Buzzword bingo

Buzzword bingo (also known as **bullshit bingo**) is a game that finally achieved wild popularity on 22 February 1994, when it was featured in a Dilbert comic strip in which the characters play during an office meeting.

Buzzword bingo is generally played in situations where audience members feel that the speaker, in an effort to mask a lack of actual knowledge, is relying too heavily on buzzwords rather than providing relevant details. Lectures by boring professors or business meetings led by guest speakers or notable company personalities from higher up the pay scale are often viewed as a good opportunity for buzzword bingo, as the language used by these speakers often includes predictable references to arcane concepts, which are perfect for use in the creation of buzzword bingo cards. The idea is that each participant prepares a list of stop words or buzzwords that he expects the speaker will use. These words are written on a bingo card containing a square grid of n rows and n columns ($n \in \mathbb{N}$). For example, the following bingo card could be used to play buzzword bingo in a lecture about genetics.

BINGO			
cell	bacteria	PCR	virus
allele	chromosome	DNA	meiosis
protein	phenotype	gene	mutation
genome	reccesive	RNA	mitosis

A participant may tick off a word on his card if it is uttered by the speaker. The goal of the game is to tick off all words in a given row or column, and then yell "Bingo!". Having the courage to actually yell "Bingo!" is an important element of the game. In order to avoid the reprimands that would likely result from doing so, participants may resort to looking at one another and silently mouthing the word "Bingo". An alternate variation requires the person who has achieved bingo to raise his or her hand and use the word "Bingo" within the context of a comment or question.

Assignment

Write a class BuzzBingo that can be used to instantiate objects that represent the bingo card as used in a buzzword bingo game. The objects of the class BuzzBingo must at least support the following methods.

- An initialization method that takes an integer $n \in \mathbb{N}$ and a sequence (a list or a tuple) of words. The method must assure that the newly created object of the class BuzzBingo represents a bingo card whose $n \times n$ grid is filled with the given sequence of words from left to right and from top to bottom, that allows to cancel words at a later stage. Of course, upon initialization none of the words on the bingo cards is cancelled. In case the number of words in the given sequence does not correspond to the number of squares in the grid, the initialization method must raise an AssertionError with the message invalid card.
- The methodes __str__ en __repr__ that both return the same string representation of the object. This string represents a rectangular grid that indicates which words on the bingo card have been cancelled. Words that have been cancelled are represented by the letter x and words that have not been cancelled are represented by a dash (-).
- A method cancelWord that can be used to cancel a word on the bingo card. The word to be cancelled must passed as an argument to the method. Apart from the fact that the method must make sure that the object remembers that the given word has been cancelled, it must also return a tuple (r, c) that indicates that the given word was found on row r and column c on the bingo card. The rows of the grid are indexed from left to right, and the columns from top to bottom, starting at zero. In case the given word is not found on the bingo card, the method must raise an AssertionError with the message *word* not found on card. In case the given word was already cancelled, the method must raise an AssertionError with the message *word* was already cancelled. In both messages, the given word must be filled in at the position of *word*.
- A method cancelWords that can be used to cancel a sequence of words on the bingo card. The words to be cancelled must passed to the method as a list or a tuple. The method must return a list containig the position where each of the given words was found on the bingo card. The same tuple returned by the method cancelWord must be used to indicate where a word was found on the card.
- A method won that can be used to check whether the game has been won based on the words cancelled on the bingo card. The method must return a Boolean value that indicates whether or not the bingo card has a row or a column in which all words have been cancelled.

Example

```
>>> bingo = BuzzBingo(4, [
...     'cell', 'bacteria', 'PCR', 'virus',
...     'allele', 'chromosome', 'DNA', 'meiosis',
...     'protein', 'phenotype', 'gene', 'mutation',
...     'genome', 'recessive', 'RNA', 'mitosis'
... ])
>>> bingo
----
----
----
----
>>> bingo.won()
```

```
False

>>> bingo.cancelWord('phenotype')
(2, 1)
>>> print(bingo)
----
----
-x--
----
>>> bingo.won()
False

>>> bingo.cancelWord('dominant')
Traceback (most recent call last):
AssertionError: dominant not found on card

>>> bingo.cancelWord('phenotype')
Traceback (most recent call last):
AssertionError: phenotype was already cancelled

>>> bingo.cancelWords(['PCR', 'chromosome', 'protein'])
[(0, 2), (1, 1), (2, 0)]
>>> bingo
--x-
-x--
xx--
----
>>> bingo.won()
False

>>> bingo.cancelWords(('recessive', 'bacteria', 'mitosis'))
[(3, 1), (0, 1), (3, 3)]
>>> print(bingo)
-xx-
-x--
xx--
-x-x
>>> bingo.won()
True

>>> bingo = BuzzBingo(2, ('cell', 'bacteria', 'PCR'))
Traceback (most recent call last):
AssertionError: invalid card
>>> bingo = BuzzBingo(2, ['cell', 'bacteria', 'PCR', 'virus', 'allele'])
Traceback (most recent call last):
AssertionError: invalid card
```

Het spelletje **buzzword bingo** (ook bekend onder de naam **bullshit bingo**) kreeg vooral bekendheid toen het in 1994 werd gebruikt in een strip van Dilbert.

Buzzword bingo wordt vaak gespeeld om de tijd te doden tijdens saaie lessen, vergaderingen of andere bijeenkomsten waarin naar het oordeel van de luisteraars te veel geleuterd wordt. Het idee is dat elke deelnemer voorafgaand aan een les of vergadering een lijstje maakt van stopwoorden of buzzwords waarvan hij verwacht dat de spreker die zal gebruiken. Deze woorden worden op een bingokaart ingevuld in een vierkant rooster met n rijen en n kolommen ($n \in \mathbb{N}$). Voor een les genetica zou je bijvoorbeeld de volgende de kaart kunnen opstellen.

