Anagram

Anagram (přesmyčka) je textový řetězec, který z původního řetězce vznikne tak, že se použijí všechna původní písmena, ale změní se jejich pořadí. Například: 'debit card' → 'bad credit'.

Napište program, který hledá dvojice anagramů.

- **Vstup:** jméno souboruje první argument příkazové řádky
  - Soubor obsahuje na každém řádku jeden textový řetězec obsahující malá/velká písmena a mezery
- **Výstup:** několik řádků nebo None na standardní výstup
  - Pokud jsou ve vstupním souboru anagramy, program vytiskne indexy řádků, které jsou anagramy, indexy se vypíšou jako celá čísla oddělená mezerou
  - Pro každou nalezenou dvojici anagramů bude jedna řádka výstupu
  - Pokud soubor žádný anagram neobsahuje, bude výstupem slovo None
- Řádky indexujeme od 0
- Výstup (dvojice čísel) musí být seřazeny vzestupně dle prvního indexu, v případě shodnosti prvního indexu jsou seřazeny vzestupně dle druhého indexu
- Příklad: pokud nalezneme shody mezi řádky: 10 a 11; 10 a 12; 10 a 13; 11 a 13; 12 a 11, pak vypíšeme:

```
10 11
10 12
10 13
11 12
11 13
```

- První číslo ve dvojici je vždy menší než druhé
- Je nutné vypsat všechny dvojice splňující výše uvedené podmínky.
- Program nahrajte do Bruta v souboru **anagram.py** — úloha **QTAnagram**
- Je zaručeno, že vstupní soubor existuje a obsahuje alespoň jednu řádku.
- Mezery se při určování anagramů neuvažují.
- Při určování anagramů je třeba rozlišovat velikost písmen (anagramy jsou case-sensitive).
- Nápověda: složitost řešení je vzhledem k počtu řetězců (a jejich délce) kvadratická nebo lepší.

Příklady anagramů:

- 'abc', anagramem je např. 'bca', 'cab', 'c a b', ' a c b' (mezery se neuvažují)
- 'abc' a ' a a b c ' nejsou anagramy (nesouhlasí počet písmen)
- 'Ac' a 'CA' nejsou anagramy (case-sensitive)
- 'forty five', příklad anagramu: 'over fifty' nebo 'overfifty' nebo 'fiftyover'

Bodování:

Jednotlivé části testů budou spuštěny až po dosažení plného počtu bodů v předchozí části, např. testování příkladů typu 'None' bude provedeno až poté, co všechny předchozí testy dopadnou správně.

Příklad I Volání `python3 anagram.py abc.txt`
Obsah souboru `abc.txt`

```
abc
Abc
a c b
   cA b
```

Výstup

```
0 2
1 3
```

Příklad II Volání `python3 anagram.py simple.txt`
Obsah souboru `simple.txt`

```
tacts
lemur
bonny
argue
asttc
ruelm
onbyn
gaeru
```

Výstup

```
0 4
1 5
2 6
3 7
```

Příklad III Volání `python3 anagram.py long.txt`
Obsah souboru `long.txt`

```
Cockpit
Destroy
Firedog
Concern
Fiction
Postbox
Peanuts
Percent
Network
```

Výstup

```
None
```

Popis části	**Počet testů**	**Timeout**	**Max. bodů**	**Hodnocení**
Krátké I (do 4 řádků, bez mezer)	50	celkem 2 s	**1**	0.02b/test
Krátké II (do 4 řádků, včetně mezer)	50	celkem 2 s	**1**	0.02b/test
Slovník (do 10 řádků, anglická slova)	50	celkem 2 s	**1**	0.02b/test
Náhodné (do 20 řádků, náhodné řetězce do 100 znaků)	100	celkem 2 s	**2**	0.02b/test
None (do 50 řádků, řešením je 'None')	10	celkem 2 s	**1**	0.1b/test

Příklad IV Volání `python3 anagram.py long2.txt`
Obsah souboru `long2.txt`

```
Seaward
Pension
Thirsty
Unlevel
Helpful
Dynamic
Abalone
Kennedy
Equinox
Comfort
AbalonePensionDynamicHelpfulComfortSeawardKennedyUnlevelThirsty Equinox
KennedySeawardHelpfulDynamic EquinoxComfortUnlevelPension  ThirstyAbalone
ThirstyAbaloneUnlevelKennedySeaward  DynamicHelpful EquinoxComfortPension
AbaloneDynamicSeawardPensionEquinoxThirstyComfortUnlevel HelpfulKennedy
KennedyAbalone ThirstyComfortPension DynamicEquinoxSeaward UnlevelHelpful
AbaloneDynamicThirstyKennedyUnlevelEquinoxHelpful Seaward Pension Comfort
ThirstyAbalonePension KennedyDynamicEquinox  HelpfulUnlevelSeawardComfort
KennedyUnlevelEquinoxComfortSeawardAbalonePension HelpfulDynamic Thirsty
```

Výstup

```
10 11
10 12
10 13
10 14
10 15
10 16
10 17
11 12
11 13
11 14
11 15
11 16
11 17
12 13
12 14
12 15
12 16
12 17
13 14
13 15
13 16
13 17
14 15
14 16
14 17
15 16
15 17
16 17
```

Komentář Jeden řetězec může samozřejmě být anagramem několika jiných řetězců. V tomto případě řádek 10 (číslujeme od nuly) je anagramem všech dalších řádků (11,12,...,17) a stejně tak řádek číslo 11 je anagramem těch následujících. Jelikož vypisujeme tak, že první číslo je menší než druhé, vypíšeme pouze kombinaci '10 11', zatímco '11 10' ne.