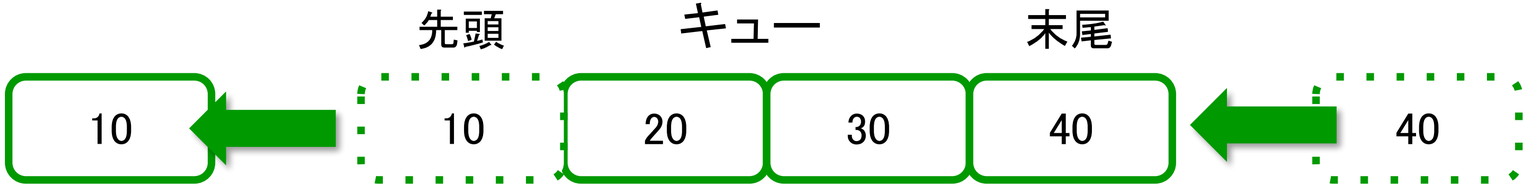
2. キューの復習

キューとはなにか？（復習）

- リストの中で、要素の追加が一方の端（末尾）からしか、削除はもう一方の端（先頭）からしかできないものを「キュー」と呼びます
  - 要素が「列に並ぶ」イメージ
  - 最初に取り出せるデータは、最初に追加されたものなので、「FIFO(First In First Out)」とも呼ばれる
- デキュー（Dequeue）とエンキュー（Enqueue）という2つの操作を提供します



Dequeue
先頭から要素を削除する操作

Enqueue
末尾に要素を追加する操作

身近なキュー（復習）

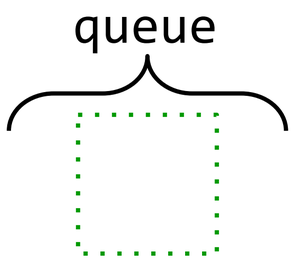
- 文字通り、人の「行列」
  - 来た人は最後尾から並ばなければならない
  - 先頭の人から、順番に出ていく
  - 「前からN番目に割り込む」ことは普通は許されない
- キュー的な考え方でいうと、、、
  - 最後に来た人が、一番後ろにいる
  - 最初に来た人は、一番前にいる

キューの簡単な使い方

- Pythonではdeque（デック）というデータ構造があり、データの両端から追加・削除できるようになっています。
  - 今回は、dequeを「キューとして」使うため、先頭から取り出す、最後尾から追加する以外の操作を行いません。

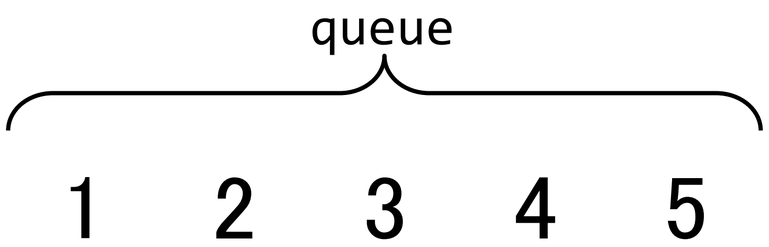
- 例１：空のdequeを作成

```
from collections import deque
queue = deque()
```



- 例２：リストからdequeを作成

```
from collections import deque
queue = deque([1, 2, 3, 4, 5])
```

