CSCI 150: Exam 2 Practice

October 6, 2021

1. Trace the following code:

```
def main1():
    a = 4
    b = 7
    s = 1

    while a < b:
        s *= a
        a += 1

    print(s)

main1()
```

2. Trace the following code:

```
def f1():
    print('Hello there')

def f2(x: int):
    if x < 2:
        f1()
        x = 7
    elif x == 7:
        f1()
        f1()
    else:
        f1()
        print('Goodbye!')
        f1()

def main2():
    f2(5)
    f2(0)


main2()
```

3. Trace the following code:

```
def top(s: str) -> bool:
    i = 0
    k = 0
    while i < len(s):
        if s[i] <= 'm':
            k +=1
        i += 1
    return k

def main3():
    print(top('exam'))
    print(top('hello'))

main3()
```

4. Write a function `largest_div` which takes in two non-zero integer parameters `a` and `b` and returns a float, which is the larger of a/b or b/a. (You can assume a and b will never be zero, so that you never get a division by zero error.)

   For example:

   - `largest_div(6,3)` should return 2.0
   - `largest_div(1,1)` should return 1.0
   - `largest_div(-5,2)` should return -0.4 (since $-0.4 > -2.5$)
   - `largest_div(-3,-7)` should return 2.3333333333333335

5. Write a function `prod` which takes in two non-zero integer parameters `a` and `b` and returns an integer which is $a \times b$. However, you must do this using only addition! (Recall that $5 \times 3 = 5 + 5 + 5$, for example.) You can assume that neither a or b will ever be zero or negative.

   For example:

   - `prod(6,3)` should return 18
   - `prod(3,6)` should return 18
   - `prod(7,4)` should return 28
   - `prod(1,1)` should return 1

6. Write a function `my_replace` which takes in three string parameters, `s`, `t`, and `u`, where `s` can be of any length and `t` and `u` are each always a single character, and returns `s`, but with each occurrence of `t` in `s` replaced with `u`. You must do this using a `while` loop, and without using the built-in Python `replace` method.

   For example:

   - `my_replace('hello','l','m')` should return `'hemmo'`
   - `my_replace('hello','h','m')` should return `'mello'`
   - `my_replace('hello','x','m')` should return `'hello'`
   - `my_replace('','l','m')` should return `''`
   - `my_replace('testing','t','q')` should return `'qesqing'`

7. Write a function `count_two` which takes in three string parameters, `s`, `t`, and `u`, where `s` can be of any length and `t` and `u` are each always a single character, and returns an integer which counts the total number of occurrences of either `t` or `u` in `s`. (You can assume that `t` and `u` will never be the same character.) Do not use the built-in method `count` in any of your code.

   For example:

- `count_two('hello','l','o')` should return 3
- `count_two('hello','e','o')` should return 2
- `count_two('hello','x','o')` should return 1
- `count_two('hello','x','k')` should return 0

8. Write a function `string_min` which takes in a string `s` of lower case characters and returns the *index* of the character closest to the front of the alphabet. You can assume that `s` will always be non-empty. If the minimal character appears more than once, the first index (i.e. lowest number) should be returned.

   For example:

   - `string_min('catalog')` should return `1` (since 'a' is the lowest character and its first occurrence is in index 1.)
   - `string_min('computer')` should return `0`
   - `string_min('string')` should return `5`
   - `string_min('movie')` should return `4`