CS1010E Mid-term Test (AY2020/2021, SEM1)

Section 1: Evaluate the following Python expressions

1. Evaluate: `1+2-3*4+5`
    a. -1
    b. -4
    c. 2
    d. -24
    e. 1

2. Evaluate: `'abc'[3]`
    a. `'a'`
    b. `'c'`
    c. `'abc'`
    d. `''` (Empty string)
    e. Error

3. Evaluate: `'abc'[4:9]`
    a. `'a'`
    b. `'c'`
    c. `'abc'`
    d. `''` (Empty string)
    e. Error

4. Evaluate: `False or False or False`
    a. True
    b. False
    c. 0
    d. 1
    e. Error

5. Evaluate: `anUndefinedVariable or True`
    a. True
    b. False

c. 0
d. 1
e. Error

6. Evaluate: `False and anUndefinedVariable or True`
   a. False
   b. True
   c. 0
   d. 1
   e. Error

7. Evaluate: `'a' + 'b' * 3 * 2`
   a. 'ababab'
   b. 'abababababab'
   c. 'abbbbbb'
   d. '' (Empty string)
   e. Error

8. Evaluate: `[(1,2,(3,4)),(5,(6))][1][-1]`
   a. (6,)
   b. 6
   c. 5
   d. 2
   e. (3,4)

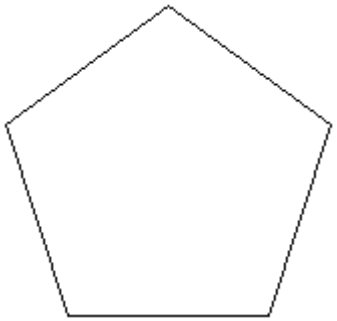
9. Evaluate: `list((1)) + list([2]) + list('3')`
   a. [1, 2, '3']
   b. Error!
   c. [(1,), [2], ['3']]
   d. [[1], [2], ['3']]
   e. [1, [2], [3]]

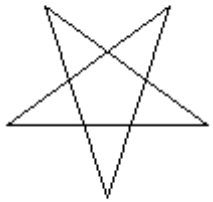
10. Evaluate: `(lambda x,y: lambda z:x(y(z)))(lambda x:x+1,lambda y:y*2)(2)`
   a. (3,4)
   b. 5
   c. 6
   d. 4
   e. 8

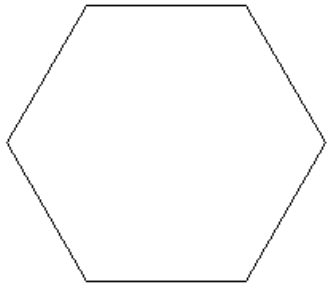
Section 2 : Output Prediction: If we put each of the following into a .py file and run. What will be the output?

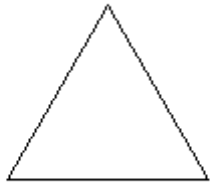
11. What will the following code draw? (Pictures are not drawn in scale)

```
from turtle import *
def drawSomething():
    for _ in range(6):
        fd(100)
        rt(360 - 360//5)
    ht() # hiding the turtle cursor
drawSomething()
```

a) 

b) 

c) 

d) 

E) None of the above

12. What is the output of this following code?

```
def doubleSeq(l):
    for i in range(len(l)//2):
        l[i] //= 2
    return l

print(doubleSeq([1,2,3,4,5,6]))
```

a. Error
b. (0, 1, 1, 4, 5, 6)
c. [0, 1, 1, 4, 5, 6]
d. [0, 1, 1, 2, 2, 3]
e. (0, 1, 1, 2, 2, 3)

13. What is the output of this following code?

```
t1 = [1,2,3]
t2 = (t1,t1)
t2[0][2] = 0
print(t2)
```

a. Error
b. [((1, 2, 3), (1, 2, 3)), (1, 2, 3)]
c. ([1, 2, 0], [1, 2, 0])
d. ([1, 2, 0], [1, 2, 3])
e. ([1, 2, 3], [1, 2, 3])

14. What is the output of this following code? (Note that 362880 = 1 x 2 x 3 x .. 9)

```
ans = 1
for i in range(0,10,5):
    ans *= i
print(ans)
```

a. 6
b. 362880
c. 0
d. 1
e. 46

15. What is the output of this following code?

```
def foo(n):
    output = 0
    for i in range(n):
        for j in range(n):
            if i == j:
                output += i*j
    return output

print(foo(3))
```

a. 0
b. 3
c. 5
d. 14
e. a number larger than 14

16. What is the output of this following code?

```
def foo():
    if True:
        return 999
    return callWhat()
print(foo())
def callWhat():
    return 123
```

a. Error!
b. 123
c. 999
d. None
e. None of the above

17. What is the output of this following code?

```
def foo(x, y):
    return lambda z: z - x + y

print([foo(1,2)(3)])
```

a. [4]
b. [2]
c. 2
d. [0]
e. Error!

