Partitioning the phone book

Back in the days, phone numbers were looked up in a **phone book**: a long listing of telephone subscribers in a geographical area, with subscriber names listed in alphabetical order. The advent of the Internet and smart phones in the 21st century greatly reduced the need for paper phone books.

Opened view of the white pages.

Assignment

Because the list of phone numbers could grow quite long, phone books were traditionally subdivided into multiple volumes. Each of these volumes contained the phone numbers of subscribers whose names start with a sequence letters from the alphabet. In this assignment we will use two ways of describing a partitioning of a phone book.

A first **string representation** describes each volume of a partitioning by two uppercase letters that are separated by a dash. These uppercase letters are the first and the last letter in the sequence of letters of the alphabet with which the names in the volume start. The successive volumes are separated from each other by a single space. Here's an example string representation of a partitioning of a phone book:

A-D E-J K-O P-Z

The string representation should describe a partitioning of the entire alphabet and the volumes must be consecutive. As a shorthand, volumes that are limited to names starting with a single letter are simply represented by this one letter. As a consequence, we do not write D-D but D.

A second **numerical representation** describes a partitioning of a phone book as a tuple of strictly positive integers. Each of these integers indicates the number of letters in the sequence of letters of the alphabet with which the names in the volume start. For example, the partitioning that was used as an example above can also be described as $(4, 6, 5, 11)$. It must hold that the sum of the numbers of the numerical representation equals 26.

Say that we know the distribution of the first letters of telephone subscriber's last names:

A B C D E F G H I J K L M N 0 P Q R S T U V W X Y Z
16 4 17 10 15 4 4 6 7 14 9 17 27 6 1 9 0 12 20 8 0 3 4 0 3 4

In total there are 220 names and the telephone company has decided to print 4 volumes. Ideally, each of these volumes should thus contain $\frac{220}{4} = 55$ names. One possible partitioning is A-D E-J K-O P-Z with counts of 47, 50, 60 and 63 names. In absolute value these counts deviate 8, 5, 5 and 8 names from the ideal 55 names per volume. We refer to the total sum of these deviations as the **delta** of the partitioning of the phone book. For this partitioning, the delta is thus equal to 26. Another partitioning is A-G H-O P Q-Z, with name counts of 70, 87, 9 and 54, deviations of 15, 32, 46 and 1 from the ideal 55 names per volume, and a delta of 94. We might ask you to look for the optimal partitioning (the one having the smallest delta), but this would lead us to far. You are asked to:

- Write a function volumes that takes the numerical representation of a partitioning of a phone book. If the argument passed to the function does not represent a valid numerical representation of a partitioning of a phone book, the function must raise an AssertionError with the message invalid partitioning. Otherwise, the function must return the string representation of the given partitioning.
- Write a function counts that takes the string representation of a partitioning of a phone book. If the argument passed to the function does not represent a valid string representation of a partitioning of a phone book, the function must raise an AssertionError with the message invalid partitioning. Otherwise, the function must return the numerical representation of the given partitioning.
- Write a function delta that takes two arguments: a partitioning of a phone book and a list or tuple containing the distribution of the first letters of telephone subscriber's last names. If the first argument does not represent a valid numerical or string representation of a partitioning of a phone book, the function must raise an AssertionError with the message invalid partitioning. Otherwise, the function must return the delta of the given partitioning of the phone book for the given distribution of the first letters of telephone subscriber's last names.

