Word zippers

In a game of word zippers, you get two words A and C, and the aim is to place a new word B in between those words. The word B must be chosen so that the first two words AB and the last two words BC each form a new word again. The length of the word B is also given. E.g. the four letter word type fits between the words hydro and writer, which results in the compound words hydrotype and typewriter.

Assignment

- Write a function

  readWords(filename)

  that takes the location of a text file as its argument. This text file must contain a list of all words that are seen as valid. Each of these words must be on a separate line. The file wordlist.txt is used in the examples of this assignment, and can also be used to test your solution. The function must print a (compound) object as a result, that contains all valid words from the file. The choice of the valid data type for this object is made in function of the speed with which the function wordzippers (see below) is executed. A less suitable choice will lead to a solution that can not determine a sequence of word zippers within a set time limit.
- Write a function

  wordzippers(zipper, words)

  that prints all valid words with the given length that fit between the given words. As a first parameter, a string with the format word-....-word must be given to the function. The first given word is in front of the first hyphen, and the second given word comes after the second hyphen. The words do not contain any hyphens themselves. The number of full stops that is between the two hyphens, gives the length of the wanted word that fits between the two given words. The second parameter that is given to the function is a (compound) object that contains all words that are considered valid. When searching the word that can form a word zipper, no distinction should be made between uppercase and lowercase letters, neither in the words from the given file, nor in the word that was given in the first parameter.
  As a result, the function must print a string, with the format word-WORDS-word. The words before and after the hyphens are the given words, and are written in lowercase letters. The words between the hyphens are the words that form a valid word zipper. These words must be written in uppercase letters. If there are multiple words between the hyphens, they must be listed alphabetically and separated by commas. If no words can be found, three question marks must be placed in between the hyphens.

Example

```
>>> words = readWords('wordlist.txt')
>>> wordzippers('gyne-....-wrote', words)
'gyne-TYPE-wrote'
>>> wordzippers('hydro-....-writer', words)
'hydro-TYPE-writer'
>>> wordzippers('java-.....-python', words)
'java-???-python'
```

```
>>> wordzippers('agit-....-wood', words)
'agit-PROP-wood'
>>> wordzippers('arch-...-thing', words)
'arch-SEE-thing'
>>> wordzippers('frog-...-puller', words)
'frog-LEG-puller'
>>> wordzippers('arche-....-wrote', words)
'arche-TYPE-wrote'
```

Bij een spelletje woordritsen krijg je twee woorden A en C gegeven, waartussen je een nieuw woord B moet zien te plaatsen. Het woord B moet zo gekozen worden, dat de eerste twee woorden AB en de laatste twee woorden BC samen weer een nieuw woord vormen. De lengte van het gezocht woord B wordt daarbij ook gegeven. Zo past het zesletterwoord staart tussen de woorden paarden en deling, wat resulteert in de samengestelde woorden paardenstaart en staartdeling.

Opgave

- Schrijf een functie

  leesWoorden(bestandsnaam)

  waaraan de locatie van een tekstbestand moet doorgegeven worden. Dit tekstbestand moet een lijst van alle woorden bevatten die als geldig beschouwd worden. Elk van deze woorden moet hierbij op een afzonderlijke regel staan. Het bestand woordenlijst.txt wordt gebruikt bij de voorbeelden van deze opgave, en kan ook gebruikt worden om je oplossing te testen. De functie moet een (samengesteld) object als resultaat teruggeven, dat alle geldige woorden uit het bestand bevat. De keuze van het geschikte gegevenstype voor dit object maak je in functie van de snelheid waarmee de functie woordenrits (zie hieronder) uitgevoerd wordt. Een minder geschikte keuze zal immers leiden tot een oplossing die een reeks woordritsen niet kan bepalen binnen de vooropgestelde tijdslimiet.
- Schrijf een functie

  woordritsen(rits, woorden)

  die alle geldige woorden met gegeven lengte teruggeeft die passen tussen twee opgegeven woorden. Als eerste parameter moet aan de functie een string doorgegeven worden, met als formaat woord-....-woord. Het eerste gegeven woord staat hierbij voor het eerste koppelteken, en het tweede gegeven woord na het tweede koppelteken. De woorden zelf bevatten geen koppeltekens. Het aantal punten dat tussen de twee koppeltekens staat, geeft de lengte aan van het gezochte woord dat tussen de twee gegeven woorden past. De tweede parameter die aan de functie moet doorgegeven worden is een (samengesteld) object dat alle woorden bevat die als geldig beschouwd worden. Bij het opzoeken van woorden die een woordenrits kunnen vormen, moet geen onderscheid gemaakt worden tussen hoofdletters en kleine letters, noch in de woorden uit het opgegeven bestand, noch in de opgegeven woorden van de eerste parameter.
  Als resultaat moet de functie een string teruggeven, met als formaat woord-WOORDEN-woord. De woorden voor en na de koppeltekens zijn de opgegeven woorden, en worden met kleine letters weergegeven. De woorden tussen de twee koppeltekens zijn de woorden die een geldige woordenrits vormen. Deze woorden moeten met hoofdletters weergegeven worden. Indien er meerdere woorden zijn die een geldige woordrits vormen, dan moeten ze in alfabetische volgorde opgelijst worden en van elkaar gescheiden worden door komma's.

Indien er geen woorden zijn die een geldige woordrits vormen, dan moeten drie vraagtekens tussen de twee koppeltekens geplaatst worden.

Voorbeeld

```
>>> woorden = leesWoorden('woordenlijst.txt')
>>> woordritsen('aal-....-tuin', woorden)
'aal-MOES-tuin'
>>> woordritsen('donder-...-blad', woorden)
'donder-DAG,KOP-blad'
>>> woordritsen('java-.....-python', woorden)
'java-???-python'
>>> woordritsen('paarden-.....-deling', woorden)
'paarden-STAART-deling'
>>> woordritsen('reken-.....-formule', woorden)
'reken-WONDER-formule'
>>> woordritsen('veld-.....-veld', woorden)
'veld-BONEN,PROEF,SPORT,TEKEN,ZICHT-veld'
>>> woordritsen('woord-.....-bloem', woorden)
'woord-KUNST-bloem'
```