CIS 210 Example Midterm Questions

Note: These questions are not a comprehensive study guide! They are given here to provide you with a sense of the types of multiple choice questions that may be on the midterm.

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

The format of the midterm will be multiple choice questions (Scantron), and one or more questions where you will be given a problem specification and write a solution in Python code using CIS 210 style guidelines. If the problem has a posted solution, that is, was assigned as a weekly project, a correct response will be code identical to or very close to the posted solution.

1) Given the following Python code:

```
def q1(z):
    '''midterm'''
    y = 2
    print(x, y)
    print(z)
    result = y * z
    return result
```

What is the result of executing

```text
>>> x = 10
>>> y = q1(x)
>>> print(x, y)
>>> print(z)
```

(a) (b) (c) (d) (e)
NameError 2 NameError 10 2 10 2 10 2
10 10 20 10
10 20 10 20 NameError
10 NameError TypeError

2) Given the following Python code:

```
def q2a(x):
    '''midterm'''
    x = x ** 2
    return q2b(x)

def q2b(z):
    '''midterm'''
    z = str(z)
    print(z)
    msg = 'the end'
    return msg
```

Parameter passing with immutable data types; functions calling functions; return values (computational modeling; what happens when a function is executed.)
What is the result of executing

```python
>>> q2a(2)
??
```

(a) '4'
(b) NameError
(c) 'the end'
(d) 4
(e) 2

3) Interpreted as a decimal number, binary 1000001 is

- Binary representation of data: interpreting binary representation of decimal numbers.

(a) 1  (b) 33  (c) 65  (d) 130  (e) one million and one

4) Given the following Python code:

Understanding code; Boolean expression; good programming style (the given code is better than the suggested change, though they both have the same result).

```python
1 def isInCircle(x, y, r):
2  
3   '''(number, number, number) -> Boolean
4   Returns True if point (x, y) is in
5   the circle with radius r.
6   
7   >>> isInCircle(0, 0, 1)
8   True
9   >>> isInCircle(.5, .5, 1)
10  True
11  >>> isInCircle(1, 2, 1)
12  False
13  '''
14  d = math.sqrt(x**2 + y**2)
15  isIn = d <= r
16  return isIn
```

What would be the result of changing the code in line 15 to

```python
isIn = (d <= r) == True
```

and then executing

```python
>>> isInCircle(0, 0, 1)
```

(a) no change  (b) will return False  (c) TypeError  (d) NameError  (e) SyntaxError

5-7) Given the following Python code:
def factr(n):
    """midterm ""
    if n == 0:
        return 1
    else:
        return n * factr(n-1)

def facti1(n):
    """midterm ""
    nfact = 1
    for i in range(2, n+1):
        nfact = nfact * i
    return nfact

def facti2(n):
    """midterm ""
    nfact = 1
    i = 2
    while i <= n:
        nfact = nfact * i
    return nfact

What will be the result of executing

Recursive and iterative implementations of math factorial; understanding recursive functions; issue with while loops (use for when possible).

5) >>> factr(3)

(a) 6  (b) 3  (c) 2  (d) 1  (e) infinite loop

6) >>> facti1(3)

(a) 6  (b) 3  (c) 2  (d) 1  (e) infinite loop

7) >>> facti2(3)

(a) 6  (b) 3  (c) 2  (d) 1  (e) infinite loop

8-11) Given the following UNTESTED Python code:

Understanding code; importance of good test cases for detecting logical errors (bugs); good test suites comprise basic, edge, and examples from equivalence classes for expected input and results.
def q8(s):
    '''midterm'''

    if len(s) != 0:
        prev_char = s[0]
        dup_ct = 1
        high_ct = 1
    else:
        high_ct = 0

    for i in range(1, len(s)):
        if s[i] == prev_char:
            dup_ct += 1
        else:
            prev_char = s[i]
            if dup_ct > high_ct:
                high_ct = dup_ct
            dup_ct = 1

    return high_ct

What will be the result of executing

8) >>> q8('abbccdde')
   (a) 0    (b) 1    (c) 2    (d) 3    (e) 4

9) >>> q8('abbccddddd')
   (a) 0    (b) 1    (c) 2    (d) 3    (e) 4

10) >>> q8('')
    (a) 0    (b) 1    (c) 2    (d) TypeError    (e) infinite loop

11) Additional helpful tests should include

    (a) 'effgghhi'    (b) 'a'    (c) 'aaabbc'    (d) (a) and (b)    (e) (b) and (c)
    'jkkllmnnnm'    '#'    'abc'

12-15) Given the following UNTESTED Python code:

    Understanding code; understanding Python syntax (whitespace and code blocks); bug detection.

def q12(s, c):
    '''midterm'''
ct = 0
for ch in s:
    if ch == c:
        ct += 1

return ct

What value will be returned when the following code is executed?

12) >>> q12('abc', 'b')

(a) 0   (b) 1   (c) 2   (d) None   (e) infinite loop

13) >>> q12('abc', 'd')

(a) 0   (b) 1   (c) 2   (d) None   (e) infinite loop

14) >>> q12('abbc', 'b')

(a) 0   (b) 1   (c) 2   (d) None   (e) infinite loop

15) The error in function q12 is a(n) ?? error.

