CIS 210 Fall 2019 Midterm Example Questions  KEY

Note: These questions are not a comprehensive study guide! They are given here to provide a sense of the types of questions that may be on the CIS 210 midterm exam.

To prepare thoroughly for the midterm exam you should review projects and project solutions, quizzes, class notes, labs, and readings from the text.

The midterm exam will be given in-class, and will comprise multiple choice questions, short-answer questions, and questions where the solution will require you to write Python code according to the usual CIS 210 style guidelines.

No outside resources are allowed during the exam, with the exception of one 3x5” index card of handwritten notes.

```python
>>> isinstance(101, float) == True  #booleans, built-in func, type
False

>>> isinstance(101, float)  #coding style
False
```

What will be printed when the following Python code is executed?

```python
n = 5
mysum = 0
for ctr in range(1, n):
    mysum = mysum + ctr
print(mysum)
10
```

This code is an example of # accumulator pattern

a) accumulator pattern  b) TypeError  c) conditional
d) indefinite iteration  e) infinite loop

Given the following Python code: # order of operations, type

```python
1 - >>> ftemp = 212
2 - >>> ctemp = (ftemp - 32) * 5/9
3 - >>> ctemp = ftemp - 32 * 5/9
```

The value of ctemp will [??] from line 2 to line 3; the type of ctemp will [??] from line 2 to line 3

a) stay the same/change  b) change/stay the same  c) stay the same/stay the same
d) change/change
Given the following Python code:

```python
import math

def isInCircle(x, y, r):
    '''(number, number, number) -> ??
    Returns True if point (x, y) is in the circle with radius r.
    >>> isInCircle(0, 0, 1)
    True
    >>> isInCircle(.5, .5, 1)
    True
    >>> isInCircle(1, 2, 1)
    False
    '''
    d = math.sqrt(x**2 + y**2)
    isIn = d <= r
    return isIn
```

Complete the type contract: `bool`

Which code would give the same results as `isInCircle` lines 12-14 (changes are in bold)?

a) `d = math.sqrt(x**2 + y**2)
return d = r`

b) `d = math.sqrt(pow(x, 2) + pow(y, 2))
return d <= r`

c) `d = math.sqrt(x**2 + y**2)
return d < r`

d) `d = math.sqrt(pow(x, 2) + pow(y, 2))
isIn = d < r
return isIn`

e) `d = math.sqrt(x**2 + y**2)
return d`

**Computational Problem Solving** - Replace `??` with the correct response.

**TASK/PROBLEM → Computational Thinking ↔ SOLUTION/ALGORITHM/COMPUTATIONAL PROCESS**

**ALGORITHM ↔ Design/Coding ↔ COMPUTER PROGRAM**

**COMPUTER PROGRAM ↔ ?? ↔ HIGH QUALITY (RELIABLE/REUSABLE) COMPUTER PROGRAM**

a) looping         b) executing         c) printing         d) testing and debugging
Given the following Python code:

```python
def q29(s1):  
    '''(str) -> str
    s2 = ''
    for ch in s1:
        if ch not in s2:
            s2 += ch
    return s2
```

Which brief description is appropriate for `q29`?

a) copies s1 to s2; returns s2

b) copies all characters except the last character in s1 to s2; returns s2
c) copies 1st occurrence of each character in s1 to s2; returns s2
d) determines whether s1 is an empty string
e) creates and returns s2, a string of the characters that repeat (occur more than once) in s1

Given:

```python
def q1(astr):
    '''(str) -> ??'''
    slen = len(astr)
    ctr = 0
    for ch in astr:
        ctr += 1
    return slen == ctr
```

```python
astr = 'CIS 2xx CIS 3xx'
>>> q1(astr)
True
```
Given the following Python code:

def fizzbuzz(n):
    '''(int) -> None

    Play fizzbuzz up to n.
    Results are printed during play;
    None value is returned
    '''
    for i in range(1, n+1):
        m3 = (i % 3) == 0
        m5 = (i % 5) == 0

        if m3 and m5:
            print('fizzbuzz')
        elif m3:
            print('fizz')
        elif m5:
            print('buzz')
        else:
            print(i)

    print('Game over!')

    return None

for is a Python

- a) keyword  b) identifier  c) primitive element  d) namespace  e) library module

i is a Python

- a) keyword  b) identifier  c) primitive element  d) namespace  e) library module

m3 is a Python

- a) keyword  b) identifier  c) primitive element  d) namespace  e) library module

3 is a Python

- a) keyword  b) identifier  c) primitive element  d) namespace  e) library module

if is a Python

- a) keyword  b) identifier  c) primitive element  d) namespace  e) library module