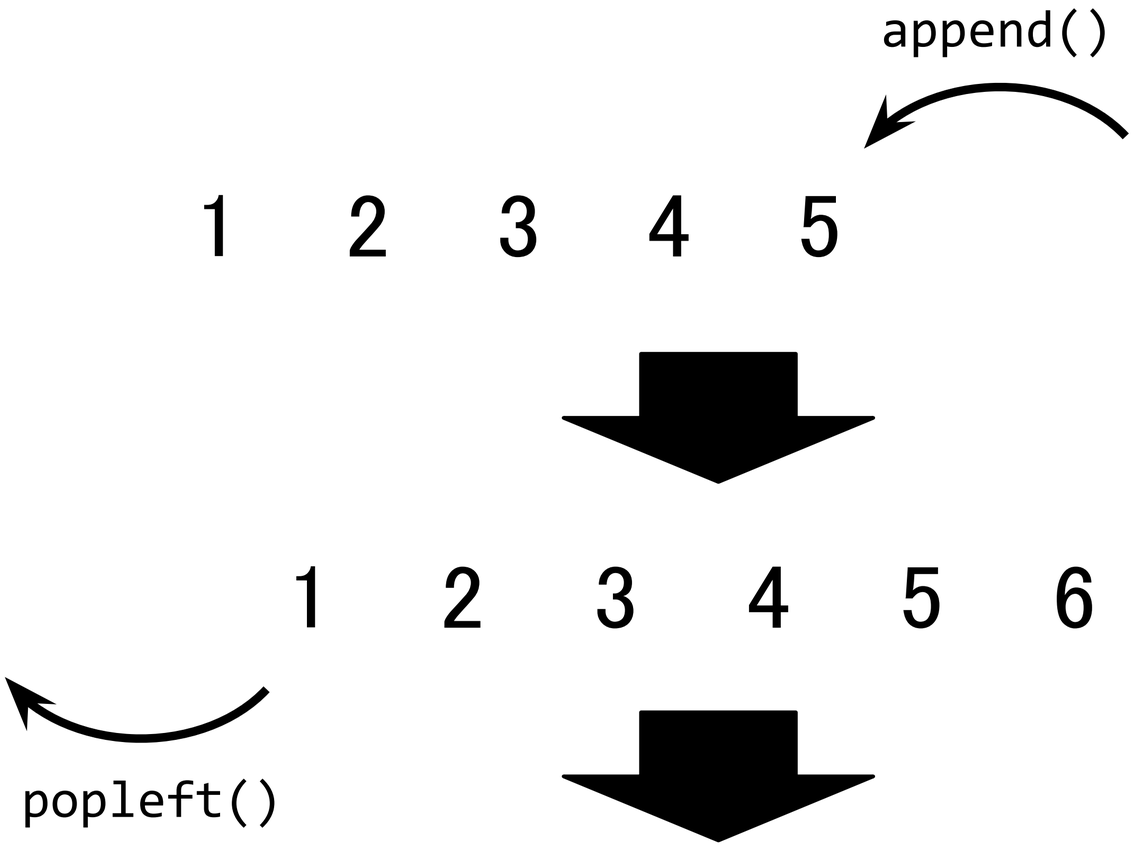


キューの簡単な使い方

- 例3：最後尾に追加

```
from collections import deque
queue = deque([1, 2, 3, 4, 5])

queue.append(6)
```



- 例4：先頭から取り出す

```
（続き）

x = queue.popleft()
```

queue: 2 3 4 5 6

x 1

キューの簡単な使い方

- dequeのデータの個数を調べるには、len() を使うことができます（リストと同じ）

- 例5：データの個数を取得する

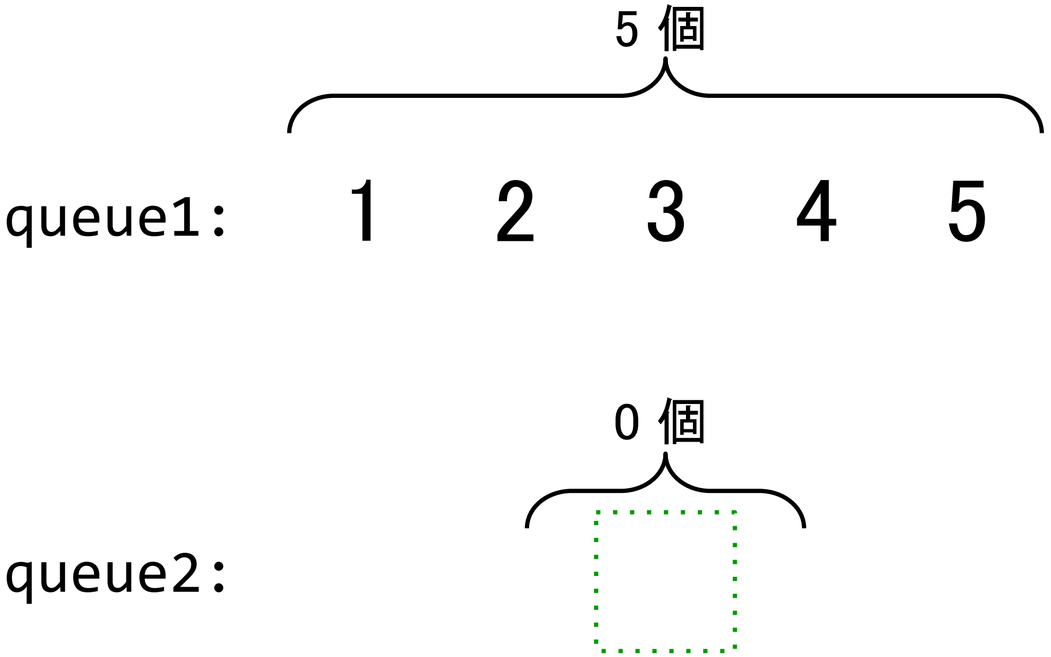
```
from collections import deque

queue1 = deque([1, 2, 3, 4, 5])

print( len(queue1) )   # 5 が返ってくる

queue2 = deque()

print( len(queue2) )   # 0 が返ってくる
```



popleft()に関する注意

- データ数が 0 のときに popleft() を実行すると・・・

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_3792/2755806108.py in <module>
      3 queue = deque()
      4
----> 5 queue.popleft()

IndexError: pop from an empty deque
```

- 実行する前に len() を使ってエラーを回避する必要があります。

```
if len(queue) > 0:
    queue.popleft()
```

空でない（要素数が0より大きい）場合にのみ、popleft() を実行する。

```
if queue:
    queue.popleft()
```

こちらでもOK