CIS 210 Example Exam 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 final exam.

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

The final exam 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 index card of handwritten notes.

1-3. Given:

```python
numbers = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '0']
special = ['!', '@', '#', '$', '%', '&', '*']
def q1(psw):
    ''' exam function '''
    score = 0
    for ch in psw:
        if ch in numbers:
            score += 2
        elif ch in special:
            score += 3
        else:
            score += 1
    return score > 8:
```

1. An appropriate type contract for function q1 would be

   a) (str) -> bool          b) (str) -> int
   c) (int) -> bool          d) (int) -> str

2. Function q1

   a) returns a non-None value; no side effect
   b) returns None value; causes a side effect
   c) returns a non-None value; causes a side effect
   d) returns None value; no side effect

3. What will be the result of executing the following code:

   >>> q1('hello!')

   a) 6          b) 8          c) True          d) False
4-6. Given:

```python
def q4(li):
    '''exam function. li is a sorted list of integers'''
    g = 0
    for i in range(1, len(li)):
        d = li[i] - li[i-1]
        if d > g:
            g = d
    print(g)
    return None
```

4. An appropriate type contract for function q4 would be

- a) (int) -> None
- b) (int) -> int
- c) (list) -> None
- d) (list) -> int

5. Function q4

- a) returns a non-None value; no side effect
- b) returns None value; causes a side effect
- c) returns a non-None value; causes a side effect
- d) returns None value; no side effect

6. What will be the result of executing the following code:

```python
>>> q4([1, 2, 4, 7])
```

- a) 0
- b) 1
- c) 2
- d) 3

7. Given:

```python
def q7(n):
    '''(int) -> int
    n is a non-negative integer'''
    if n == 0:
        return 0
    else:
        return 10 + q7(n - 1)
```

What will be the result of executing the following code:

```python
>>> q7(3)
```

- a) 10
- b) 20
- c) 30
- d) 40
9-10. Given:

def q9(s):
    '''(str) -> str
    
    Return string with information about string length.
    
    >>> q9('hello, world')
    'hello, world-12'
    '''
    ct = len(s)
    s = s + '-' + ct
    return s

9. Executing function q9 will cause a

a) logic error  b) run time error – IndexError  c) run time error – TypeError

10. The error in function q9 stems from what characteristic of Python?

a) static typing  b) strong typing  c) parameter passing  d) None value

11. Given:

def q11(s):
    ''' Exam function '''
    vowels = ['a', 'e', 'i', 'o', 'u']
    svowels = [ch for ch in vowels if ch in s]
    return svowels

What will be the result of executing the following code:

>>> q11('quick brown fox')

a) ['i', 'o', 'u']  b) ['a', 'e', 'i', 'o', 'u']  
c) 3  
d) None
12. Given:

def q12(s):
    ''' exam function '''
    vowels = ['a', 'e', 'i', 'o', 'u']
    vowelct = 0
    for item in s:
        if item in vowels:
            vowelct += 1
    return vowelct

What will be the result of executing the following code:

```python
>>> q12('the quick brown fox')
```

a) 0  b) 1  c) 5  d) None

13. Given:

def q13(s):
    ''' exam function '''
    vowels = ['a', 'e', 'i', 'o', 'u']
    vowelct = 0
    for item in s:
        if item in vowels:
            vowelct += 1
    return vowelct

What will be the result of executing the following code:

```python
>>> q13('the quick brown fox')
```

a) 0  b) 1  c) 5  d) None
14-17. Given:

def q14(alist):
    ''' exam function '''
    countd = {
        for item in alist:
            if item in countd:
                countd[item] += 1
            else:
                countd[item] = 1

    countli = countd.values()
    maxct = max(countli)

    mli = []
    for k in countd:
        if countd[k] == maxct:
            mli.append(k)

    return mli

14. An appropriate type contract for function q14 would be

a) (list) -> dict       b) (dict) -> list

15. Function q14

a) returns a non-None value; no side effect
b) returns None value; causes a side effect
c) returns a non-None value; causes a side effect
d) returns None value; no side effect

16. What will be the result of executing the following code:

```python
>>> q14([1, 1, 1, 2, 2, 2, 3, 3, 3])
```

a) []       b) [1, 2, 3]

c) {1:3, 2:3, 3:3}       d) [1, 1, 1, 2, 2, 2, 3, 3, 3]

17. The better test suite for function q14 would be

a) q14([5, 7, 1, 3])
    q14([1, 2, 2, 3, 99])
    q14([99])
    q14([0, 0, 1, 1])

b) q14([1, 2, 3, 4])
    q14([100, 99, 98, 97])
    q14([2, 4, 6, 8])
    q14([1, 7, 5, 3])
18-19. Given:

def q18_aux(codeli, resultli):
    ''' exam function '''

    for item in codeli:
        resultli.append(item[0])

    return None

def q18_main():
    ''' exam function '''

    codeli = [('a', 'z'), ('b', 'y'), ('c','x')]
    resultli = []
    q18_aux(codeli, resultli)

    # checkpoint 1
    print(codeli)

    # checkpoint 2
    print(resultli)

    return None

18. What will be printed at checkpoint 1?

>>> q18_main()
a) [('a', 'z'), ('b', 'y'), ('c','x')]    b) [('a', 'z')]
c) ['a', 'b', 'c']                        d) None

