Lab 1: Functions, Control **lab01.zip (lab01.zip)**

Due by 11:59pm on Wednesday, August 30.

Starter Files

Download lab01.zip (lab01.zip). Inside the archive, you will find starter files for the questions in this lab, along with a copy of the Ok (ok) autograder.

For quickly generating ok commands, you can now use the ok command generator (https://go.cs61a.org/ok-help).

Quick Logistics Review

Using Python

Using OK

Pair Programming

Topics

Consult this section if you need a refresher on the material for this lab. It's okay to skip directly to the questions and refer back here should you get stuck.

Division, Floor Div, and Modulo

Functions

Control

Error Messages

Required Questions

Getting Started Videos

Syllabus Quiz

Q1: Syllabus Quiz

Please fill out the Syllabus Quiz (https://go.cs61a.org/syllabus-quiz), which is based off of our policies found on the course syllabus (https://cs61a.org/articles/about/).

What Would Python Display? (WWPD)

Q2: WWPD: Control

Use Ok to test your knowledge with the following "What Would Python Display?" questions:

```
python3 ok -q control -u
```

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

>>> xk(10, 6)
______

>>> xk(4, 6)
______

>>> xk(0, 0)
______
```

```
>>> def how_big(x):
...     if x > 10:
...         print('huge')
...     elif x > 5:
...         return 'big'
...     elif x > 0:
...         print('small')
...     else:
...         print("nothing")
>>> how_big(7)
______

>>> how_big(12)
______

>>> how_big(1)
______

>>> how_big(-1)
______
```

Hint: Make sure your `while` loop conditions eventually evaluate to a false value, or they'll never stop! Typing `Ctrl-C` will stop infinite loops in the interpreter.

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

```
>>> positive = 28
>>> while positive:
...    print("positive?")
...    positive -= 3
------
```

```
>>> negative = -12
>>> while negative:
...    if negative + 6:
...        print(negative)
...    negative += 3
------
```

Q3: Debugging Quiz

The following is a quick quiz on different debugging techniques that will be helpful for you to use in this class. You can refer to the debugging article (/articles/debugging/) to answer the questions.

Use Ok to test your understanding:

```
python3 ok -q debugging-quiz -u
```

Code Writing Questions

Q4: Falling Factorial

Let's write a function `falling`, which is a "falling" factorial that takes two arguments, `n` and `k`, and returns the product of `k` consecutive numbers, starting from `n` and working downwards. When `k` is 0, the function should return 1.

```
def falling(n, k):
    """Compute the falling factorial of n to depth k.

    >>> falling(6, 3)  # 6 * 5 * 4
    120
    >>> falling(4, 3)  # 4 * 3 * 2
    24
    >>> falling(4, 1)  # 4
    4
    >>> falling(4, 0)
    1
    """
    "*** YOUR CODE HERE ***"
```

Use Ok to test your code:

```
python3 ok -q falling
```

Q5: Divisible By k

Write a function `divisible_by_k` that takes positive integers `n` and `k`. It prints all positive integers less than or equal to `n` that are divisible by `k` from smallest to largest. Then, it returns how many numbers were printed.

```
def divisible_by_k(n, k):
    """
    >>> a = divisible_by_k(10, 2)  # 2, 4, 6, 8, and 10 are divisible by 2
    2
    4
    6
    8
    10
    >>> a
    5
    >>> b = divisible_by_k(3, 1)  # 1, 2, and 3 are divisible by 1
    1
    2
    3
    >>> b
    3
    >>> c = divisible_by_k(6, 7)  # There are no integers up to 6 divisible by 7
    >>> c
    0
    """
    "*** YOUR CODE HERE ***"
```

Use Ok to test your code:

```
python3 ok -q divisible_by_k
```

Q6: Sum Digits

Write a function that takes in a nonnegative integer and sums its digits. (Using floor division and modulo might be helpful here!)

```
def sum_digits(y):
    """Sum all the digits of y.

    >>> sum_digits(10) # 1 + 0 = 1
    1
    >>> sum_digits(4224) # 4 + 2 + 2 + 4 = 12
    12
    >>> sum_digits(1234567890)
    45
    >>> a = sum_digits(123) # make sure that you are using return rather than print
    >>> a
    6
    """
    "*** YOUR CODE HERE ***"
```

Use Ok to test your code:

```
python3 ok -q sum_digits
```

Check Your Score Locally

You can locally check your score on each question of this assignment by running

```
python3 ok --score
```

This does NOT submit the assignment! When you are satisfied with your score, submit the assignment to Gradescope to receive credit for it.

Submit

Make sure to submit this assignment by uploading any files you've edited **to the appropriate Gradescope assignment.** For a refresher on how to do this, refer to Lab 00 (https://cs61a.org/lab/lab00/#submit-with-gradescope).

Optional Questions

These questions are optional, but you must complete them in order to be checked off before the end of the lab period. They are also useful practice!

Q7: WWPD: What If?

Use Ok to test your knowledge with the following "What Would Python Display?" questions:

```
python3 ok -q if-statements -u
```

Hint: `print` (unlike `return`) does *not* cause the function to exit.

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

```
>>> 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)
------

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

Q8: Double Eights

Write a function that takes in a number and determines if the digits contain two adjacent 8s.

```
def double_eights(n):
    """Return true if n has two eights in a row.
    >>> double_eights(8)
    False
    >>> double_eights(88)
    True
    >>> double_eights(2882)
    True
    >>> double_eights(880088)
    True
    >>> double_eights(12345)
    False
    >>> double_eights(80808080)
    False
    """
    "*** YOUR CODE HERE ***"
```

Use Ok to test your code:

```
python3 ok -q double_eights
```