CS1026 Review/Practice Questions – Quiz 2
Fall 2019

c

1. What happens in this code snippet if `commonDivisor(10,5)` is executed?

```
def commonDivisor(n,d):
  if (n % d == 0):
    return n
  else:
    return None
```

a. `2`
b. `5`
c. `10`
d. `None`
e. `None of the above`

d.

2. What is printed by the following code snippet?

```
numLis t= [1,3,6,9]
for i in range(len(numList)):
  print (i, end="-")
print()
```

a. `1-3-6-9-`
b. `1-3-6-9`
c. `0-1-2-3`
d. `0-1-2-3-`
e. `None of the above`

d.

3. Which of the following creates an empty set?

a. `S={}`
b. `S=[]`
c. `S=set([''])`
d. `S=set()`
e. `None of the above`

4. What is printed by the following code snippet?

a.

```
wordsList = ['a']
wordsList.append('b')
wordsList.append('c')
wordsList.pop()
wordsList.append('d')
wordsList.append('e')
print(wordsList[:2])
```

pop the end of the list

=> wordlist[0], wordlist[1].

a. ['a','b']
b. ['b','c']
c. ['d','e']
d. ['c','d']
e. None of the above

左闭右开.

5. What is printed by the following code snippet?

C

This is.

```
line = "This is the quiz 3 for fall 2019"
sum = 0
for w in line:
    if w.isdigit():
        sum += int(w)
print(sum)
```

every character.

3+2+0+1+9.

a. 2022
b. 2019
c. 15
d. 3
e. None of the above

d.

6. Which of the following code snippets remove '$' and '!' in any order from right side of a sting, s?

rightstrip

a. s.rstrip("$")
b. s.lstrip("!$")
c. s.rstrip("$").rstrip("!")
d. s.rstrip("!$")
e. None of the above

strip 函数仅对处于最左端/最右端字符串有效，对中间无效.

都可以，总能除去左右端为"!$"的.

7. In the code snippet below, if the file **input.txt** contains the following three lines:

```
Monday!
Tuesday.
Wednesday?
```

What would be the output?

```
infile = open("input.txt", "r")
for word in infile :
    word = word.lstrip(".!")
    print(word)
```

a.
```
Monday!

Tuesday.

Wednesday?
```

b.
```
Monday!
Tuesday.
Wednesday?
```

c.
```
Monday

Tuesday

Wednesday?
```

d.
```
Monday
Tuesday
Wednesday?
```

e. None of the above

8. What is the output of the following code snippet?

```
even = {2,4,6,8,10}
prime = {2,3,5,7}
print(prime.union(even))
```

a. {2}
b. {2, 3, 4, 5, 6, 7, 8, 10}
c. {2, 2, 3, 4, 5, 6, 7, 8, 10}
d. {3, 5, 7}
e. None of the above

9. Consider that we have:

```
student =
[{'name':'John','lastName':'Smith','course':['1026A','1026B','1027A']},
 {'name':'Mario','lastName':'Rossi','course':['2026A','2026B','2027A']}]
```

Which of the following code snippet prints **'2026B'**?

a. `student[1]['course'][2]`
b. `student[1]['course'][1]`
c. `student['Mario']['course'][1]`
d. `student[0]['course'][1]`
e. `None of the above`

10. Consider the following code segment:

```
print("W", end="")
try :
    inFile = open("test.txt", "r")
    line = inFile.readline()
    value = int(line)
    print("X", end="")
except IOError :
    print("Y", end="")
except ValueError :
    print("Z", end="")
```

What output is generated when this program runs if **test.txt** is not opened successfully?

a. `WY`
b. `WZ`
c. `WX`
d. `WYX`
e. None of the above

11. Consider the following code segment:

```
class Student :
    def __init__(self, aName, anId):
        self._name = aName
        self._id = anId
    def getName(self):
        return self._name
    def setName(self, newName):
        self._name = newName
    def getId(self):
        return self._id

student1=Student("John","1234567")
```

Which of the following code snippet is the correct way to print **'John'**?

a. `print (student1._name)`
b. `print (._name)`
c. `print (student1.getName())`
d. `print (getName())`
e. None of the above

12. Consider the following code segment:

```
class Student :
    def __init__(self, aName, anId):
        self._name = aName
        self._id = anId
    def getName(self):
        return self._name
    def setName(self, newName):
        self._name = newName
    def getId(self):
        return self._id

student1 = Student("John","1234567")
student3 = Student("Kate","4567890")
student2 = student1
student2.setName("Mario")
```

What is the name student1?

a. `"Mario"`
b. `"John"`
c. `"Kate"`
d. `None`
e. None of the above

13. Consider that we store names and phone numbers in a dictionary as follows:

phoneBook={'John':'5554441234','Kate':'4443331234'}

Which of the following code snippets prints just names in the phonebook in one line?

a.
```
for k in phoneBook:
    print (phoneBook[k])
```
b.
```
for k in phoneBook:
    print (k)
```
c.
```
for k in phoneBook:
    print (k, end=' ')
```
d.
```
for k in phoneBook:
    print (phoneBook[k],end=' ')
```
e. None of the above

The program `wordFrequency.py` finds the frequency of the words in a predefined list of words that are in a document that is provided by the user, as input. There is a function `wordFrequency(l,wDict,words)` that updates the frequency of words in the dictionary (`wDict`) using the word in a line (`l`) and the list of predefined words (`words`); it also strips off any `period` at the beginning or end of the word. The Function `readFile()` receives the name of the file from the user and open the file. It returns the file object.

