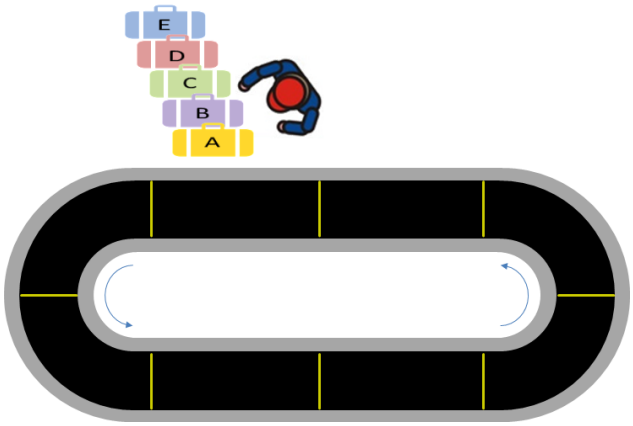
Baggage carousel

Upon arrival at an airport, passengers can pick up their luggage at a moving baggage carousel. The carousel rotates counterclockwise and is subdivided into compartments that each fit a single suitcase. First, the suitcases are transported from the aircraft to the baggage carousel by means of an automatic transport system. Afterwards, a porter drops the suitcases one by one on the carousel, always placing the next suitcase in the **third empty compartment** that passes by.

For example, suppose the porter has to drop five suitcases on a baggage carousel having eight compartments. For ease of reference, we have labeled the suitcases with the uppercase letters A, B, C, D and E, in the order the porter has to drop them on the carousel.



After the porter has dropped the last suitcase (E) on the baggage carousel, we get the following situation.

However, the carousel keeps spinning after all suitcases have been dropped, so that a few moments later the situation may look like this.

Assignment

In this assignment we represent a baggage carousel as a list. The compartments of the carousel correspond to the elements of the list. Each element that corresponds to a compartment containing a suitcase is assigned a single character string (the label of the suitcase). We assume that all suitcases on the carousel are labeled with a different character. Each element that corresponds to an empty compartment is assigned the value None.

The order of the elements in the list corresponds to the order of the compartments that pass by the position of the porter, until all compartments have passed him. For example, the carousel that corresponds to the middle figure in the introduction is represented by the list ['E', 'A', None, 'D', 'B', None, None, 'C']. That same carousel, but read from the situation in the bottom figure in the introduction, is represented by the list ['B', None, None, 'C', 'E', 'A', None, 'D']. These are two different lists that represent the same baggage carousel. You are asked to:

- Write a function baggageCarousel that takes two arguments: an integer c that indicates the number of compartments of a baggage carousel, and a string whose characters correspond to the labels of a sequence of suitcases that must be dropped onto an empty carousel in the order in which the characters occur in the string. All characters in the string are different, and the number of characters is less than or equal to c. The function must return a list that represents the baggage carousel with c compartments after all suitcases were dropped on the carousel by the porter. Keep in mind that the porter drops each suitcase in the **third empty compartment** that passes by. Make sure that the first element of the returned list corresponds to the compartment in which the last suitcase was dropped.
- Write a function rotated that takes two lists, each representing a baggage carousel. The function must return a Boolean value that indicates whether or not the given lists represent the same carousel. Two lists represent the same baggage carousel if they have an equal number of compartments, that are filled with the same bags in the same order. The only difference that may occur is that the compartments are read from a different starting position.

Example

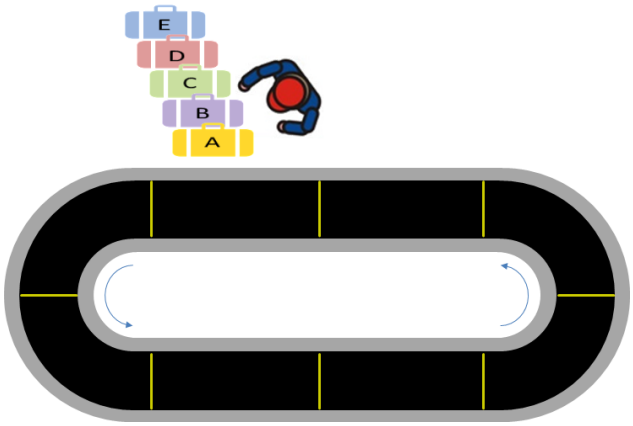
```
>>> carousel(8, 'ABCDE')
['E', 'A', None, 'D', 'B', None, None, 'C']
>>> carousel(5, 'ABCDE')
['E', 'C', 'B', 'D', 'A']

>>> rotated(['E', 'A', None, 'D', 'B', None, None, 'C'], ['B', None, None, 'C', 'E', 'A', None, 'D'])
True
>>> rotated(['E', 'A', None, 'D', 'B', None, None, 'C'], ['B', None, None, 'A', 'E', 'C', None, 'D'])
False
>>> rotated(['E', 'A', None, 'D', 'B', None, None, 'C'], ['C', 'E', 'A', None, 'D', 'B'])
False
```

