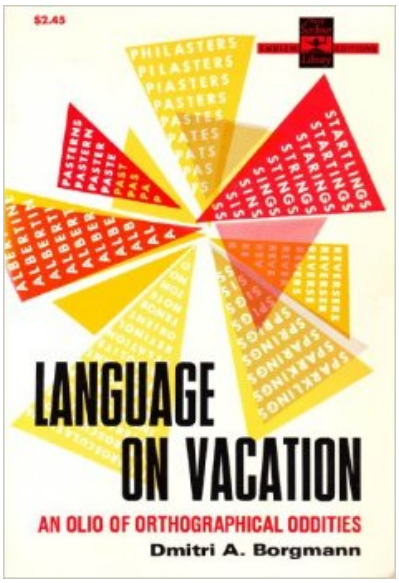
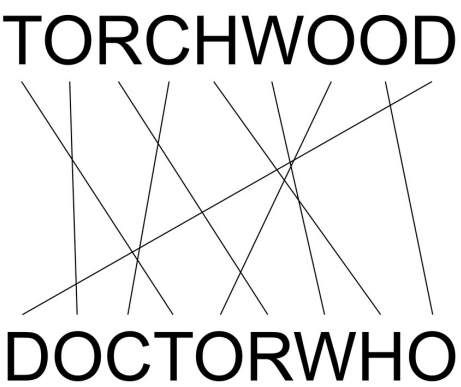
Isograms

An **isogram** is a word in which no letter is repeated. Examples of isograms are *uncopyrightable* and *ambidextrously*. Conveniently, the word itself is an isogram.

Theoretically the limit is 26 letters, but that's an Everest that no one has scaled. In his book Language on Vacation: An Olio of Orthographical Oddities, Dmitri Borgmann has conquered some lesser peaks trying to find the longest isogrammic word. The longest one he found was *dermatoglyphics* at 15 letters. He coins several longer hypothetical words, such as *thumbscrew-japingly* (18 letters, defined as "as if mocking a thumbscrew") and — with the uttermost limit in the way of verbal creativeness — *pubvexingfjord-schmaltzy* (23 letters, defined as "as if in the manner of the extreme sentimentalism generated in some individuals by the sight of a majestic fjord, which sentimentalism is annoying to the clientele of an English inn"). Maybe what we lack is imagination.



An **anagram** is a type of word play, the result of rearranging the letters of a word or phrase to produce a new word or phrase, using all original letters exactly once. For example, the word *Torchwood* can be rearranged into *Doctor Who*. Any word or phrase that exactly reproduces the letters in another order is an anagram. However, the goal of serious or skilled anagrammatists is to produce anagrams that in some way reflect or comment on the subject. Such an anagram may be a synonym or antonym of its subject, a parody, a criticism, or praise. For example, *William Shakespeare* is an anagram of *I am a weakish speller*.



Assignment

In this exercise, words are represented as strings that only contain letters. Your task is to implement the following three functions, that each take one or two words. All functions must treat the given words in a case insensitive way.

- Write a function `occurrences` that takes a single word. The function must return a list of 26 integers, each indicating the number of occurrences in the given word of the letter at the corresponding position in the alphabet.
- Use the function `occurrences` to write a function `isogram` that takes a single word. The function must return a Boolean value that indicates whether or not the given word is an isogram.
- Use the function `occurrences` to write a function `anagram` that takes two words. The function must return a Boolean value that indicates whether or not the given words are anagrams of each other.