BINGO			
cell	bacteria	PCR	virus
allele	chromosome	DNA	meiosis
protein	phenotype	gene	mutation
genome	reccesive	RNA	mitosis

Als een deelnemer de spreker een woord hoort zeggen dat hij op zijn kaart heeft staan, dan mag het woord geschrapt worden. Wie als eerste alle woorden op een rij of een kolom van zijn kaart geschrapt heeft, en daarna heel hard "Bingo!" roept, heeft het spelletje gewonnen.

Opgave

Schrijf een klasse BuzzBingo waarmee objecten kunnen aangemaakt worden die een bingokaart voorstellen waarmee een spelletje buzzword bingo kan gespeeld worden. De objecten van de klasse BuzzBingo moeten minstens de volgende methoden ondersteunen.

- Een initialisatiemethode waaraan een getal $n \in \mathbb{N}$ en een reeks (een lijst of een tuple) woorden moeten doorgegeven worden. De methode moet ervoor zorgen dat het nieuw aangemaakte object van de klasse BuzzBingo een bingokaart voorstelt waarvan het $n \times n$ rooster van links naar rechts en van boven naar onder gevuld met de gegeven reeks woorden, en waarbij de woorden later kunnen geschrapt worden. Bij initialisatie is er op de bingokaart uiteraard nog geen enkel woord geschrapt. Als het aantal woorden in de

gegeven reeks niet overeenkomt met het aantal vakjes in het rooster, dan moet de initialisatiemethode een AssertionError opwerpen met de boodschap ongeldige kaart.

- De methoden __str__ en __repr__ die beide dezelfde stringvoorstelling van het object teruggeven. Deze stringvoorstelling heeft de vorm van een vierkant rooster dat aangeeft welke woorden op de bingokaart reeds geschrapt zijn. Woorden die geschrapt zijn, worden aangeduid met de letter x en woorden die nog niet geschrapt zijn, worden aangeduid met een koppenteken (-).
- Een methode schrapWoord die kan gebruikt worden om op de bingokaart een woord te schrappen. Het te schrappen woord moet als argument aan de methode doorgegeven worden. Naast het feit dat de methode ervoor moet zorgen dat het object bijhoudt dat het gegeven woord geschrapt werd, moet de methode ook een tuple (r, k) teruggeven dat aangeeft dat het gegegeven woord gevonden werd op rij r en kolom k op de bingo kaart. Hierbij worden de rijen van het rooster oplopend genummerd van links naar rechts, en de kolommen oplopend van boven naar onder. De nummering begint telkens vanaf nul. Indien het gegeven woord niet voorkomt op de bingokaart, dan moet de methode een AssertionError opwerpen met de boodschap *woord* staat niet op de kaart. Indien het gegeven woord reeds geschrapt is, dan moet de methode een AssertionError opwerpen met de boodschap *woord* is reeds geschrapt. In beide boodschappen moet het gegeven woord ingevuld worden op de plaats van *woord*.
- Een methode schrapWoorden die kan gebruikt worden om op de bingokaart een reeks woorden te schrappen. De te schrappen woorden moeten als een lijst of een tuple aan de methode doorgegeven worden. De methode moet een lijst teruggeven, waarin voor elk van de gegeven woorden staat waar dat woord gevonden werd op de bingokaart. Hetzelfde tuple als datgene dat teruggegeven wordt door de methode schrapWoord wordt gebruikt om de positie aan te geven waarop een woord gevonden werd.
- Een methode gewonnen die kan gebruikt worden om na te gaan of een speler gewonnen is op basis van de geschrapte woorden op de bingokaart. De methode moet een Booleaanse waarde teruggeven die aangeeft of er op de bingokaart al dan niet een rij of een kolom voorkomt waarin alle woorden geschrapt zijn.

Voorbeeld

```
>>> bingo = BuzzBingo(4, [
...     'cell', 'bacteria', 'PCR', 'virus',
...     'allele', 'chromosome', 'DNA', 'meiosis',
...     'protein', 'phenotype', 'gene', 'mutation',
...     'genome', 'recessive', 'RNA', 'mitosis'
... ])
>>> bingo
----
----
----
----
>>> bingo.gewonnen()
False

>>> bingo.schrapWoord('phenotype')
(2, 1)
>>> print(bingo)
----
----
-x--
```

```
----
>>> bingo.gewonnen()
False

>>> bingo.schrapWoord('dominant')
Traceback (most recent call last):
AssertionError: dominant staat niet op de kaart

>>> bingo.schrapWoord('phenotype')
Traceback (most recent call last):
AssertionError: phenotype is reeds geschrapt

>>> bingo.schrapWoorden(['PCR', 'chromosome', 'protein'])
[(0, 2), (1, 1), (2, 0)]
>>> bingo
--x-
-x--
xx--
----
>>> bingo.gewonnen()
False

>>> bingo.schrapWoorden(('recessive', 'bacteria', 'mitosis'))
[(3, 1), (0, 1), (3, 3)]
>>> print(bingo)
-xx-
-x--
xx--
-x-x
>>> bingo.gewonnen()
True

>>> bingo = BuzzBingo(2, ('cell', 'bacteria', 'PCR'))
Traceback (most recent call last):
AssertionError: ongeldige kaart
>>> bingo = BuzzBingo(2, ['cell', 'bacteria', 'PCR', 'virus', 'allele'])
Traceback (most recent call last):
AssertionError: ongeldige kaart
```