Bij aankomst op een luchthaven kunnen de passagiers hun koffers afhalen op een bewegende bagageband. De bagageband roteert in tegenwijzerzin en is onderverdeeld in afgescheiden compartimenten waarin telkens één koffer past. Eerst worden de koffers door een automatisch transportsysteem vanaf het vliegtuig tot bij de bagageband gebracht. Daarna plaatst een kruier de koffers één voor één op de bagageband, door telkens de volgende koffer te plaatsen in het **derde lege compartiment** dat bij hem voorbijkomt.

Stel bijvoorbeeld dat de kruier op die manier vijf koffers op een bagageband met acht

compartimenten moet plaatsen. Voor de eenvoud hebben we de koffers gelabeld met de hoofdletters A, B, C, D en E, in de volgorde waarin ze door de kruier op de band geplaatst worden.



Net nadat de kruier de laatste koffer (E) op de bagageband geplaatst heeft, krijgen we dus de volgende situatie.

De band blijft daarna verder ronddraaien, zodat de situatie er een tijdje later bijvoorbeeld als volgt kan uitzien.

Opgave

In deze opgave stellen we een bagageband voor als een lijst. De compartimenten van de bagageband corresponderen met de elementen van de lijst. Aan een element dat correspondeert met een compartiment waarin een koffer ligt, wordt een string toegekend die bestaat uit één enkel karakter (het label van de koffer). We gaan ervan uit dat alle koffers op de bagageband gelabeld zijn met een verschillend karakter. Aan een element dat correspondeert met een leeg compartiment wordt de waarde None toegekend.

De volgorde van de elementen in de lijst komt overeen met de volgorde van de compartimenten die voorbijkomen op de positie waar de kruier staat, en dit tot hij alle compartimenten heeft zien

passeren. Zo wordt de bagageband die correspondeert met de middelste figuur van de inleiding voorgesteld als de lijst `['E', 'A', None, 'D', 'B', None, None, 'C']`. Dezelfde bagageband, maar dan uitgelezen in de situatie uit de onderste figuur van de inleiding, wordt voorgesteld door de lijst `['B', None, None, 'C', 'E', 'A', None, 'D']`. Dit zijn dus twee verschillende lijsten die in principe dezelfde bagageband voorstellen. Gevraagd wordt:

- Schrijf een functie `bagageband` waaraan twee argumenten moeten doorgegeven worden: een natuurlijk getal c dat aangeeft hoeveel compartimenten een bagageband heeft, en een string waarvan de karakters corresponderen met de labels van een aantal koffers die op de lege band moeten geplaatst worden in de volgorde waarin de karakters voorkomen in de string. Alle karakters in de string zijn verschillend, en het aantal karakters is kleiner of gelijk aan c. De functie moet een lijst teruggeven die de bagageband met c compartimenten voorstelt nadat de koffers één voor één door de kruier op de bagageband geplaatst werden. Hou hierbij rekening met het feit dat de kruier telkens de volgende koffer plaatst in het **derde lege compartiment** dat bij hem voorbijkomt. Zorg ervoor dat het eerste element van deze lijst correspondeert met het compartiment waarin de laatste koffer geplaatst werd.
- Schrijf een functie `doorgedraaid` waaraan twee lijsten moeten doorgegeven worden, die elk een bagageband voorstellen. De functie moet een Booleaanse waarde teruggeven die aangeeft of de twee gegeven lijsten al dan niet dezelfde bagageband voorstellen. Twee lijsten stellen dezelfde bagageband voor als ze evenveel compartimenten hebben, die gevuld zijn met dezelfde koffers en dit in dezelfde volgorde. Het enige verschil dat mag optreden tussen de lijsten is dat men de compartimenten op een verschillende plaats is beginnen uitlezen.

Voorbeeld

```
>>> bagageband(8, 'ABCDE')
['E', 'A', None, 'D', 'B', None, None, 'C']
>>> bagageband(5, 'ABCDE')
['E', 'C', 'B', 'D', 'A']

>>> doorgedraaid(['E', 'A', None, 'D', 'B', None, None, 'C'], ['B', None, None, 'C', 'E', 'A', None, 'D'])
True
>>> doorgedraaid(['E', 'A', None, 'D', 'B', None, None, 'C'], ['B', None, None, 'A', 'E', 'C', None, 'D'])
False
>>> doorgedraaid(['E', 'A', None, 'D', 'B', None, None, 'C'], ['C', 'E', 'A', None, 'D', 'B'])
False
```