15-112 F22

Quiz1 version A

You **MUST** stop writing and hand in this **entire** quiz when instructed in lecture.

- You may not unstaple any pages.
- Failure to hand in an intact quiz will be considered cheating. Discussing the quiz with anyone in any way, even briefly, is cheating. (You may discuss it only once the quiz has been posted to the course website.)
- You may not use your own scrap paper. If you must use additional scrap paper, raise your hand and we will provide some. You must hand any scrap paper in with your paper quiz, and we will not grade it.
- You may not ask questions during the quiz, except for English-language clarifications. If you are unsure how to interpret a problem, take your best guess.
- You may not use any concepts (including builtin functions) we have not covered in the notes this semester.
- You may not use strings, loops, lists, indexing, tuples, dictionaries, sets, or recursion.
- Remember the word 'shoe.' It is your secret word to use at the end of the quiz.
- We may test your code using additional test cases. Do not hardcode.
- Assume almostEqual(x, y) and roundHalfUp(n) are both supplied for you. You must write all other helper functions you wish to use.

True or False [4pts ea]

Fill in the bubble indicating whether each of the following claims are True or False.

TF1:

○ True   ○ False   The following line of code will crash:

```
print(f"x = {1 + 3}")
```

TF2:

○ True   ○ False   The following line of code prints 11:

```
print(1+3**2+1)
```

TF3:

○ True   ○ False   The three basic error types are runtime, logical, and syntax.

TF4:

○ True   ○ False   Assuming almostEqual is properly defined as in the course notes, the following line prints True:

```
print(almostEqual(4//3, 1.0))
```

CT1: Code Tracing [12pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below.

```python
def ct1(m):
    a = m % 10
    b = m // 100
    c = a > b
    if type(c) == int:
            print(f"c = {c}")
    elif type(c) == bool:
            if b < 5:
                b += 1
            print(f'b = {b}')
    if c == False:
            print(f'a * b = {a * b}')
    else:
            print(f'a + b = {a + b}')
    return b + a


print(ct1(123))
```

CT2: Code Tracing [12pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below.

```python
def f(x):
    print(x)
    x = 2 * x
    return x + 1

def g(x):
    print(x)
    return f(x + 4)

def ct2(x):
    x = x // 2
    y = g(x)
    print(y)
    return y % (x + 3)


print(ct2(5) + 1)
```

Free Response 1: isSmallPal(n) [60pts]

We will say an integer is a "small palindrome" (a coined term) if it is exactly 4 digits long (ignoring leading 0's) and also a palindrome. A number is a palindrome if its value is the same when its digits are reversed.

Here are some examples of small palindromes:

```
1221
3883
9999
-4224   #Negative numbers are ok
```

And these values are not small palindromes:

```
1234  # not a palindrome
1212  # not a palindrome
333   # not enough digits
66666 # too many digits
1.221 # not an int
'wow' # also not an int
```

With this in mind, and without using strings, loops, lists, or other prohibited concepts or functions which are not in the week 1 notes, write the function isSmallPal(n) that takes a value of any type and returns True if n is a small palindrome, and false otherwise.

You may begin or continue your FR1 answer here, if you wish

Secret word [1pt bonus]

If you know your secret word, write it below for 1 bonus point.

_____

bonusCT: Code Tracing [2pts bonus]

This question is optional. Indicate what the following code prints. Place your answers (and nothing else) in the box below. Assume roundHalfUp(n) has been defined.

```python
def q(n):
    if n == 0:
        return 1
    else:
        return roundHalfUp(n/abs(n))

def r(m, n):
    return (q(m**11) +
            q(n**12) +
            q((m+n)**13))

def bonusCt1(x, y, z):
    return (100 * r(x, y) +
            10 * r(x, z) +
            r(y, z))

print(bonusCt1(235, -104, -417))
```