Упражнение 2 – Задачи Python

Зад.1 Напишете функция, която приема списък от стрингове и намира
броя на стринговете, които са с дължина >=2 и първия и последния символ съвпадат.

```
def match_ends(words):
match_ends(['aba', 'xyz', 'aa', 'x', 'bbb']) # 3
match_ends(['', 'x', 'xy', 'xyx', 'xx'])     # 2
match_ends(['aaa', 'be', 'abc', 'hello'])    # 1
```

Зад.2 Напишете функция, която приема списък от стрингове и го връща сортиран,
като всички стрингове започващи с 'x' са в началото.

```
def front_x(words):
front_x(['bbb', 'ccc', 'axx', 'xzz', 'xaa']) # ['xaa', 'xzz', 'axx', 'bbb', 'ccc'])
front_x(['ccc', 'bbb', 'aaa', 'xcc', 'xaa']) # ['xaa', 'xcc', 'aaa', 'bbb', 'ccc'])
front_x(['mix', 'xyz', 'apple', 'xanadu', 'aardvark']) # ['xanadu', 'xyz', 'aardvark', 'apple', 'mix'])
```

Зад.3 Функция - приема списък от непразни кортежи и сортира списъка в нарастващ ред на последните елементи на всеки кортеж.

```
def sort_last(tuples):
sort_last([(1, 3), (3, 2), (2, 1)])   # [(2, 1), (3, 2), (1, 3)])
sort_last([(2, 3), (1, 2), (3, 1)])   # [(3, 1), (1, 2), (2, 3)])
sort_last([(1, 7), (1, 3), (3, 4, 5), (2, 2)])  # [(2, 2), (1, 3), (3, 4, 5), (1, 7)])
```

Зад.4 Функция - приема списък от числа, премахва всички съседни повтарящи се елементи.

```
def remove_adjacent(nums):
remove_adjacent([1, 2, 2, 3])    # [1, 2, 3]
remove_adjacent([2, 2, 3, 3, 3]) # [2, 3]
```

Зад.5 Функция - слива два сортирани списъка

```
def linear_merge(list1, list2):
linear_merge(['aa', 'xx', 'zz'], ['bb', 'cc']) # ['aa', 'bb', 'cc', 'xx', 'zz']
linear_merge(['aa', 'xx'], ['bb', 'cc', 'zz']) # ['aa', 'bb', 'cc', 'xx', 'zz']
linear_merge(['aa', 'aa'], ['aa', 'bb', 'bb']) # ['aa', 'aa', 'aa', 'bb', 'bb']
```

Зад.6 Функция - приема стринг и връща стринг съдържащ първите два и последните два символа на първоначалния стринг.

```
def both_ends(s):
both_ends('spring') # 'spng'
both_ends('Hello')  # 'Helo'
both_ends('a')      # ''
both_ends('xyz')    # 'xyyz'
```

Зад.7 Функция - приема стринг и замества всички символи еднакви на първия с '*'

```
def fix_start(s):
fix_start('babble')   # 'ba**le')
fix_start('aardvark') # 'a*rdv*rk'
fix_start('google')   # 'goo*le'
fix_start('donut')    # 'donut'
```

Зад.8 Функция - приема два стринг и връща един стринг съдържащ двата разделени с интервал.
разменя първите два символа на двата стринга.
def mix_up(a, b):
mix_up('mix', 'pod')   # 'pox mid'
mix_up('dog', 'dinner') # 'dig donner'
mix_up('gnash', 'sport') # 'spash gnort'
mix_up('pezzy', 'firm') # 'fizzy perm'

Зад.9 Функция - приема стринг, ако е дължина поне 3 добавя 'ing' в края.
Освен ако не завършва на 'ing' тогава добавя 'ly'.
ако стринга е с дължина < 3 не го променя
def verbing(s):
verbing('hail')   # 'hailing'
verbing('swiming') # 'swimingly'
verbing('do')   # 'do'

Зад.10 Функция: намира първото срещане на подстринга 'not' и 'bad'.
Ако 'bad' е след 'not' замества целия израз 'not'...'bad' със стринга 'good'
def not_bad(s):
not_bad('This movie is not so bad')   # 'This movie is good'
not_bad('This dinner is not that bad!') # 'This dinner is good!'
not_bad('This tea is not hot')   # 'This tea is not hot'
not_bad("It's bad yet not")   # "It's bad yet not"

Зад.11 Функция: разделя стринг на две половини.
връща: a-front + b-front + a-back + b-back
def front_back(a, b):
front_back('abcd', 'xy')   # 'abxcdy'
front_back('abcde', 'xyz')   # 'abcxydez'
front_back('Kitten', 'Donut') # 'KitDontenut'

Зад.12 Функция: приема списък от числа, връща списък само от четните числа.
Зад.13 Функция: приема два списъка, връща списък с общите елементи на двата списъка.
Зад.14 Функция: приема число, и намира всички негови делители.
Зад.15 Функция: приема число, намира всички по-малки прости числа.