Lab01_Guide

Table of contents

Lab01_Guide 2

Lab01_Guide

1. WWPD: Control

```
>>> def xk(c, d):
...     if c == 4:
...         return 6
...     elif d >= 4:
...         return 6 + 7 + c
...     else:
...         return 25
>>> xk(10, 10)
23

>>> xk(10, 6)
23

>>> xk(4, 6)
6

>>> xk(0, 0)
25
```

⚠ The function how_big can be very misleading. Pay attention to the frame of this function!

If you don't understand the output of the function, try to run the code!

```
>>> def how_big(x):
...     if x > 10:
...         print('huge')
...     elif x > 5:
...         return 'big'
...     if x > 0:
...         print('positive')
...     else:
...         print(0)
>>> how_big(7)          # A returned string is displayed with single
quotes
'big'

>>> print(how_big(7))   # A printed string has no quotes
big

>>> how_big(12)
huge
positive

>>> print(how_big(12))
huge
positive
None

>>> print(how_big(1), how_big(0))
positive
0
None None
```

```
>>> n = 3
>>> while n >= 0:
...     n -= 1
...     print(n)
2
1
0
-1
```

```
>>> negative = -12
>>> while negative:  # All numbers are true values except 0
...    if negative + 6:
```

```
...         print(negative)
...     negative += 3
-12
-9
-3
```

2. Debugging Quiz (Skip)

3. Falling Factorial

Use a, b = c, d to assign c to a and d to b in one line.

```python
def falling(n, k):
    total, stop = 1, n-k
    while n > stop:
        total, n = total * n, n-1
    return total
```

4. Divisible By k

Remeber to return how many numbers are printed.

```python
def divisible_by_k(n, k):
    count = 0
    i = 1
    while i <= n:
        if i % k == 0:
            print(i)
            count += 1
        i += 1
    return count
```

5. Sum Digits

```
def sum_digits(y):
    total = 0
    while y > 0:
        total, y = total + y % 10, y // 10
    return total
```

6. Syllabus Quiz (Skip)

7. WWPD: What If?

```
>>> def ab(c, d):
...     if c > 5:
...         print(c)
...     elif c > 7:
...         print(d)
...     print('foo')
>>> ab(10, 20)
10
foo
```

```
>>> def bake(cake, make):
...     if cake == 0:
...         cake = cake + 1
...         print(cake)
...     if cake == 1:
...         print(make)
...     else:
...         return cake
...     return make
>>> bake(0, 29)
1
29
```

```
>>> bake(1, "mashed potatoes")
mashed potatoes
1
'mashed potatoes'
```

8. Double Eights

1. Implement 1

Use True and False to determine whether the former number is eight.

```python
def double_eights(n):
    prev_eight = False
    while n > 0:
        last_digit = n % 10
        if last_digit == 8 and prev_eight:
            return True
        elif last_digit == 8:
            prev_eight = True
        else:
            prev_eight = False
        n = n // 10
    return False
```

2. Implement 2

```python
def double_eights(n):
    while n:
        if n % 10 == 8 and n // 10 % 10 == 8:
            return True
        n //= 10
    return False
```