COMP10001 Foundations of Computing
Semester 1, 2019

Tutorial Questions: Week 5

— VERSION: 1474, DATE: MARCH 13, 2019 —

Discussion

1. What is a “function”? How do we call (use) one? How do we define one ourselves?
2. What does it mean to “return” a value from a function and why would we want to? Does a function always need a return value?
3. Why are functions so useful? Could we live without functions?
4. Why are brackets important when calling a function? Are they needed even if it takes no arguments?
5. What is a “method”? How do methods differ from functions? How are they the same?

   Now try Exercises 1 & 2

6. What is “iteration” in programming? Why do we need it?
7. What are the two types of loop in python? How do we write them?
8. What do we mean by the “loop variable” in a for loop?

   Now try Exercises 3 - 5

Exercises

1. What’s wrong with this code? How can you fix it?

```
def calc(n1, n2):
    answer = n1 + (n1 * n2)
    print(answer)

num = int(input("Enter␣the␣second␣number:␣"))
result = calc(2, num)
print("The␣result␣is:", result)
```

2. Evaluate the following method calls given the assignment `s="Computing␣is␣fun!"` Think about the input and output of each method. You’re not expected to know all the methods available to you: if you haven’t seen some of these before, have a guess at what they do and you’ll probably be right!

   (a) `s.isupper()`

   (b) `s.upper()`

   (c) `s.endswith("fun!")`

   (d) `s.count('n')`

3. What is wrong with this code? How would you fix it?

```
def largest_num(nums):
    maxnum = nums[0]
    for num in nums:
        if num > maxnum:
            maxnum = num
            return maxnum

print(largest_num([1, 2, 3]))
```

4. What is the output of the following snippets of code containing loops ?

   (a)
   ```
   for i in range(5):
       print(i**2)
   ```

   (b)
   ```
   for ingredient in ("ham", "cheese", "hollondaise", "lettuce"):
       if ingredient.startswith('h'):
           print(ingredient, "is␣delicious")
       else:
           print(ingredient, "is␣tasty")
   ```

   (c)
   ```
   i = 0
   colours = ("olive", "red", "violet", "turquoise", "red", "red", "amber")
   while i < len(colours):
       if colours[i] == "red":
           print("Found␣red␣at␣index", i)
       i += 1
   ```

5. Do the following code snippets do the same thing? What are some advantages and disadvantages of each snippet?

   ```
   print("We␣need␣some␣saws")
   print("We␣need␣some␣hammers")
   print("We␣need␣some␣cogs")
   print("We␣need␣some␣nails")
   ```

   ```
   def get_str(part):
       return f"We␣need␣some␣{part}"

   print(get_str("saws"))
   print(get_str("hammers"))
   print(get_str("cogs"))
   print(get_str("nails"))
   ```

   ```
   def get_str(part):
       return f"We␣need␣some␣{part}"

   parts = ("saws", "hammers", "cogs", "nails")

   for part in parts:
       print(get_str(part))
   ```

Problems

1. Write a function which takes an integer input `n` and prints the thirteen times tables from `1 * 13` until `n * 13`.

2. Write a function which converts a temperature between degrees Celsius and Fahrenheit. It should take a float, the temperature to convert, and a string, either `'c'` or `'f'` indicating a conversion from degrees Celsius and Fahrenheit respectively. The formulae for conversion are below.

$$C = \frac{F - 32}{1.8} \qquad F = C \times 1.8 + 32$$

3. Write a function which takes a string, finds the first vowel in it and returns the amount of times that vowel appears in it. If it's empty, return 0.