Example

```
>>> occurrences('ambidextrously')
[1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0]
>>> occurrences('DOCTORWHO')
[0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 3, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0]
>>> occurrences('uncopyrightable')
[1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0]

>>> isogram('ambidextrously')
True
>>> isogram('DOCTORWHO')
False
>>> isogram('uncopyrightable')
True

>>> anagram('DOCTORWHO', 'Torchwood')
True
>>> anagram('isogram', 'anagram')
False
>>> anagram('Aristotelian', 'retaliations')
True
```

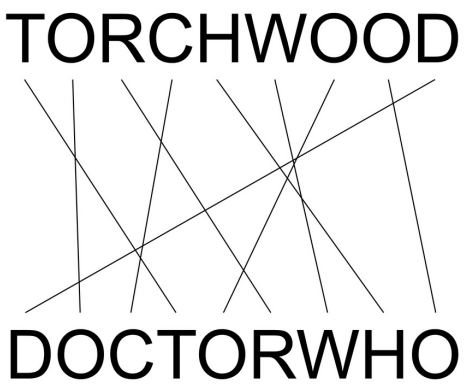
Een **isogram** is een woord waarin elke letter van het alfabet hoogstens één keer voorkomt. Voorbeelden van isogrammen zijn *filmproducent* en *vragenlijst*. De maximale lengte van een isogram in het Latijnse alfabet is 26 letters, maar er zijn geen voorbeelden bekend van woorden die deze theoretische grens bereiken. De langste Nederlandse isogrammen die in Van Dale of het Groene Boekje voorkomen, tellen 15 letters: *campinghoudster*, *dampkringslucht*, *sandwichformule* en *whiskyproducent*.

Onder de liefhebbers van het Opperlans worden isogrammen vaak aangeduid als *prachtsymboliek*, wat zelf namelijk ook een isogram is. Zoals bij elk Opperlans onderwerp, moet er onderscheid gemaakt worden tussen het langste isogram dat daadwerkelijk is waargenomen in normaal taalgebruik (*landbouwgeschrift*, 17 letters), en het langste isogram dat na lang puzzelen en een ingewikkeld verhaal verdedigbaar wordt gemaakt.

Van de 308 gemeenten in Vlaanderen zijn er 42 waarvan de naam een isogram vormt, waarvan *Zwijndrecht* de langste is met 11 verschillende letters. Van de 262 gemeenten in Wallonië zijn er

75 waarvan de naam een isogram vormt, waarvan *Wadelincourt* de langste is met 12 verschillende letters. Van de 19 gemeenten in het Brussels Gewest is *Vorst* de enige wiens naam een isogram vormt.

Een **anagram** is een woord dat gevormd wordt uit de letters van een ander woord maar in een andere volgorde. Het maken van anagrammen is een geliefd woordspel. Anagrammen worden ook vaak gebruikt als pseudoniem.



Opgave

In deze opgave worden woorden voorgesteld als strings die enkel bestaan uit letters. Je opdracht bestaat erin de volgende drie functie te schrijven, waaraan steeds één of twee woorden moeten doorgegeven worden. De functies mogen geen onderscheid maken tussen hoofdletters en kleine letters.

- Schrijf een functie voorkomens waaraan een woord moet doorgegeven worden. De functie moet een lijst van 26 natuurlijk getallen teruggeven, waarbij elk getal aangeeft hoeveel keer de letter op de corresponderende plaats in het alfabet voorkomt in het gegeven woord.
- Gebruik de functie voorkomens om een functie isogram te schrijven waaraan een woord moet doorgegeven worden. De functie moet een Booleaanse waarde teruggeven, die aangeeft of het gegeven woord een isogram is.
- Gebruik de functie voorkomens om een functie anagram te schrijven waaraan twee woorden moet doorgegeven worden. De functie moet een Booleaanse waarde teruggeven, die

aangeeft of de gegeven woorden anagrammen van elkaar zijn.

Voorbeeld

```
>>> voorkomens('filmproducent')
[0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0]
>>> voorkomens('DOCTORWHO')
[0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 3, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0]
>>> voorkomens('whiskyproducent')
[0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0]

>>> isogram('filmproducent')
True
>>> isogram('DOCTORWHO')
False
>>> isogram('whiskyproducent')
True

>>> anagram('DOCTORWHO', 'Torchwood')
True
>>> anagram('isogram', 'anagram')
False
>>> anagram('CENTRALISERENDE', 'decentraliseren')
True
```