fullName:________________________ andrewID:_________________ section:___

15-112 F25
Quiz1 version B

You **must write your name on this paper and hand this back** in immediately after the assessment. If we do not receive it immediately, you will receive a zero on the assessment. **Do not unstaple any pages**. All pages must be handed in intact.

Do not use your own scrap paper. You should not need it, but if you must absolutely have scrap paper, raise your hand and we will provide some. Write your andrewID clearly on it and hand it in with your quiz. We will not grade anything on scrap paper.

You may not ask questions during the quiz, except for English-language clarification questions. If you are unsure about a problem, take your best guess.

Before and during the quiz, you may not view any other notes, prior work, websites or resources, including any form of AI. You may not use calculators, phones, laptops, or any other devices. You may not communicate with anyone else except for current 112 TAs or faculty during the assessment. All syllabus policies apply.

You may not discuss this test with anyone else, even briefly, in any form, until we have released grades. Failure to abide by these rules may result in an academic integrity violation.

Do not use strings, loops, sets, dictionaries, recursion, or anything we have not yet covered in class or the notes. Do not hardcode your answers. Assume almostEqual(x, y) and rounded(n) are both supplied for you. You must write all other helper functions you wish to use unless we specify otherwise.

Do not open this or look inside (even briefly) before the instructors signal for you to begin. When the instructors indicate that time is up, you must *immediately* stop writing, close this document, and hand it in. Do not look at anyone else's quiz.

Code Tracing

CT1[10 pts]:

Indicate what the following code prints. Place your answer (and nothing else) in the box below. If a line of code crashes, just print "crash" (without quotes) and stop the CT at that point.

```
def ct1(x, y):
    if x == 3:
        if y < 2:
            print('A')
        else:
            print('B')
    if y < 2:
        print('C')
    elif x < y:
        print('D')
    else:
        print('E')
    print('F')

print(ct1(3, -4))
print(ct1(-3, 4))
```

CT2[10 pts]:
Indicate what the following code prints. Place your answer (and nothing else) in the box below. If a line of code crashes, just print "crash" (without quotes) and stop the CT at that point.

```
import math

def ct2(x):
    y = int(int(x) * x)
    y /= 10
    print(math.floor(y) % 10)
    print(abs(math.ceil(-y)))
    return type(y)

print(ct2('3') != type('3.14'))
```

CT3[10 pts]:

Indicate what the following code prints. Place your answer (and nothing else) in the box below. If a line of code crashes, just print "crash" (without quotes) and stop the CT at that point.

```
def ct3(x, y):
    print(x%y, y%x)
    print(x//y, y//x)
    y **= 2
    print(y, float(y))
    return 8+5*2**3-2*4

print(ct3(13,3))
```

CT4[10 pts]:

Indicate what the following code prints. Place your answer (and nothing else) in the box below. If a line of code crashes, just print "crash" (without quotes) and stop the CT at that point.

```
def f1(x, d):
    return (x > 0) or (x/d < 0)

def f2(x, d):
    y = int(x > 0)
    z = int(x < 0)
    print(10*y + z) # note: this is print, not return

def f3(x, d):
    return int((x > 0) and (x/d < 0))

def ct4(x, f, g, h):
    print(f(x, 0))
    print(g(x, 0))
    print(h(x, 0))
    return 42
print(ct4(1, f1, f2, f3))
```

Free Response

Your functions should work generally for the kinds of inputs specified in the problem statement, and we may test your code using additional test cases. We will manually grade free responses for partial credit if you do not pass all the test cases.

FR1[30pts]: isGrowish(v)

We will say that a value v is "growish" (a coined term) if:

- v is an integer (positive or negative)
- v has either 2 or 3 digits
- the digits in v strictly increase from left to right

For example:

- 178 is growish because it has 3 digits and 1 < 7 < 8.
- -781 is not growish because 1 < 8.
- 1234 is not growish because it has 4 digits.

With that, write the function `isGrowish(v)` that takes a value v that can be of any type, and returns `True` if v is growish and `False` otherwise.

Here are some test cases for you:

```
    assert(isGrowish(12) == True)
    assert(isGrowish(78) == True)
    assert(isGrowish(178) == True)
    assert(isGrowish(-178) == True)
    assert(isGrowish(11) == False) # not growing
    assert(isGrowish(781) == False) # not growing
    assert(isGrowish(9) == False) # less than 2 digits
    assert(isGrowish(1234) == False) # more than 3 digits
    assert(isGrowish(-1234) == False) # more than 3 digits
    assert(isGrowish('yikes') == False) # not an int
    assert(isGrowish(178.0) == False) # not an int
```

Write your answer on the following page

FR2[30pts]: firstKDigits(n, k)

Write the function `firstKDigits(n, k)` that takes a possibly-negative int n and a positive int k, and returns the first (that is, leftmost) k digits in n. If n has fewer than k digits, just return the digits in n.

Here are some test cases for you:

```
    assert(firstKDigits(357, 1) == 3)
    assert(firstKDigits(357, 2) == 35)
    assert(firstKDigits(357, 3) == 357)
    assert(firstKDigits(357, 4) == 357)
    assert(firstKDigits(-357, 1) == 3)
```

Write your answer here or on the following page

BonusCT

These CTs are optional, and intended to be very challenging. They are worth very few points.
Indicate what the following code prints. Place your answers (and nothing else) in the boxes below. If a line of code crashes, just print "crash" (without quotes) and stop the CT at that point.

bonusCT1[1pt]:

```
def f(x, d):
    return d if x == (round(x**(1/d)) ** d) else 0
def bonusCt1(x):
    return f(x, 4) or f(x, 3) or f(x, 2)
print(bonusCt1(16))
print(bonusCt1(20))
print(bonusCt1(25))
print(bonusCt1(27))
```

bonusCT2[1pt]:

```
def bonusCt2(n):
    def f(n, k, d):
        e = int((n - int(n))*10**k) % 10
        return e if e > d else d
    z = f(n, 1, 0)
    z = f(n, 2, z)
    z = f(n, 3, z)
    z = f(n, 4, z)
    return z
print(bonusCt2(823.547109))
```