Examples

```
>>> volumes((4, 6, 5, 11))
'A-D E-J K-O P-Z'
>>> volumes((7, 8, 1, 10))
'A-G H-O P Q-Z'
>>> volumes((4, 7, 5, 10))
'A-D E-K L-P Q-Z'
>>> volumes((8, 3, 9, 7))
Traceback (most recent call last):
AssertionError: invalid partitioning

>>> counts('A-D E-J K-O P-Z')
(4, 6, 5, 11)
>>> counts('A-G H-O P Q-Z')
(7, 8, 1, 10)
>>> counts('A-D E-K L-P Q-Z')
(4, 7, 5, 10)
>>> counts('A-D F-K L-P Q-Z')
Traceback (most recent call last):
AssertionError: invalid partitioning

>>> names = (16, 4, 17, 10, 15, 4, 4, 6, 7, 14, 9, 17, 27, 6, 1, 9, 0, 12, 20, 8, 0, 3, 4, 0, 3, 4)
>>> delta('A-D E-J K-O P-Z', names)
26.0
>>> delta((7, 8, 1, 10), names)
```

```
94.0
>>> delta('A-D E-K L-P Q-Z', names)
18.0
>>> delta(42, names)
Traceback (most recent call last):
AssertionError: invalid partitioning
```

In de goeie oude tijd werden telefoonnummers nog opgezocht in een **telefoonboek**: een grote verzameling telefoonnummers die alfabetisch gerangschikt staan op naam. Sinds deze informatie ook via het Internet te raadplegen is, is het gebruik van de papieren versie sterk afgenomen.

Geopend telefoonboek.

In België werd het eerste telefoonboek uitgegeven in 1884. Later werden de telefoonboeken uitgegeven door de Regie van Telegrafie en Telefonie (RTT) en Belgacom. In 1968 verwierf I.T.T.-Promedia de concessie voor het uitgeven van telefoonboeken. Hun Gouden Gids verscheen in 1969, eerst in Antwerpen en Luik. Een jaar later was deze cyclus rond en ontving elke telefoonabonnee zijn Gouden Gids en Witte Gids.

Een conflict tussen Belgacom en Promedia — bekend als de *Belgische telefoonboekenoorlog* — leidde ertoe dat de abonnees tussen 1995 en 1998 vier telefoongidsen ontvingen: een gouden en een witte gids van Belgacom én een gouden en een witte gids van I.T.T.. Na het oplossen van dit conflict ging Promedia — in 2007 omgedoopt tot Truvo — verder met de uitgave en distributie van beide telefoonboeken.

Opgave

Omdat de lijst van telefoonnummers soms heel groot kon worden, werden telefoonboeken traditioneel opgedeeld in meerdere volumes. Daarbij bevatte elk volume de telefoonnummers van de namen die beginnen met een reeks opeenvolgende letters uit het alfabet. In deze opgave zullen we twee manieren gebruiken om een **opdeling van een telefoonboek** te omschrijven.

Een eerste manier bestaat uit een **stringvoorstelling** waarin elk volume wordt weergegeven als twee hoofdletters gescheiden door een koppelteken, waarbij de hoofdletters de eerste en de laatste letter aangeven van de reeks opeenvolgende letters uit het alfabet waarmee de namen in het volume beginnen. De verschillende volumes worden van elkaar gescheiden door één enkele spatie. Op die manier kan een opdeling van een telefoonboek bijvoorbeeld omschreven worden door de string

A-D E-J K-O P-Z

Bij deze omschrijving moet gelden dat het volledige alfabet wordt opgedeeld, en dat de volumes ook op elkaar volgen. Bovendien worden volumes die beperkt zijn tot namen die beginnen met één enkele letter kortweg voorgesteld door die ene letter. We schrijven dus niet D-D maar gewoonweg D.

Een tweede **numerieke voorstelling** stelt de opdeling van een telefoonboek voor als een tuple van strikt positieve natuurlijke getallen, waarbij elk getal staat voor het aantal letters in de reeks opeenvolgende letters uit het alfabet waarmee de namen in het volume beginnen. Zo kan de opdeling die we hierboven als voorbeeld gebruikt hebben ook omschreven worden als $(4, 6, 5, 11)$. Er moet gelden dat de som van de getallen van een numerieke voorstelling gelijk is aan 26.

