**Student Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section Number \_\_\_\_\_\_\_\_\_**

*This exam is double-sided. Be sure to pay attention to all instructions.*

Q1. [**28 pts.**] Compute the value of these expressions assuming the following variable values.

s = 'Tacocat'  
a = 3

|  |  |  |
| --- | --- | --- |
|  | Expression | Expression Value |
| a) | s[0] == s[6] | False |
| b) | s.split('c') | ['Ta', 'o', 'at'] |
| c) | s[a] | 'o' |
| d) | s.index('a', a) | 5 |
| e) | s == a | False |
| f) | s \* a | 'TacocatTacocatTacocat' |
| g) | range(a) | [0, 1, 2] |

|  |
| --- |
| Q2 Sample Output |
| >>> matchCount([1,9,2,9], 9, 2)  **True** >>> matchCount([8,3,2,3], 8, 2)  **False** >>> matchCount([1,3,7,3], 7, 1)  **True** |

Q2. [**16 pts.**] Write a function matchCount() that takes three parameters (a list, a value, and a count), and returns whether value appears in list exactly count times. You can assume that count > 0.

def matchCount(list, value, count):

for element in list:  
 if element == value:  
 count = count - 1

return count == 0

Q3. [**25 pts.**] Given the following code, you execute main(). Draw the function frames, variable declarations and value assignments for the following code THE FIRST TIME it stops at the line marked “STOP HERE”.

**def** update(list, v):  
 e = len(list)-1  
 list[v] = list[e]  
 list[e] = list[v]  
 print ("STOP HERE")

**def** main():  
 head = [1, 3, 8]  
 **for** idx **in** range(0,1):  
 update(head, idx)

Note: This sort of question will not be appearing on the exam as I have mentioned. In its place there will be a coding question

main()

Q4. [**16 pts**.] The lastOdd function is intended to return the location of the last odd element of a non-empty list. If no value in list is odd, then it should return -1 (negative one).

|  |
| --- |
| Q4 Expected Output |
| >>> lastOdd([1,9,2,9])  **3** >>> lastOdd([2,4,6,8]) **-1** >>> lastOdd([2,3,6,8]) **1** |

**def** lastOdd(list)  
 end = len(list)  
 **while** end < 0:  
 **if** end % 2 == 1:  
 **return** end  
  
 end = end + 1

Circle four defects **and explain how you would fix them**.

1. Missing ‘:’ at the end of the function definition. Add it in
2. end = len(list) is too high; this must be end = len(list) – 1
3. While end >= 0 must be condition to while loop
4. If list[end] % 2 must be the condition of if
5. end = end – 1 must be the decrementing count
6. must have closing return -1 after while loop

|  |
| --- |
| Q5 Sample Output |
| >>> process(['a', 'b', 'c'], [1,2,3])  ['a', 1, 'b', 2, 'c', 3]  >>> process([1,2,3], [9,8,7])  [1, 9, 2, 8, 3, 7] |

Q5. [**15 pts.]** Write a process(list1, list2) function that takes two equal-length lists containing values and returns a new list that contains alternating elements from list1 and list2 respectively.

You can assume that both lists have at least one element in them.

def process(list1, list2):  
 combined = []

for idx in range(len(list1)):  
 combined.append(list1[idx])  
 combined.append(list2[idx])

return combined