15-112 F21 Practice Quiz 1

25 minutes

Note: This practice quiz was written by TAs. It may not reflect the exact difficulty or length of the actual quiz.

1. **Short Answer & Multiple Choice**

True or False: If you do not put a return statement in a function, that function will automatically return `None`.

True or False: If you define a variable inside a function, you can access that variable anywhere from the same Python file.

What will happen when `f()` is called?

```
def f():
    n = 0
    k = 2
    return n != 0 and k / n
```

a. A runtime error
b. The function will return `None`
c. It will short-circuit and return `False`
d. Nothing

What does `type(5/3)` evaluate to?

a. `float`
b. `type`
c. `int`
d. `None`

What is the value of `x` if `x = 5 % 3 ** 2 + 4 // 2`?

2. **Code Tracing**: Indicate what these print. Place your answers (and nothing else) in the boxes below the code.

```
def ct1(x, y):
    if x % 3 == 0 and (y % 2 == 0 or x/0 == 1):
        print(x + y)
        if y > 4 or x == 4:
            print(y - x)
            return False
        else:
            print(x*y)
            return True
    else:
        print(x / y)
        return False
print(ct1(6, 8) or ct1(5, 1))
```

```
def ct2(n):
    x = 112
    if (2 * n <= 15):
        x = n // 2
    if(1 / 10 + 2 / 10 != 3 / 10):
        print("pass")
    print(min(x, n), n // x, pow(x, max(n, x)))
print(ct2(5))
```

```
def f(x):
    return 3*x - 2
def g(x):
    print(x)
    return f(x + 5)
def ct3(x):
    print(f(x-2))
    x -= 2
    print(g(x))
    x %= 4
    print((g(x) % 6) // 2)
print(ct3(4))
```

3. **Free Response:** `isSevenish(n)`

   Write the function `isSevenish(n)` that takes in any Python value `n` and returns `True` if `n` is a (possibly-negative) four-digit integer and the product of the digits of `n` is within 2 of a multiple of 7. In all other cases, the function should return `False`.

   For example:

```
assert(isSevenish(1357) == True) # 1*3*5*7 is a multiple of 7
assert(isSevenish(5312) == True) # 5*3*1*2 = 30, which is 2 below 28
assert(isSevenish(1315) == True) # 1*3*1*5 = 15, which is 1 above 14
assert(isSevenish(-2222) == True) # 2*2*2*2 = 16, which is 2 above 14
assert(isSevenish(1234) == False) # 1*2*3*4 = 24 is 3 above 21
                                  # and 4 below 28
assert(isSevenish(777) == False) # 777 is not a 4 digit number
assert(isSevenish("sign up 4 puzzle hunt") == False) # not an integer
```

4. **Free Response:** `checkIntersect(m0, b0, m1, b1, cx, cy)`

   Write the function `checkIntersect(m0, b0, m1, b1, cx, cy)` that takes in one line with slope `m0` and y-intercept `b0`, another line with slope `m1` and y-intercept `b1`, and the coordinates `cx` and `cy`. In other words, we are given the following system of equations:

$$\begin{cases} y = m0 * x + b0 \\ y = m1 * x + b1 \end{cases}$$

   You may assume that the function is always provided with integer inputs for `m0`, `y0`, `m1`, and `y1`. However, you may not assume this is always the case for `cx` and `cy`.

   If `cx` and `cy` are both integers, the function should return `True` if the lines intersect at exactly `(cx, cy)` and `False` otherwise.

   If `cx` and `cy` are not both integers, the function should instead return `True` if the lines intersect at **any point** and `False` otherwise.

   For example:

```
assert(checkIntersect(1, 3, -1, 7, 2, 5) == True)
assert(checkIntersect(1, 3, -3, 7, 1, 4) == True)
assert(checkIntersect(0, 3, 1, 4, 1, 4) == False)
assert(checkIntersect(1, 3, 1, 4, 1, 4) == False) # parallel lines
assert(checkIntersect(1, 2, 3, 4, "beep", "boop") == True)
assert(checkIntersect(1, 2, 1, 4, "mike", "koz") == False)
```

   Hint: What if the two lines are parallel?