'fizzbuzz' is a Python

- a) keyword  b) identifier  c) primitive element  d) namespace  e) library module
Given the following Python code:

```python
def q24(s):
    '''
    (??) -> ??
    Test function.
    >>> q24('The quick brown fox')
    ??
    >>> q24('Hello, world.')
    ??
    '''
    result = 999
    for i in range(len(s)):
        if s[i] == 'E' or s[i] == 'e':
            result = i
    return result
q24('Hello')
```

Complete the type contract for `q24`:

`(str) -> int`

Executing this function will

a) Return the number of occurrences of 'e' in s, or 999 if none.
b) Return the number of occurrences of 'E' in s, or 999 if none.
c) Return the sum of a) and b), or 999 if none.
d) Return the position of the first occurrence of 'e' or 'E' in s, or 999 if none.
e) Return the position of the last occurrence of 'e' or 'E' in s, or 999 if none.

The first time the for loop executes, the value of `i` is

a)'H'   b)0   c)1   d)4   e)5

The first time the for loop executes, the value of `s[i] == 'E' or s[i] == 'e'` is

a)'E'   b)'e'   c)True   d)False   e) 'False'

To determine this value, Python evaluated

a)    b)    c)    d)
`s[i] == 'E'    s[i] == 'E'    s[i] == 'e'    result += 1
`s[i] == 'e'`
Given the following Python code:

```python
def q30(score):
    ''' exam function '''

    gradepoint = 0
    if score >= 90:
        gradepoint = 4
    if score >= 80:
        gradepoint = 3
    if score >= 70:
        gradepoint = 2
    if score >= 60:
        gradepoint = 1

    return gradepoint
```

What is the result of executing >>> q30(80)?

a) 4  b) 3  c) 2  d) 1  e) NameError

Given the following UNTESTED Python code:

```python
def q3(myStr):
    '''final exam function'''

    newStr = ''
    for ch in myStr:
        if ch not in newStr:
            newStr += ch

    return newStr
```

What will be the result of executing

```python
>>> q3('abab')
```

a) 'abab'  b) 'ab'  c) 'ba'  d) 'a'  e) 'b'
Given the following Python code:

```python
# variable scope, namespaces

def isOdd(i):
    '''(int) -> bool
    exercise
    '''
    return i % 2 != 0

def q8(msg):
    '''(str) -> ??
    exercise
    '''
    odd_ct = 0
    for ch in msg:
        if isOdd(int(ch)):
            odd_ct += 1
    return odd_ct

def main():
    '''exercise'''
    code = '001100001100'
    print(q8(code))
    return None

Complete the type contract for q8:

```
text
```

What will be the result of executing
```python
>>> main()
```

What would be the result of executing
```python
print(msg) between lines 12 and 13?
```
a) NameError  b) 0  c) '001100001100'  d) str

```python
print(ch) between lines 14 and 15 the first time the for loop is executed?
```
a) '0'  b) '1'  c) 'm'  d) 0  e) 1

```python
print(i) between lines 5 and 6 the first time isOdd is executed?
```
a) '0'  b) '1'  c) 'm'  d) 0  e) 1
print(odd_ct) between lines 5 and 6 the first time isOdd is executed?

a) '0'  b) '1'  c) NameError  d) 0  e) 1

print(code) between lines 24 and 25 when main is executed?

a) 0  b) 4  c) 8  d) NameError  e) '01100001100'

print(msg) between lines 24 and 25 when main is executed?

a) 0  b) 4  c) 8  d) NameError  e) '01100001100'

While function q8 is executing, odd_ct exists in a/the _____ namespace.

a) local  b) global ('__main__')  c) built-in

Functions main, q8, and isOdd exist in a/the _____ namespace.

a) local  b) global ('__main__')  c) built-in

Given:

def q3(x, y):
    '''(int, int) -> None'''
    x = f(x, y)
    y = f(y, x)
    print(x, y)
    return None

def f(x, y):
    '''(int, int) -> int'''
    x = 2 * x
    y = 2 * y
    if y > x:
        return y - x
    else:
        return x - y

>>> q3(20, 5)
30 50
Basic code tracing - what happens when a function is executed. Accumulator pattern.

Given function `approx_sqrt`:

```python
def approx_sqrt(num, iterations):
    '''(number, int) -> float

    Generates an approximate square root of num, a positive integer, via an iterative process that runs iterations times. The approximate square root is returned.
    