(a) syntax   (b) runtime   (c) logic   (d) documentation   (e) infinite loop

16) Given the following Python code:

Dictionary data type; unordered, key access data type; Python string format method.

locD = {'210': 'PAC 123', '211': 'STB 145'}

def q16(freqD):
    """midterm""

        print('{:<6} {:<9}'.format('NAME', 'ROOM'))

    for pair in freqD.items():
        print('{:<6} {:<9}'.format(pair[0], pair[1]))

return None

What will be the result of executing

>>> q16(locD)
17-20) When the following Python code is executed:

List data type; mutable data type; mutable data types are updated in place (and immutable data types are not). Many list methods update a list as a side effect, and return None.

```python
>>> s = 'ABC'
>>> myl = ['d', 'e', 'f']
>>> s = s.lower()
>>> myl.append('g')
>>> print(s)
?– 17
>>> print(myl)
?– 18
>>> s.capitalize()
>>> myl = myl.append('h')
>>> print(s)
?– 19
>>> print(myl)
?– 20
```

17) What is printed at ?– 17?

a) 'abc'  b) 'ABC'  c) 'aBC'  d) 'Abc'  e) None

18) What is printed at ?– 18?

a) ['d', 'e', 'f', 'g']  b) ['d', 'e', 'f']  c) ['g', 'd', 'e', 'f']  d) ['g']  e) None

19) What is printed at ?– 19?

a) 'abc'  b) 'ABC'  c) 'aBC'  d) 'Abc'  e) None

20) What is printed at ?– 20?

a) ['d', 'e', 'f', 'g']  b) ['d', 'e', 'f']  c) ['g', 'd', 'e', 'f']  d) ['g']  e) None

21-24) When the following Python code is executed:

```python
>>> z = [1, 2, 3, 4]
>>> id(z)
4381224136
>>> z.remove(2)
>>> print(z)
```
>>> id(z)

>>> a = [False, 99.9, 'hi']
>>> b = [False, 99.9, 'hi']
>>> print(id(a) == id(b))

>>> print(a[2] == b[-1])

21) What is printed at ?? – 21?

a) [1, 2, 3, 4] b) [1, 3, 4] c) [1, 2, 4] d) [1, 4] e) None

22) What is printed at ?? – 22?

a) 4381224136 b) 4380207176 c) (a) or (b) d) True e) False

23) What is printed at ?? – 23?

a) 4381224136 b) 4380207176 c) (a) or (b) d) True e) False

24) What is printed at ?? – 24?

a) False b) 99.9 c) 'hi' d) True e) None

25-29) Given the following Python code:

Parameter passing with mutable data types causes an alias. This means that updating a mutable data type in a local namespace may be visible outside of function execution. Use list copy method to avoid aliasing. List method pop updates a list and returns a value: importance of understanding what functions do, including side effects and returned values. Computational modeling.

```python
def q25_a(x):
    """midterm""
    y = q25_b(x)
    print(x)
    print(y)
    return None

def q25_b(x):
    """midterm""
    y = x.pop()
    return y
```

```python
x = ['CIS210', 'CIS211', 'CIS212']
```

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

```python
>>> q25_a(x)
?? – 25
```
>>> z = x.copy()
>>> q25_a(z)
?? – 26
>>> print(x)
?? – 27
>>> print(z)
?? – 28
>>> print(y)
?? – 29

25) What is printed at ?? – 25?

a) ['CIS210', 'CIS211', 'CIS212']  
b) ['CIS210', 'CIS211', 'CIS212']  
c) ['CIS210', 'CIS211']
   CIS212

d) ['CIS210', 'CIS211']  
e) NameError

26) What is printed at ?? – 26?

a) ['CIS210', 'CIS211']  
b) ['CIS210', 'CIS211']  
c) ['CIS210']
   CIS211

d) []  
e) NameError

27) What is printed at ?? – 27?

a) ['CIS210', 'CIS211', 'CIS212']  
b) ['CIS210', 'CIS211']  
c) ['CIS210']

d) []  
e) NameError

28) What is printed at ?? – 28?

a) ['CIS210', 'CIS211', 'CIS212']  
b) ['CIS210', 'CIS211']  
c) ['CIS210']

d) []  
e) NameError

29) What is printed at ?? – 29?

a) ['CIS210', 'CIS211', 'CIS212']  
b) ['CIS210', 'CIS211']  
c) ['CIS210']

d) []  
e) NameError

30-31) Given the following Python code:
Functions may be passed as arguments, the same as other Python data types. Understanding code execution; functions calling functions; returned values.

def q30(li, cond):
    '''midterm'''
    result = []
    for elem in li:
        if cond(elem):
            result.append(elem)
    return result

def q30a(n):
    '''midterm'''
    return n % 2 == 0

def q30b(n):
    '''midterm'''
    return n > 0

numbers = [21, -5, -87, 3, 242, -7, -1, 2345]

30) What is the result of executing  
>>> len(q30(numbers, q30a))
a) 0  b) 1  c) 2  d) 3  e) 4

31) What is the result of executing  
>>> len(q30(numbers, q30b))
a) 0  b) 1  c) 2  d) 3  e) 4

32) What is the result of executing  
>>> print('{}').format('The end.'))

Basic string formatting.
a) {}  b) end The.  c) The end.