s1 = "Hi"` and `s2 = "ya"`. In the function call `model_two(s1 + s2)`:

a) What is the argument for the function call? s1 + s2 (the expression)

b) Write the implied assignment statement that happens during the call. word = s1 + s2

c) What will be the value of parameter `word` when `model_two` begins executing? "Hiya"

d) Predict the output that will be produced by the function call. HiyaHiyaHiyaHiya

13. Review the two implied assignment statements that you have written. What exactly gets "passed" when you call a function?

The value of the argument is passed; it gets assigned to the parameter. Note that the argument itself is not passed.

14. Change `model_two` so that, instead of multiplying `word` by the length of `word`, it will multiply by an integer passed as the second argument to the function. Write the new version of `model_two` in the space below. Use `times` for the name of the new integer parameter.

```
def model_two(word: str, times: int) -> None:
    ans = word * times
    print(ans)
```

15. How does the call to `model_two` in `main` need to change so that it matches the new function definition? Give an example.

A second argument (of type `int`) must be added to the function call: `model_two("Hi", 2)`

Model 3 Returning Values

Functions may optionally send a value back to the calling function using a `return` statement. Consider the following code:

```
def model_three(word: str) -> str:
    ans = word * len(word)
    return ans
```

```
 4
 5  def main() -> None:
 6      print("Starting main...")
 7      w = input("Enter a word: ")
 8      result = model_three(w)
 9      print(result)
10      print("All done!")
11
12  main()
```

16. Draw the tracing diagram corresponding to the code here assuming the user enters `Hi`:

main's Frame	**model_three's Frame**
w -> "Hi"	word -> "Hi"
result -> "HiHi"	ans -> "HiHi"

17. Aside from the function name, how does line 8 in this exercise differ from line 8 in the previous exercise?

It is now an assignment statement.

18. At what step number (in the simulation) has `model_three` completed its execution, but control has not yet returned to the `main` function?

Step 11.

19. In general, what value will be returned by `model_three`?

Whatever is the value of the variable `ans`.

20. What changes in the frame for `main` at Line 8 during the execution of the program?

The variable `result` has been added to the frame. It has the value returned by `model_three`.

21. What value is returned by a function when there is no return statement?

The value `None`.

22. Change line 8 so that `model_three` is still called but there is no assignment to `result`. What do you predict will happen in `main` after the `model_three` function call completes?

Because `result` is not being assigned the value returned by the `model_three` function call, trying to print the value of `result` will cause a `NameError`.

23. Why is a function that returns the value of a variable more useful than a function that simply prints the value of that variable?

Printing a value to the screen only benefits the program in that moment. If you instead return the value and assign it to a variable, you can continue using it in the calling function.

24. Redesign the following program, by moving all the provided code statements in functions for respective distinctive features and to increase re-usability (and reduce repetition). Your resulting program must also have a `main` function that is the entry point in your program (that is, it gets executed first).

```python
cube_len = int(input("Cube length? "))
box_w = int(input("Box width? "))
box_h = int(input("Box height? "))
box_l = int(input("Box length? "))

fit_w = box_w // cube_len
fit_h = box_h // cube_len
fit_l = box_l // cube_len

print(str(fit_w)+" Rubik's cubes will fit width-wise.")
print(str(fit_h)+" Rubik's cubes will fit height-wise.")
print(str(fit_l)+" Rubik's cubes will fit length-wise.")

res = fit_w * fit_h * fit_l
print(str(res)+" Rubik's cubes will fit in that container.")
```

```python
def get_fit(large: int, small: int, side: str) -> int:
    fit = large // small
    print(f"{fit} Rubik's cubes will fit {side}-wise.")
    return fit
def main() -> None:
    cube_len = int(input("Cube length? "))
    box_w = int(input("Box width? "))
    box_h = int(input("Box height? "))
    box_l = int(input("Box length? "))
    res = get_fit(box_w, cube_len, 'width') *
        get_fit(box_h, cube_len, 'height') *
        get_fit(box_l, cube_len, 'length')
    print(str(res)+" Rubik's cubes will fit in that container.")
main()
```