Quiz 6 Writing complex python functions Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

For the following questions, write your code in the space provided, or circle the letter of the MOST correct answer.

1. Write a function called count\_odds that receives a list of integers. It counts the number of odd integers in the list, and returns the count.

|  |
| --- |
| Example calls:  a = count\_odds([ 1, 3, 5 ])  b = count\_odds([ 2, 1, 2, 3, 4 ])  c = count\_odds([ 0, 2, 2 ])  print a, b, c  Prints:  3 2 0 |

1. The function close\_far is provided for you. Write two calls to the function; once using the parameters 5, 7, -5; once using 5, 7, 14. Print the return values. The expected output is shown below. (Note: abs(x) returns the absolute value of x.)

|  |
| --- |
| Provided code:  def close\_far(a, b, c):  if abs(a-b) <= 2 and abs(a-c) >= 10:  return True  return False  Expected output:  True False |

1. Write a function named count\_digits that receives a string parameter and returns the number of digits (0-9) in the string.

|  |
| --- |
| Example calls:  a = count\_digits("april showers") # 0  b = count\_digits("10 april showers 33") # 4  c = count\_digits("3.1415926") # 8  print a, b, c  Prints:  0 4 8 |

1. Write a function named guess\_number that receives 1 numeric parameter. It asks the user for a number between 1 and 10 until the user guesses the number in the parameter. It returns the number of times the user guessed.

|  |
| --- |
| Example call:  a = guess\_number(7)  print a  Output and user input:  Guess my number between 1 and 10: **4**  Guess my number between 1 and 10: **8**  Guess my number between 1 and 10: **5**  Guess my number between 1 and 10: **7**  4 |

1. Write a function named sum\_after\_6 that receives a list of integers. It adds up all numbers that follow a 6, and returns the total.

|  |
| --- |
| Example calls:  a = sum\_after\_6( [ 1, 6, 3, 5, 6 ] )  b = sum\_after\_6( [ 6, 1, 6, 3, 6, 5 ] )  c = sum\_after\_6( [ 1, 3, 5, 6 ] )  print a, b, c  Prints:  3 9 0 |

1. Select the box that shows the output after the program is executed.

|  |
| --- |
| def problem6(s):  s2 = s[1:2] + s[3:len(s)-1] + s[:4]  return s2  a = problem6("abcdef")  b = problem6("uvwxyz")  print a, b |

|  |  |  |  |
| --- | --- | --- | --- |
| (a) abcdef zyxwvu | (b) bdeabcd vxyuvwx | (c) bcdeabcd vwxyuvwx | (d) abcd uvwx |

1. Select the box that shows the output after the program is executed.

|  |
| --- |
| def problem7(nums):  i = 0  while nums[i] < 4:  i = i + 1  j = nums[i]  return nums[j]  a = problem7([1,4,0,2,3])  print a |

|  |  |  |  |
| --- | --- | --- | --- |
| (a) 1 | (b) 4 | (c) 5 | (d) 3 |

1. Select the box that shows the output after the program is executed.

|  |
| --- |
| def problem8(a, b, c):  x = 0  if a <= b and a >= c:  x += 1  if b > c and a != b:  x += 2  elif b < a or b != c:  x += 4  return x  a = problem8(4, 5, 3)  b = problem8(6, 6, 8)  print a, b |

|  |  |  |  |
| --- | --- | --- | --- |
| (a) 7 3 | (b) 3 4 | (c) 5 3 | (d) 4 2 |

1. Select the box that shows the output after the program is executed.

|  |
| --- |
| def problem9(nums):  total = 0  for i in range(len(nums)):  if nums[i] != 13 and (i==0 or nums[i-1] != 13):  total += nums[i]  return total  a = problem9([13, 1, 2, 13, 2, 1, 13])  b = problem9([1, 2, 13, 2, 1, 13])  print a, b |

|  |  |  |  |
| --- | --- | --- | --- |
| (a) 3 4 | (b) 3 3 | (c) 6 6 | (d) 16 17 |

1. Select the box that shows the output after the program is executed.

|  |
| --- |
| def problem10(pennies, nickels, change):  max\_nickels = change / 5  max\_pennies = change % 5  if max\_nickels > nickels:  max\_pennies += 5 \* (max\_nickels - nickels)  max\_nickels = nickels  if max\_pennies > pennies:  return -1  return max\_pennies + max\_nickels  a = problem10(1, 1, 9)  b = problem10(10, 0, 9)  c = problem10(10, 1, 9)  print a, b, c |

|  |  |  |  |
| --- | --- | --- | --- |
| (a) 2 10 11 | (b) 9 9 9 | (c) 2 -1 -1 | (d) -1 9 5 |