Stel nu dat we het aantal namen kennen per beginletter, bijvoorbeeld:

A B C D E F G H I J K L M N 0 P Q R S T U V W X Y Z
16 4 17 10 15 4 4 6 7 14 9 17 27 6 1 9 0 12 20 8 0 3 4 0 3 4

In totaal zijn er 220 namen, en de telefoonmaatschappij heeft beslist om een editie uit te brengen met 4 volumes. Idealiter bevat elk volume dus $\frac{220}{4} = 55$ namen. Een mogelijke verdeling is A-D E-J K-O P-Z met respectievelijk 47, 50, 60 en 63 namen. In absolute waarde wijken deze volumes dus 8, 5, 5 en 8 namen af van de ideale 55. We noemen de totale som van deze afwijkingen de **delta** van de opdeling van het telefoonboek. In dit geval bedraagt de delta dus 26. Een andere opdeling A-G H-O P Q-Z met aantallen namen 70, 87, 9 en 54 en afwijkingen van respectievelijk 15, 32, 46 en 1 van de ideale 55 namen, levert dus een delta van 94 op. We zouden op zoek kunnen gaan naar de opdeling van het telefoonboek met de kleinst mogelijke delta, maar dat zou iets te ingewikkeld zijn. Gevraagd wordt:

- Schrijf een functie volumes waaraan de numerieke voorstelling van een opdeling van een telefoonboek moet doorgegeven worden. Indien het argument dat aan deze functie wordt doorgegeven geen geldige numerieke voorstelling van een opdeling is, dan moet de functie een AssertionError opwerpen met de boodschap ongeldige opdeling. Anders moet de functie de stringvoorstelling van de gegeven opdeling teruggeven.
- Schrijf een functie aantallen waaraan de stringvoorstelling van een opdeling van een telefoonboek moet doorgegeven worden. Indien het argument dat aan deze functie wordt doorgegeven geen geldige stringvoorstelling van een opdeling is, dan moet de functie een AssertionError opwerpen met de boodschap ongeldige opdeling. Anders moet de functie de numerieke voorstelling van de gegeven opdeling teruggeven.
- Schrijf een functie delta waaraan twee argumenten moeten doorgegeven worden: een opdeling van een telefoonboek, en een lijst of tuple met de aantallen namen die beginnen met de opeenvolgende letters van het alfabet. Indien het eerste argument geen geldige numerieke of stringvoorstelling van een opdeling is, dan moet de functie een AssertionError opwerpen met de boodschap ongeldige opdeling. Anders moet de functie de delta teruggeven van de gegeven opdeling voor het gegeven aantal namen per letter van het alfabet.

Voorbeeld

```
>>> volumes( )
'A-D E-J K-O P-Z'
>>> volumes((7, 8, 1, 10))
'A-G H-O P Q-Z'
```

```
>>> volumes((4, 7, 5, 10))
'A-D E-K L-P Q-Z'
>>> volumes((8, 3, 9, 7))
Traceback (most recent call last):
AssertionError: ongeldige opdeling

>>> aantallen('A-D E-J K-O P-Z')
(4, 6, 5, 11)
>>> aantallen('A-G H-O P Q-Z')
(7, 8, 1, 10)
>>> aantallen('A-D E-K L-P Q-Z')
(4, 7, 5, 10)
>>> aantallen('A-D F-K L-P Q-Z')
Traceback (most recent call last):
AssertionError: ongeldige opdeling

>>> namen = (16, 4, 17, 10, 15, 4, 4, 6, 7, 14, 9, 17, 27, 6, 1, 9, 0, 12, 20, 8, 0, 3, 4, 0, 3, 4)
>>> delta('A-D E-J K-O P-Z', namen)
26.0
>>> delta((7, 8, 1, 10), namen)
94.0
>>> delta('A-D E-K L-P Q-Z', namen)
18.0
>>> delta(42, namen)
Traceback (most recent call last):
AssertionError: ongeldige opdeling
```