COSC2429 – Intro to Programming
Assessment 2 – Sem A 2021

Please follow the instruction carefully. Please copy the header from the Canvas site and put it at the top of your py file.

- Please write correct docstrings and code comment.
- There are a total of 4 problems. You must put all the code for all the problems below in 1 py file with your student id (e.g., s12345678.py) and upload it to Assessment 2 on Canvas once done. **DO NOT ZIP FILE PLEASE!**
- Do not include test of your function, and **NO USER INPUT PLEASE!**

Problem 1:

Write a function least_o(lst) in Python that takes one parameter, a list of strings, as input and returns the string in the list that has the fewest total occurrences of the lowercase letter 'o' (note that this may be zero occurrences). Uppercase ‘O’s should not be counted

You may assume that no two strings in the list have the same total number of 'o', that there is at least one string in the list, and that all elements of the list are valid strings.

For example: If you receive the list [‘tooo’, ‘ho’, ‘nO’, ‘loo’], you must return ‘ho’.

Function format:

```
def least_o(lst):
    # your logic here
    return s
```

Note: You can write it either as an iterative or recursive function.

Problem 2:

Write a function the take in a str and an int and return the reverse str split by the int number “almost evenly”. If the int to split by >= length of the str, you have to reverse the whole str.
Some examples:

```
Input s: abcdefghi and i: 6
Output: badcfeghi
Input s: 12345678 and i: 3
Output: 32165487
Input s: 1234567890123456789 and i: 4
Output: 5432109876543219876
Input s: kjsf8734(* %*df and i: 5
Output: sjk78f(43% *fd*
Input s: 12345 and i: 10
Output: 54321
Input s: kdjflk and i: 6
Output: klfjdk
```

Input s: I am a dumb(ledore) lecturer and i: 7
Output: ma Id a (bmuodel)ertcelreru

Function format:

```
def reverse(s, i):
        # your logic here
        return new_s
```

Problem 3:

Write a function to calculate the shipping fee on an ecommerce website. The function will take in a list and a dict, for example:

```
shopping_list = [‘apple’, ‘orange’, ‘kiwi’, ‘orange’, ‘apple’, ‘apple’]
location_dict = {
        ‘apple’: [1,3,4,7],
        ‘orange’: [2,4,5,7],
        ‘kiwi’: [1,3,5,6]
        }
```

The shopping_list includes all items the customer put in his/her cart. There can be multiple occurrences of an item, so you have to sum them up. The location_dict includes the items as keys of the dict, while the dict values are lists of warehouses locations that item is in. There are 4 warehouses. The int numbers in the list are the numbers of items in each warehouse location. Stock number can be zero in 1 warehouse, but there will definitely be enough stock for the order. If the buyer wants more items than the stock of 1 warehouse, you have to get them from a second warehouse. If there are items that can be dispatched from the same warehouse, you must combine them to save the shipping fee for the customer. If they are from different warehouses, you must calculate them as 2 shipments. The shipping fee will be 20,000VND per shipment. Return the final total shipping fee and total number of shipments.

