**PYTHON PRACTICE ASSIGNMENTS**

Q1: Write the function isPower(x,y) which takes in two integers x and y. The function checks if y is a power of x. If y is a power of x, return what power it is. Example: x = 3, y = 9. 9 is a power of 3, and three must be raised to the second power, so isPower would return 2. You are not allowed to use the math module.

>>> isPower(4, 64)

3

>>> isPower(5, 81)

-1

>>> isPower(‘5’, [81])

‘error’

Q2: Write the method divisorList(aList, divisor) that takes in a list with elements of any data type and a divisor which is in the form of an integer. The output is a sorted list of only the integers that are divisible by the divisor.

>>> divisorList([1,1,1,'a',90],1)

[1, 1, 1, 90]

>>> divisorList([4,4.5,34,'Hello',77],2)

[4, 34]

>>> divisorList([4,4.5,34,{1:2, 3:4},77],”2”)

‘error’

Q3: Write the method translate(translationDict, txt) that takes in a dictionary and a string. The dictionary contains keys of strings that have a value of another string. You will be translating the input string using the key-value pairs in the dictionary to a new string that has replaced all the words in the input string with the words' value in the dictionary. Your function should convert txt to all lowercase letters before making the translations. If a word in the string is not in the translation dictionary, the word will retain its original form.

>>> myDict = {'up': 'down', 'down': 'up', 'left': 'right', 'right': 'left'}

>>> text = 'up down left right forward'

>>> translate(myDict, text)

'down up right left forward'

Q4: Write the function joinedList(n), where n is an integer number. This functions returns (not prints) the list [1,2,...,n,n,...2,1] for every n>0 and the list [n,n+1,n+2, -1, -1, -2,…,n] for every n<0. If an invalid input is provided, the function must return a string with an error message (the string ‘error’ is enough). Function returns [0] when n=0

>>> joinedList(5)

[1, 2, 3, 4, 5, 5, 4, 3, 2, 1]

>>> joinedList(-3)

[-3, -2, -1, -1, -2, -3]

>>> joinedList('5')

‘error’

Q5: Write the function removePunctuation(txt), where txt is a string. This function returns a string that replaces every character that is not an alphabet letter into a space. If a string is not provided as an input, the function must return a string with an error message (the string ‘error’ is enough)

>>> removePunctuation("I like chocolate cake!!(!! It's the best flavor..;.$ for real")

'I like chocolate cake It s the best flavor for real'

>>> removePunctuation("Dots...................... many dots..X")

'Dots many dots X'

>>> emovePunctuation(55)

‘error’

Q6: Write the function countWords(txt), where txt is a string. This function returns a dictionary whose keys are included words, and values their word counts. Count contraction words as (can’t), (don’t), (isn’t), etc. If a string is not provided as an input, the function must return a string with an error message (the string ‘error’ is enough). Hint: Remove punctuation before counting the words, careful with contraction words

>>> article1=”He will be the president of the company; right now he's a vice president. But he himself, is no sure of it... (Later he will see the importance of these 3.)”

>>> expected={'he': 3,"he's": 1, 'will': 2, 'be': 1, 'the': 3,

'president': 2, 'of': 3, 'company': 1, 'right': 1, 'now': 1, 'is': 1, 'a':

1, 'vice': 1, 'but': 1, 'himself': 1, 'no': 1, 'sure': 1, 'it': 1,

'later': 1, 'see': 1, 'importance': 1, 'these': 1}

>>> countWords(article1)==expected

True

>>> countWords(55)

'error'

>>> countWords([3.5,6])

'error'

Q7: Write the function studentGrades(gradeList) that takes a nested list with the following structure: - First list is always a descriptive header. - Subsequent lists hold all the data. - For lists that hold data, the first element is always a string, the rest of the elements are numeric values. Each list (except for the first one) represents the grades of the student and the first element of each list contains the name of the student.

grades = [ ['Student', 'Quiz 1', 'Quiz 2', 'Quiz 3'],# List 1, header ['John', 100, 90, 80], ['McVay', 88, 99, 111], ['Rita', 45, 56, 67], ['Ketan', 59, 61, 67], ['Saranya', 73, 79, 83], ['Min', 89, 97, 101]]

and returns ONE list with the average score for each student in INTEGER format (you can use round or int). If a list is not provided as an input, the function must return a string with an error message (the string ‘error’ is enough).

>>> grades = [

['Student', 'Quiz 1', 'Quiz 2', 'Quiz 3'],

['John', 100, 90, 80],

['McVay', 88, 99, 111],

['Rita', 45, 56, 67],

['Ketan', 59, 61, 67],

['Saranya', 73, 79, 83],

['Min', 89, 97, 101]]

>>> studentGrades(grades)

[90, 99, 56, 62, 78, 95]

>>> grades = [

['Student', 'Quiz 1', 'Quiz 2'],

['John', 100, 90],

['McVay', 88, 99],

['Min', 89, 97]]

>>> studentGrades(grades)

[95, 93, 93]

>>> studentGrades(55)

'error'

Q8: Write the function isPalindrome(text). This function takes a string as a parameter and returns the Boolean value True if the string is a palindrome, False otherwise. • A string is a palindrome if it is spelled the same both forward and backward. • Sentences can also be palindromes; therefore, punctuation, capitalization, and spaces should be ignored • If the user provides an input that is not a string, program should return False.

List of palindromes: <http://www.palindromelist.net/>

>>> isPalindrome("alula")

True

>>> isPalindrome("love")

False

>>> isPalindrome("Madam")

True

>>> isPalindrome(12.5)

False

>>> isPalindrome(12.21)

False

>>> isPalindrome("Cigar? Toss it in a can.! It is so tragic.")

True

>>> isPalindrome("travel.. a town in Alaska")

False