Bifid cipher

The *bifid cipher* is one of the classical cipher techniques that can also easily be executed by hand. The technique was invented around 1901 by amateur cryptographer Felix Delastelle. The cipher is a combination of substitution and fractionizing. A square $n \times n$ grid ($2 \leq n \leq 10$) is used. In this grid you will find all symbols that occur in the text that needs ciphering. In order to further explain this technique we will use the 9×9 grid below as an example. Pay attention to the fact that a space occurs as a symbol on the sixth row and the eighth column (rows and columns are numbered from zero).

	0	1	2	3	4	5	6	7	8
0	A	B	C	D	E	F	G	H	I
1	J	K	L	M	N	O	P	Q	R
2	S	T	U	V	W	X	Y	Z	0
3	1	2	3	4	5	6	7	8	9
4	a	b	c	d	e	f	g	h	i
5	j	k	l	m	n	o	p	q	r
6	s	t	u	v	w	x	y	z	
7	.	,	;	:	?	!	"	'	-
8	(	)	[	]	{	}	$	=	%

In order to cipher the original text `This is a dead parrot!`, every symbol of the text is converted to the corresponding row and column number where the symbol can be found. For example, we find uppercase letter `T` in the grid on row 2 and column 1. These row and column numbers are written vertically under the corresponding symbols from the original text.

```
original text: T h i s   i s   a   d e a d   p a r r o t !
               ---------------------------------------------
          row: 2 4 4 6 6 4 6 6 4 6 4 4 4 4 6 5 4 5 5 5 6 7
       column: 1 7 8 0 8 8 0 8 0 8 3 4 0 3 8 6 0 8 8 5 1 5
```

After that, the digits are written after each other: first the row numbers, followed by the column numbers.

```
2 4 4 6 6 4 6 6 ... 5 5 6 7 1 7 8 0 ... 8 6 0 8 8 5 1 5
```

At last, the digits are put together in groups of two, and every pair of digits is converted to the corresponding symbol in the square grid. The first digit of every pair represents the row number in the grid, the second represents the column number. The pair `2|4`, for example, corresponds with the uppercase letter `W`, that we can find in the grid on row 2 and column 4.

```
2|4 4|6 6|4 6|6 ... 5|5 6|7 1|7 8|0 ... 8|6 0|8 8|5 1|5
 W   g   w   y       o   z   Q   (       $   I   }   O
```

Here we see how the original text `This is a dead parrot!` was converted according to the bifid cipher in the coded text `WgwygeexfozQ(%II5D$I}O`. In order the decipher the text, we should apply the reverse process.

Assignment

Define a class Bifid that can be used to code an decode texts using the bifid cipher with a given square grid. This class must support the following methods:

- An initializing method __init__ that allows making a bifid cipher for a given square $n \times n$ grid. Two arguments must be given to this method: the value n and a string with length n^2 with the symbols of the grid, printed from left to right and from top to bottom. The initializing method should verify $2 \leq n \leq 10$ and that the string given has the correct length. Look at the example below to determine which action the method should take if the conditions aren't met.
- A method symbol that prints the symbol that can be found in the grid on a given row and a given column. The row and column number must be given to the method as separate arguments. Look at the example below to see which action the method should take if the position given in invalid.
- A method position that prints a tuple with the row number and the column number of the position in the grid on which a given symbol can be found. The symbol must be given as an argument to the method. Look at the example below to determine which action the method should take if the symbol does not consist of only one character, or if the symbol can't be found in the grid.
- A method code that prints the coded version of a given text as a result. The text must be coded according to the bifid cipher with the grid that was given when making the Bifid object. The original text must be given to the function as an argument.
- A method decode that prints the original version of a ciphered text as a result. The text must be decoded according to the bifid cipher with the grid that was given when making the Bifid object. The coded text must be given to the function as an argument.