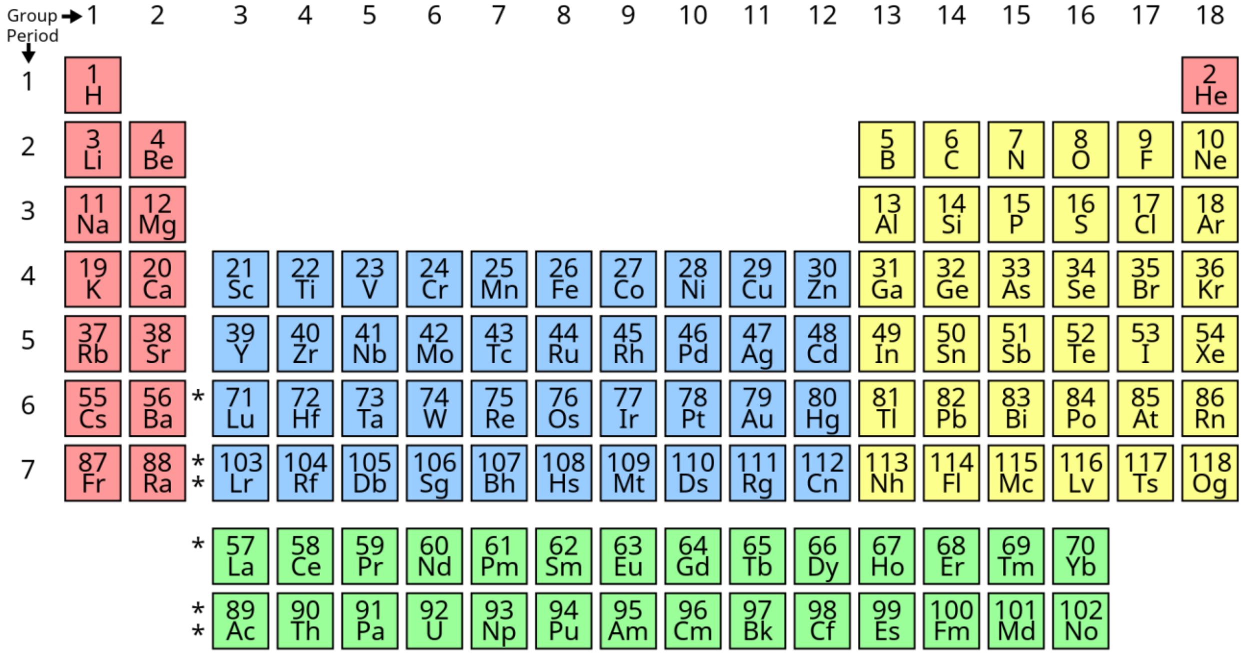
Function format:

```
def shipping_fee(shopping_list, location_dict):
        # your logic here
        return shipping_fee, number_of_shipments
```

NOTE: both returns must be int, not float.

Problem 4:

A chemical research lab wants to send some secret data to another lab. To avoid the information being stolen. They encode it using the periodic table chart:



Group → / Period ↓	1	2		3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1 H																		2 He
2	3 Li	4 Be												5 B	6 C	7 N	8 O	9 F	10 Ne
3	11 Na	12 Mg												13 Al	14 Si	15 P	16 S	17 Cl	18 Ar
4	19 K	20 Ca		21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 As	34 Se	35 Br	36 Kr
5	37 Rb	38 Sr		39 Y	40 Zr	41 Nb	42 Mo	43 Tc	44 Ru	45 Rh	46 Pd	47 Ag	48 Cd	49 In	50 Sn	51 Sb	52 Te	53 I	54 Xe
6	55 Cs	56 Ba	*	71 Lu	72 Hf	73 Ta	74 W	75 Re	76 Os	77 Ir	78 Pt	79 Au	80 Hg	81 Tl	82 Pb	83 Bi	84 Po	85 At	86 Rn
7	87 Fr	88 Ra	**	103 Lr	104 Rf	105 Db	106 Sg	107 Bh	108 Hs	109 Mt	110 Ds	111 Rg	112 Cn	113 Nh	114 Fl	115 Mc	116 Lv	117 Ts	118 Og
			*	57 La	58 Ce	59 Pr	60 Nd	61 Pm	62 Sm	63 Eu	64 Gd	65 Tb	66 Dy	67 Ho	68 Er	69 Tm	70 Yb		
			**	89 Ac	90 Th	91 Pa	92 U	93 Np	94 Pu	95 Am	96 Cm	97 Bk	98 Cf	99 Es	100 Fm	101 Md	102 No		

IMPORTANT: There can be multiple versions on this table online. Please use this one only.

The rules are:

- If you see a substring that matches one of the elements in the table, you replace it with the number and VICE VERSA.
- You prioritize the encryption of 2-character substring over 1-character one. For example, if you have the word “name”, “Na” will be encrypted, not “N”.
- However, for numbers, to keep it simple, we encrypt 1 digit at a time.
- Since the length of the replace string varies, you put them in a {} to ensure the correct encryption and decryption.
- Please keep all the punctuation and special characters the same.
- For example:

I am 15 year old. => {53} {95} {H}{B} {39}e{18} {8}ld.

Function format:

```
        def encrypt(s):
                # your logic here
                return encrypted_s
        def decrypt(encrypted_s):
                # your logic here
                return s
```

NOTE: beware of case sensitive