19. What will be printed at checkpoint 2?

>>> q18_main()
a) []                                      b) ['a']
c) ['a', 'b', 'c']                        d) None
20-22. Given:

def foo(li, item):
    ''' exam function '''
    li.append(item)
    print(li)
    li = li.append(item)
    print(li)
    return item + 1

def bar():
    ''' (...) -> None '''
    x = 99
    foo([], x)
    # checkpoint 1
    x = foo([], x)
    # checkpoint 2
    return None

What will be the result of executing the following code:

```python
>>> bar()
```

20. What will have been printed after the first call to foo?

<table>
<thead>
<tr>
<th>a) [99]</th>
<th>b) None</th>
<th>c) 99</th>
<th>d) [99]</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>[99]</td>
<td>[99]</td>
<td>[99, 99]</td>
</tr>
</tbody>
</table>

21. What is the value of x at checkpoint 1?

a) [99]  b) 99  c) 100  d) None

22. What is the value of x at checkpoint 2?

a) [99]  b) 99  c) 100  d) None
Given the following Python code:

```python
def q17(dd):
    '''(??-17) -> ??-18
    ...
    ddv = dd.values()
    max_ddv = max(ddv)
    res = [k for k in dd if dd[k] == max_ddv]
    return res
```

(17) Replace ??-17 with the correct type:

a) int  b) str  c) list  d) dict  e) None

(18) Replace ??-18 with the correct type:

a) int  b) str  c) list  d) dict  e) None

(19) In `res = [k for k in dd if dd[k] == max_ddv]`, k is a

a) dict key  b) dict value  c) list index  d) str  e) class

(20) What is the result when `>>> q17({1:1, 2:1, 4:1, 7:1})` is executed?

a) 1  b) 7  c) [1, 7]  d) [2, 4]  e) [1, 2, 4, 7]

(21) `res = [k for k in dd if dd[k] == max_ddv]` is an example of a(n) ?? pattern:

a) accumulator  b) filter  c) map  d) file
**k-means cluster analysis algorithm:**

(22-23) Given data values, an initial set of randomly chosen centroids, and a value for k:

data values = 2, 8, 6, 4, 10, 12, 20, 18, 16, 14

centroids = 2, 8, 14, 20

k = 4

22) After an initial round of executing the k-means cluster analysis algorithm, using Euclid distance as the similarity measure, the data will have clustered as (clusters are indicated by <>):

a) <2>, <4>, <6, 8, 10, 12, 14, 16, 18>, <20>  
b) <2, 4, 6>, <8, 10, 12>, <14, 16, 18>, <20>  
c) <2, 4>, <6, 8>, <10, 12, 14>, <16, 18, 20>  
d) <2> <4, 6, 8>, <10, 12, 14> <16, 18, 20>

23) The second round of centroids will then be:

a) 2, 8, 14, 20  
b) 3.0, 8.0, 14.0, 19.0  
c) 2.0, 6.0, 12.0, 18.0  
d) 1.0, 7.0, 13.0, 19.0

**File processing.**

(24-26) Given the following Python code:

```python

testfile = 'test.txt'
testf = open(testfile)  
testdata = testf.read()  
testf.close()
```

(24) testfile is a

**a) string**  
**b) list**  
**c) reference to a file object**  
**d) list of strings**

(25) testf is a

**a) string**  
**b) list**  
**c) reference to a file object**  
**d) list of strings**

(26) testdata is a

**a) string**  
**b) list**  
**c) reference to a file object**  
**d) list of strings**
Executing a (recursive) function.

Given the following Python code:

```python
def dtobr(n):
    '''(int) -> str
    Use recursive algorithm to convert n >= 0 to binary string.
    >>> dtobr(4)
    '100'
    >>> dtobr(0)
    '0'
    >>> dtobr(1)
    '1'
    >>> dtobr(2)
    '10'
    '''
    if n < 2:
        return str(n)
    else:
        return dtobr(n // 2) + str(n % 2)
```

When `>>> dtobr(27)` is entered in the Shell, what does Python do?

a) add an activation record to the call stack  
b) create a local namespace  
c) assign parameter `n` to argument 27  
d) call `dtobr(13)`  
e) all of these
23. Write a function, `raindata`, with one parameter, `f`, the name of a file containing rain data (daily rainfall in Eugene from March 1 through March 15).

The file will have two lines: a header line, and another line of comma separated values, which are the daily rainfall amounts:

```
# rainfall data for final exam CIS 210 W17 03/01-15/yyyy Eugene, OR
.2, .19, .16, .2, .16, .15, .2, .12, .15, .16, .16, .15, .16, .2
```

Function `raindata` should create and return a rain dictionary, where the dictionary keys are the days of the month (1 through 15) and the associated value is the rainfall for that day.

Code should be written according to CIS 210 style guidelines, including a docstring with a type contract; brief description that explicitly refers to parameters, return value, and side effects (if any); and at least one example of use. Example of use may refer to a small example file, e.g.,

```
# rainfall data
1.5, .5
```

def raindata(f):
    '''(str) -> dict

    Rainfall data from file f are stored in a data dictionary by day of month.
    >>> raindata('rainfall.txt')
    {1:.2, 2:.19}
    '''
    with open(f) as rainf:
        rainf.readline()
        rain = rainf.readline().strip().split(',')

    raindict = {}
    day = 1
    for rainct in rain:
        raindict[day] = float(rainct)
        day += 1

    return raindict