Example

Remark: If you use the test examples below for a docstring, you should know that a single quotation mark that is used in the string that was given as second argument to the initializing method of the class Bifid, must get a double escape. In other words, the fragment \' in the string must be replaced by \\'.

```
>>> cipher = Bifid(9, 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdef' \
                   + 'ghijklmnopqrstuvwxyz .,;:?!"\'-()[]{}$=%')

>>> cipher.symbol(2, 1)
'T'
>>> cipher.symbol(7, 10)
Traceback (most recent call last):
AssertionError: invalid position in grid

>>> cipher.position('T')
(2, 1)
>>> cipher.position('WRONG')
Traceback (most recent call last):
AssertionError: symbol must consist of 1 character
>>> cipher.position('~')
Traceback (most recent call last):
AssertionError: unknown symbol: '~'
```

```
>>> cioher.code('This is a dead parrot!')
'WgwygeexfozQ(%II5D$I}O'
>>> cipher.decode('WgwygeexfozQ(%II5D$I}O')
'This is a dead parrot!'

>>> cipher = Bifid(20, '...')
Traceback (most recent call last):
AssertionError: 2 <= n <= 10 must apply
>>> cipher = Bifid(3, 'ABCDEFGHIJKLMNOPQRSTUVWXYZ')
Traceback (most recent call last):
AssertionError: number of symbols does not correspond to size grid
```

Bifidcodering is één van de klassieke coderingstechnieken die ook makkelijk met de hand kunnen uitgevoerd worden. De techniek werd rond 1901 uitgevonden door amateur-cryptograaf Felix Delastelle. De codering is een combinatie van substitutie met fractionering. Hierbij wordt gebruik gemaakt van een vierkant $n \times n$ rooster ($2 \leq n \leq 10$). In dit rooster worden alle symbolen geplaatst die in de te coderen tekst kunnen voorkomen. Om de techniek verder uit te leggen, zullen we bij wijze van voorbeeld werken met onderstaand 9×9 rooster. Let hierbij op het feit dat een spatie als symbool voorkomt in het rooster op rij zes en kolom acht (rijen en kolommen worden genummerd vanaf nul).

	0	1	2	3	4	5	6	7	8
0	A	B	C	D	E	F	G	H	I
1	J	K	L	M	N	O	P	Q	R
2	S	T	U	V	W	X	Y	Z	0
3	1	2	3	4	5	6	7	8	9
4	a	b	c	d	e	f	g	h	i
5	j	k	l	m	n	o	p	q	r
6	s	t	u	v	w	x	y	z	
7	.	,	;	:	?	!	"	'	-
8	(	)	[	]	{	}	$	=	%

Om de originele tekst `This is a dead parrot!` te coderen, wordt eerst elk symbool van de tekst omgezet naar het corresponderende rij- en kolomnummer waar het symbool terug te vinden is in het rooster. Zo vinden we bijvoorbeeld de hoofdletter `T` terug in het rooster op rij 2 en kolom 1. Deze rij- en kolomnummers worden verticaal onder de corresponderende symbolen van de originele tekst geschreven.

```
originele tekst: T h i s   i s   a   d e a d   p a r r o t !
                 ----------------------------------------------
            rij: 2 4 4 6 6 4 6 6 4 6 4 4 4 4 6 5 4 5 5 5 6 7
         kolom: 1 7 8 0 8 8 0 8 0 8 3 4 0 3 8 6 0 8 8 5 1 5
```

Daarna worden de cijfers achter elkaar uitgeschreven: eerst de rijnummers, gevolgd door de kolomnummers.

```
2 4 4 6 6 4 6 6 ... 5 5 6 7 1 7 8 0 ... 8 6 0 8 8 5 1 5
```

Tenslotte worden de cijfers in groepen van twee samengenomen, en wordt elk cijferpaar omgezet naar het corresponderende symbool in het vierkant rooster. Het eerste cijfer van elk paar stelt hierbij het rijnummer voor in het rooster, en het tweede cijfer het kolomnummer. Zo

correspondeert het paar 2|4 bijvoorbeeld met de hoofdletter W, die we in het rooster terugvinden op rij 2 en kolom 4.

```
2|4 4|6 6|4 6|6 ... 5|5 6|7 1|7 8|0 ... 8|6 0|8 8|5 1|5
 W   g   w   y       o   z   Q   (       $   I   }   O
```

Op die manier werd geïllustreerd hoe de originele tekst This is a dead parrot! volgens het bifidcijfer wordt omgezet in de gecodeerde tekst WgwygeexfozQ(%II5D$I}O. Voor decodering van een gecodeerd tekstbericht moet de omgekeerde bewerking uitgevoerd worden.