18. What is the output of this following code?

```
def foo(x):
    return lambda x: x+x
print(foo(5)(2))
```

a. 2
b. 7
c. 4
d. 8
e. 10

19. What is the output of this following code?

```
def foo(x):
    return lambda y: x(x(y))
def square(x):
    return x+3
print(foo(foo(square))(2))
```

a. 65536 (this is equal to 2 to power 16)
b. 5
c. 14
d. 8
e. 12

20. What true about this following code if the input `lst` is a list of integers with length more than 1?

```
def foo(lst):
    while len(lst) > 1:
        for i in lst[1:]:
            if lst[0] >= i:
                lst.remove(lst[0])
    return lst[0]
```

a. Always return the minimum of the list
b. Always return a list with the minimum of the list removed
c. Always return a sorted list in an ascending order
d. Always return the maximum of the list
e. This code may fall into infinite loop for some input

21. What is the functionality of this following code if the input `lst` is a list of integers with length more than 3?

```
def foo(lst):
    if not lst:
        return []
    a = min(lst)
    lst.remove(a)
    return [a] + foo(lst)
```

a. Return the minimum of the list
b. Return a list with the minimum of the list removed
c. Return a sorted list
d. Return a randomly scrambled list that may or may not be sorted
e. Return the smallest three numbers of the input

Section 3 Debugging

22. Consider the following buggy function that takes a non-empty sequence of integers as its argument:

```
1 def chking(seq):
2     d=list(seq)
3     b=len(seq)
4     for a in range(b-1):
5         for i in d:
6             if i == d[a:b]:
7                 return False
8             elif i >= max(d[a+1:b]):
9                return True
10            else:
11               return False
12    return True
```

Which line in the function will **never** get executed, regardless of the integer values contained in **`seq`**?

A. Line 7

B. Line 9

C. Line 11

D. Line 12

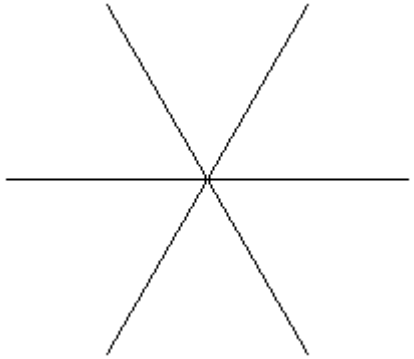
E. None of the above

23. What will be the range of the input `x` that will *crash* this function `foo()` assuming the input `x` is always an integer?

```
from math import sqrt
def foo(x):
    return x > 0 and sqrt(x+3) < 10
```

a. `x > 0`
b. `x < 3`
c. `0 <= x <= 3`
d. `All of the above`
e. `None of the above. Namely, no value of x will crash the code if x is an integer.`

24. In order to draw the following picture on the left, what are the missing lines?



```
from turtle import *
def drawSomething():
    for _ in range(6):
        ??? # Missing line
        ??? # Missing line
        ??? # Missing line
    ht()
drawSomething()
```

a) only this will work:

```
fd(100)
bk(100)
rt(360//6)
```

b) only this will work:

```
bk(100)
rt(360//6)
fd(100)
```

c) only this will work:

```
bk(100)
fd(100)
rt(360//6)
```

d) All of the three choices in all the boxes in a), b) or c) will work
e) Only two of the choices in the three boxes in a), b) or c) will work

25. Given a sorted ascending sequence with numbers and length > 1, we want to find the first largest gap between two consecutive numbers. Here is the expected behaviour of the function:

```
>>> l = [1,3,5,7,19,21,22,24,36,39]
>>> print(firstLargestGap(l))
12
```

So the gap is 12 that is between the numbers 7 and 19. Here is the code:

```
def firstLargestGap(l):
    ans = -1
    ?????????????????????? # The missing line
        gap = l[i+1]-l[i]
        if gap > ans:
            ans = gap
    return ans
```

Which one below is the correct line for the missing line?

a. `for i in range(0,len(l)-1):`
b. `for i in range(0,len(l)):`
c. `for i in range(0,len(l),2):`
d. `for i in range(0,len(l)-1,2):`
e. `for i in range(1,len(l)):`