    >>> approx_sqrt(1, 1)
    1.0
    >>> approx_sqrt(4, 1)
    ??-1
    >>> approx_sqrt(4, 5)
    2.000000000000002
    '''
    value = 1
    for ctr in range(iterations):
        value = .5 * (value + num/value)
    return ??-2
```

Replace ??-1 with the correct code:

a) 4  b) 1  c) 2  d) 2.0  e) 2.5

Replace ??-2 with the correct code:

a) num  b) value  c) 2  d) 2.0  e) 2.5

When `>>> approx_sqrt(4, 5)` is executed, the last value of `ctr` is

a) 0  b) 1  c) 4  d) 5  e) 2.000000000000002

`approx_sqrt` is an example of

a) recursion  b) Monte Carlo algorithm  c) accumulator pattern

d) encryption algorithm  e) REPL
**Debugging; test cases that generate different results are needed.**

Given the following UNTESTED Python code:

```python
def charCt(s, c):
    '''(str, str) -> int

    Return count of occurrences of char c in string s.
    >>> charCt('hello, world', 'o')
    2
    '''
    ct = 0
    for ch in s:
        if ch == c:
            ct += 1
    return ct
```

The set of test cases that will **not** find the bug in `charCt` is:

- a) `charCt('','','a')`
- b) `charCt('abc','a')`
- c) `charCt('abc','a')`
- d) `charCt('abc','b')`
- e) `charCt('ghi','x')`
- f) `charCt('x','x')`

**Python conditional statements**

Given the following Python code:

```python
def q30(score):
    ''' exam function '''
    gradepoint = 0
    if score >= 90:
        gradepoint = 4
    elif score >= 80:
        gradepoint = 3
    elif score >= 70:
        gradepoint = 2
    elif score >= 60:
        gradepoint = 1
    return gradepoint
```

What is the result of executing >>> q30(80)?

- a) 4
- b) 3
- c) 2
- d) 1
- e) NameError
Variable scope; local-global-built-in; lexical scope; be aware of what functions return.

Given the following Python code:

```python
def taxable(inc, exempt, STD_E, STD_D):
    '''(number, int, number, number)

    Adjust gross income (inc) to taxable income
    by applying standard deduction and exemptions.

    CALLED BY: est_tax
    
    >>> taxable(20000, 1, 4150, 6500)
    9350
    '''
    #print(income)
    #print(salary)
    taxable_income = inc - STD_D
    exempt_adjust = STD_E * exempt
    taxable_income = taxable_income - exempt_adjust
    return(taxable_income)

def est_tax(income, exemptions):
    '''(number, int) -> None

    Generates an estimate for federal income tax.

    CALLS: taxable
    
    >>> est_tax(20000, 1)
    1870.0
    '''
    STD_EXEMPT = 4150
    STD_DEDUCT = 6500
    TAX_RATE = .20
    taxable_income = taxable(income,exemptions,STD_EXEMPT,STD_DEDUCT)
    estimated_tax = taxable_income * TAX_RATE
    #print('Estimated tax is:', estimated_tax)
    return None

def main(salary, exemptions):
    '''driver for estimated tax functions'''
    result = est_tax(salary, exemptions)
    print(result)
    print(salary)
    print(taxable_income)
    return None
```

1   def main(salary, exemptions):
2      '''driver for estimated tax functions'''
3      result = est_tax(salary, exemptions)
4      print(result)
5      print(salary)
6      print(taxable_income)
7      return None
salary = 20000
exemptions = 1
main(salary, exemptions)

After line 4 in main is executed, what will be printed?

a) 1870.0       b) 20000       c) None       d) NameError

After line 5 in main is executed, what will be printed?

a) 1870.0       b) 20000       c) None       d) NameError

After line 6 in main is executed, what will be printed?

a) 1870.0       b) 20000       c) None       d) NameError

If the #print(income) line of code in taxable were executed, what would be printed?

a) 1870.0       b) 20000       c) None       d) NameError

If the #print(salary) line of code in taxable were executed, what would be printed?

a) 1870.0       b) 20000       c) None       d) NameError

When function taxable is called from est_tax, which of the following happens:

a) an activation record for function taxable is removed from the call stack

b) STD_EXEMPT and STD_DEDUCT are added to the global namespace

c) the estimated tax is printed

d) all of these

e) none of these

Types of errors.

TypeError, NameError, ZeroDivisionError are examples of which type of error?

a) syntax       b) runtime       c) logical/semantic       d) regression       e) integrated
Revising code.

Given the following Python code:

```python
1  def drawShape(s):
2     '''(int) --> None
3     Draw a square with sides of length s.
4     >>> drawShape(100)
5     [draws a square with sides length 100]
6     '''
7     turn = 90
8     for i in range(4):
9         fd(s)
10        lt(turn)
11     return None
```

Which lines of code would need to be changed to revise `drawShape` to draw an n-sided polygon, where n is a new argument to the function?

a) 1, 2, 14  

b) 1, 2, 4, 6, 7, 9, 10  

c) 1, 2, 4, 6, 7, 11, 12  

d) 1, 2, 4, 6, 7, 14  

e) 1, 6, 14

Given:

```python
def q1(astr):
    '''(str) --> ??'''
    slen = len(astr)
    ctr = 0
    for ch in astr:
        ctr += 1
    return slen == ctr

astr = 'CIS 2xx CIS 3xx'

>>> type(q1(astr))
bool

>>> q1(astr)
True
```
Given the following Python code:

def twice(x):
    '''test function'''
    y = 2
    result = x * y
    return result

>>> y = 5
>>> twice(y)
??-1
>>> y
??-2
>>> x
??-3

(19) x is defined in a ?? namespace:

a) local       b) global       c) built-in       d) both local and global       e) none of these

(20) y is defined in a ?? namespace:

a) local       b) global       c) built-in       d) both local and global       e) none of these

(21) twice is defined in a ?? namespace:

a) local       b) global       c) built-in       d) both local and global       e) none of these

(22) The value reported at ??-1 is

a) 4       b) 5       c) 10       d) 25       e) None

(23) The value reported at ??-2 is

a) 2       b) 5       c) 10       d) None       e) error

(24) The value reported at ??-3 is

a) 2       b) 5       c) 10       d) None       e) error
Given the following Python code:

```python
def thrice(x, y):
    '''test function'''
    z = 3
    print(x, y, z)
    result = z * x * y
    return result
```

```python
>>> x = 5
>>> y = 10
>>> answer = thrice(y, x)
>>> answer
?-1
>>> x
?-2
>>> y
?-3
>>> z
?-4
>>> result
?-5
```

What will be printed at line 4 when function `thrice` executes?

- a) 3, 5, 10  
- b) 5, 10, 3  
- c) 10, 5, 3  
- d) None  
- e) error

The value reported at ??-1 is

- a) 3  
- b) 5  
- c) 10  
- d) 150  
- e) error

The value reported at ??-2 is

- a) 3  
- b) 5  
- c) 10  
- d) 150  
- e) error

The value reported at ??-3 is

- a) 3  
- b) 5  
- c) 10  
- d) 150  
- e) error

The value reported at ??-4 is

- a) 3  
- b) 5  
- c) 10  
- d) 150  
- e) error

The value reported at ??-5 is

- a) 3  
- b) 5  
- c) 10  
- d) 150  
- e) error
Given the following Python code:

```python
from math import pi

def circle_area(rad):
    '''(num) -> float
    return the area of a circle with radius rad
    >>> circle_area(1)
    3.1
    >>> circle_area(2)
    12.6
    '''
    area = pi * rad ** 2
    area = round(area, 1)
    return area

def pizza_calculator(diameter, cost):
    '''
    (int, num) -> float
    Calculates and returns the cost per square inch
    of pizza for a pizza of given diameter and cost.
    >>> pizza_calculator(14, 18)
    0.117
    >>> pizza_calculator(14, 20.25)
    0.132
    '''
    r = diameter / 2
    area = pi * r ** 2
    cost_per_inch = cost / area
    cost_per_inch = round(cost_per_inch, 3)
    return cost_per_inch

Which of the following would you expect to see in the __main__ namespace after
from math import pi is executed?

a) import  b) math  c) pi  d) math.pi  e) math.sqrt

Which line of code below should replace lines 13 and 14 in pizza_calculator so that the pizza area
calculation is moved to circle_area?

a) area = circle_area(diameter / 2)
b) area = circle_area(rad)
c) circle_area(diameter / 2)
d) circle_area(rad)
Write a function, `add_more_digits`, to sum the digits of a non-negative integer, `n`. The sum is returned. All variables should be of type integer (no strings).

The function should be written using CIS 210 style guidelines; docstrings may be omitted. Python code should clearly reflect the underlying algorithm. Code should use only the most appropriate Python "tools" for solving the problem.

```python
def add_more_digits(n):
    '''(int) --> int

    Return sum of digits of n, a non-negative integer.
    '''
    digit_sum = 0
    ctr = 0
    while n > 0:
        digit = n % 10
        n = n // 10
        digit_sum += digit
        ctr += 1
    return digit_sum
```