The program and function need to be completed. Identify the number of the lines that are needed to complete the blanks in each ***Section*** of the code; the vertical bars indicate where the line of code is indented. The code with blanks is below; lines to choose from follow.

```
# List of words to find their frequency
WORDSLIST=["Great", "awesome","Good","happy","Excited"]

def wordFrequency(l,wDict,words):
   |_____Section #1
   for w in splittedLine:
      w=w.strip('.')
      if w in words:
         try:
            |_____Section #2
         |________
            |________

def readFile():
   try:
      fileName = input("Enter file name: ")
      inputFile=open(fileName,"r")
      |________Section #3
   |________Section #4
      print ("The text file name does not exist.")


wordCountDict={}
|________Section #5
for line in inf:
   |________Section #6


# Print out the words and their frequency
for key in wordCountDict:
   print (key, "has frequency of ", wordCountDict[key])
```

Choose the lines from the list below; some lines may be used more than once.

```
1. return inputFile
2. inputFile=open(fileName,"r")
3. inputFile=open(fileName,"w")
4. inf=readFile()
5. readFile()
6. readFile(inf)
7. wordFrequency(line,wordCountDict,WORDSLIST)
8. wordFrequency(wordCountDict,WORDSLIST)
9. wordCountDict=wordFrequency(line,wordCountDict)
10. wordCountDict+=1
11. wDict[w]+=1
12. wDict[w]=1
13. wDict[w]=0
14. wDict[key]=1
15. splittedLine=l.split()
16. splittedLine=line.split()
17. splittedLine=l.split('-')
18. splittedLine=splittedLine.strip('.')
19. except KeyError:
20. except IndexError
21. except ValueError
22. except FileNotFoundError:
```

14. Which line would complete ***Section #1*** of the function `wordFrequency(l,wDict,words)`
    a. 14
    b. 17
    c. 16
    d. 15
    e. None of the above

15. Which lines, in the order given, would complete ***Section #2*** of the function `wordFrequency(l,wDict,words)` ?
    a. 11,18,11
    b. 12,18,10
    c. 11,20,12
    d. 11,19,12
    e. None of the above

16. Which line would complete ***Section #3*** of the function `readFile()`?
    a. `6`
    b. `4`
    c. `1`
    d. `2`
    e. None of the above

17. Which line would complete ***Section #4*** of the function `readFile()`?
    a. `19`
    b. `20`
    c. `21`
    d. `22`
    e. None of the above

18. Which line would complete ***Section #5*** of the main program?
    a. `1`
    b. `6`
    c. `4`
    d. `5`
    e. None of the above

19. Which line would complete ***Section #6*** of the main program?
    a. `10`
    b. `8`
    c. `7`
    d. `9`
    e. None of the above

20. The following class, `Message`, is a class that might be used in designing an email system. Review the class definition and then answer the questions that follow.

```
class Message :
   def __init__(self, sender, recipient) :
      self._sender = sender
      self._recipient = recipient
      self._body = ""
   ## Append a line of text to the message.
   #  @param line the line of text to add to the message
   def append(self, line) :
      self._body = self._body + line + "\n"
   ## Return the entire message as a string.
   #  @return a string representation of the entire message
   def __repr__(self) :
      result = "From: " + self._sender + "\n"
      result = result + "To: " + self._recipient + "\n"
      result = result + self._body
      return result
```

i. What is the constructor for the class `Message`? def -init-
ii. What are the instance variables for this class? ~~body~~
iii. Explain what the method `__repr__` does and how it is used.
iv. Write a "getter" method for the sender instance variable
v. Write a "setter" method for the sender instance variable

21. Consider the following Python classes and then answer the questions following.

```
#--------------------
## Represent an employee with a name and salary.
class Employee :

   def __init__(self, name, salary) :
      self._name = name
      self. salary = salary

   def __repr__(self) :
      return self._name + " has a salary of %.2f" % self._salary

#--------------------
## Represent a manager with a department.
class Manager(Employee) :

   def __init__(self, name, salary, department) :
      super().__init__(name, salary)
      self._department = department

   def __repr__(self) :
      return self._name + " has a salary of %.2f" % self._salary + \
               " and manages the " + self._department + " department"
```

```
#--------------------
## Represent an executive.
class Executive(Manager) :

   def __repr__(self) :
      return self._name + " has a salary of %.2f" % self._salary + \
               " and is the executive for the " + self._department + "
department"
```

i. What is the constructor for the class `Manager`?
ii. What class is the class `Manager` a subclass of?
iii. What class is the class `Executive` a subclass of?
iv. If you create an object of the class `Manager`, what instance variables would it have?
v. If you create an object of the class `Executive`, what instance variables would it have?
vi. What does `super()` do in the class `Manager`?
vii. Write a method for the class `Manager`, that will get the department name of a manager.

viii. Consider the following lines of code.

```
emp = Employee("Mario Rossi", 18000.00)
print(man.getDepartment())
```

What happens when you run the code?