Opgave

Definieer een klasse Bifid waarmee teksten kunnen gecodeerd en gedecodeerd worden volgens de bifidcodering met een gegeven vierkant rooster. Deze klasse moet ondersteuning bieden aan de volgende methoden:

- Een initialisatiemethode __init__ die toelaat om een bifidcodering aan te maken voor een gegeven vierkant $n \times n$ rooster. Aan deze initialisatiemethode moeten twee argumenten doorgegeven worden: de waarde n en een string van lengte n^2 met de symbolen van het rooster, uitgeschreven van links naar rechts en van boven naar onder. De initialisatiemethode moet nagaan dat $2 \leq n \leq 10$ en dat de opgegeven string de correcte lengte heeft. Bekijk onderstaand voorbeeld om na te gaan welke actie de initialisatiemethode moet nemen indien niet aan deze voorwaarden voldaan is.
- Een methode symbool die het symbool teruggeeft dat in het rooster kan gevonden worden op een gegeven rij en een gegeven kolom. De gegeven rij- en kolomnummers moeten als afzonderlijke argumenten aan de methode doorgeven worden. Bekijk onderstaand voorbeeld om na te gaan welke actie de methode moet nemen indien een ongeldige positie in het rooster wordt opgegeven.
- Een methode positie die een tuple teruggeeft met het rijnummer en het kolomnummer van de positie in het rooster waarop een gegeven symbool kan teruggevonden worden. Het gegeven symbool moet als argument aan de methode doorgegeven worden. Bekijk onderstaand voorbeeld om na te gaan welke actie de methode moet nemen indien het gegeven symbool niet uit één enkel karakter bestaat, of indien het niet in het rooster teruggevonden wordt.
- Een methode codeer die de gecodeerde versie van een gegeven tekst als resultaat teruggeeft. Deze codering moet gebeuren volgens de bifidcodering met het rooster dat werd opgegeven bij het aanmaken van het Bifid object. De originele tekst moet als argument aan de functie doorgegeven worden.
- Een methode decodeer die de originele versie van een gegeven gecodeerde tekst als resultaat teruggeeft. Deze decodering moet gebeuren volgens de bifidcodering met het rooster dat werd opgegeven bij het aanmaken van het Bifid object. De gecodeerde tekst moet als argument aan de functie doorgegeven worden.

Voorbeeld

Opmerking: Indien je onderstaande testvoorbeelden opneemt in een docstring, let er dan op dat het enkele aanhalingsteken dat gebruikt wordt in de string die als tweede argument wordt doorgegeven aan de initialisatiemethode van de klasse Bifid, een dubbele escape moet meekrijgen. Met andere woorden, het fragment \' in de string moet vervangen worden door \\'.

```
>>> cijfer = Bifid(9, 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdef' \
                + 'ghijklmnopqrstuvwxyz .,;:?!"\'-()[]{}$=%')

>>> cijfer.symbool(2, 1)
'T'
>>> cijfer.symbool(7, 10)
Traceback (most recent call last):
AssertionError: ongeldige positie in rooster

>>> cijfer.positie('T')
(2, 1)
>>> cijfer.positie('FOUT')
Traceback (most recent call last):
AssertionError: symbool moet uit 1 karakter bestaan
>>> cijfer.positie('~')
Traceback (most recent call last):
AssertionError: onbekend symbool: '~'

>>> cijfer.codeer('This is a dead parrot!')
'WgwygeexfozQ(%II5D$I}O'
>>> cijfer.decodeer('WgwygeexfozQ(%II5D$I}O')
'This is a dead parrot!'

>>> cijfer = Bifid(20, '...')
Traceback (most recent call last):
AssertionError: er moet gelden dat 2 <= n <= 10
>>> cijfer = Bifid(3, 'ABCDEFGHIJKLMNOPQRSTUVWXYZ')
Traceback (most recent call last):
AssertionError: aantal symbolen komt niet overeen met grootte van het rooster
```