CS1010E Mid-term Test (AY2021/2022, SEM1)

Plan

SMH = simply, medium, hard
1-6 SMH Expressions without seq
7-9 SMH seq expressions
10-12 SMH lambda
13-20 SMH code output predictions
21 Recursion
23 Functions

QN	Questions	Answer
	Evaluate the following expression without any pre-defined variable or packages imported	
1	`1-2-3*4-5` A. -18 B. 0 C. -8 D. 4 E. 16	A
2	`(False == True) == False` A. True B. False C. None D. Error E. 0	A
3	`'4'*3+'2'*1+'0'*0` A. '4442' B. '43210' C. '432100' D. '44420' E. '1220'	A
4	`False == True == False` A. False B. True C. None	A

	D. Error E. 0	
5	'abcde'[2:5][1][0][0] A. ‘d’ B. Error C. ‘’ D. [] E. ‘cde’	A
6	(True != False) or (sqrt(-1)) A. True B. False C. None D. 1j E. Error	A
7	[1,2,3,4,5,6][1:5][:2] A. [2, 3] B. [2, 3, 4, 5] C. [4, 5] D. [2, 3, 4] E. [2]	A
8	list(['abc'])+list(('k','z')) A. ['abc', 'k', 'z'] B. ['a', 'b', 'c', 'k', 'z'] C. ['a', 'b', 'c', 'kz'] D. ['abc', 'k', 'z'] E. ['abckz']	A
9	[5,[3],[2,3]][[2,1][0]][:[1,2][1]] A. [2, 3] B. Error C. [2] D. [3] E. []	A
10	(lambda a:return a+1)(2+3) A. Error B. 6 C. 5 D. 0 E. None	A

11	`(lambda a,b,x:b(a(x)))((lambda a:a*2),(lambda a:a+1),5)` A. 11 B. 12 C. A function D. Error E. (10,6)	A
12	`(lambda a,b:lambda x:b(x(a)))('a',lambda a:a*2)(lambda a:a[1:])` A. ‘’ B. ‘a’ C. ‘aa’ D. A function E. Error	A
	If the following is in a .py file, what is the output in console if you execute/run it?	
13	`x = 1` `for i in range(0,9):` `  for j in range(4,8):` `  x = x + 1` `print(x)` A. 37 B. 36 C. 41 D. 28 E. 145 F. 199	A
14	`q = 15` `if q > 10:` `  if q < 7:` `  print('a')` `  elif q < 9:` `  print('b')` `  else:` `  print('c')` `else:` `  print('d')` A. ‘c’ B. ‘b’	A

	C. 'a' D. 'd' E. Print nothing	
15	def f1(x): return 1+f3(x) def f2(x): return 2+f4(x) def f3(x): return 3+x print(f1(4)) A. 8 B. Error C. Infinite loop D. 7 E. None	A
16	def f1(x): return '1'+f2(x) def f2(x): return f3(x)+'2' def f3(x): return '3'+f4(x) def f4(x): return '4'+str(x) print(f2(0)) A. '3402' B. '13402' C. '3042' D. '13042' E. '12340'	A
17	x = [1,2,3] def foo(l,x): if not l: return l return foo(l[1:],x) + [x(l[0])] print(foo(x,lambda x:4-x)) A. [1, 2, 3] B. [3, 2, 1] C. [0, 1, 2]	A

	D. [2, 1, 0] E. Error	
18	d = {0:9, 1:0, 2:1, 3:4, 4:1, 5:9, 6:1} a = 4 while a in d: a = d[a] print(a) A. 9 B. 0 C. 1 D. Infinite loop E. Error	A
19	lst1 = ['bc','de','ya','ab','bq','bd'] d = {} for x in lst1: d[x[1]] = x[0] print(d['b']) A. 'a' B. 'c' C. Error D. 'bc' E. ''	A
20	x = {'a','bc','de'} y = {'b','de','a','b'} print(x^y) A. {'bc', 'b'} B. {'de', 'a'} C. {} D. {'d', 'e', 'a'} E. {'bc', 'de', 'a', 'b'}	A
	Fill in the blanks	
21	The following function take in a positive integer and return an integer that keeps its digits in the same order that are even only. E.g. >>> evenDigits(123456) 246 >>> evenDigits(12332401)	Blank1: evenDigits(N//10)*10 Blank 2: N%10 Blank 3: evenDigits(N//10)

	2240 Fill in the blanks for the missing part in the code to complete the following function as mentioned above: `def evenDigits(N):` `    if N == 0:` `        return 0` `    if N%2==0:` `        return __Blank1__ + __Blank2__` `    else:` `        return  __Blank3__`	correct code: `def evenDigits(N):` `    if N == 0:` `        return 0` `    if N%2==0:` `        return evenDigits(N//10)*10+N%10` `    else:` `        return evenDigits(N//10)`
22	Given a list L with integers that are unique. The function `num_pair(L,N)` will check how many pairs of numbers in L with their sum as N. For example: `>>> L = [75, 80, 90, 77, 88, 91, 60, 74, 73, 70, 55, 93, 59]` `>>> print(num_pair(L,150))` `4` `>>> print(num_pair(L,152))` `2` Fill in the blanks for the missing part in the code to complete the following function as mentioned above: `def num_pair(L,N):` `    a = len(L)` `    count = 0` `    for i in range(0,a):` `        for j in range(__Blank1__,a):` `            if __Blank2__ == N:` `                __Blank3__` `    return count`	Blank 1: `i+1` Blank 2: `L[i]+L[j]` Or `L[j]+L[i]` Blank 3: `count += 1` or `count = count + 1` Correct complete code: `def num_pair(L,N):` `    a = len(L)` `    count = 0` `    for i in range(0,a):` `        for j in range(i+1,a):` `            if L[i]+L[j] == N:` `                count += 1` `    return count`
23	Given that `L` is a list of integers with length > 1, what does the function `foo(L,0)` do? `def foo(lst,N):` `    l1 = lst[1:]`	A

	`` l2 = [] for i in range(len(l1)): l2.append(l1[i]-lst[i]) return min(l2) >= N ``` A. Check if the list L is sorted in descending order B. Check if all the elements of L are bigger than or equal to 0 C. Check if there exists at least one element in L that is bigger than or equal to 0 D. The function actually always crashes. It won't work E. Check the minimum of L is bigger than or equal to 0	
24	Given two strings `s1` and `s2` with alphabets only, if we want to check if they are anagrams, which of the following method is wrong? A. Check if `set(s1)` and `set(s2)` are equal B. Check the counts of each character in each string if they are equal C. Sort the two lists `list(s1)` and `list(s2)`, and check if they are the same D. Generate every permutation of `s2`, and check if `s1` is one of the permutations E. For each character c in s1, check if c is in s2. If so, remove one occurrence of c from s2. If c is not in s2, then they are not anagrams. They are anagram if s2 becomes an empty string at last.	A
25	In a dictionary in Python, A. The values can be any data type B. The values cannot be integers C. The values cannot be lists D. The values cannot be tuples E. The values cannot be strings	A