- **Tuple:** una *tupla* è una collezione simile a una lista, utilizzata per creare sequenze complesse. Uno dei vantaggi delle tuple è che è possibile nidificarne il contenuto, caratteristica mediante la quale è possibile creare strutture che contengono, per esempio, record sui dipendenti o coppie di coordinate x-y.
- **Dizionari:** con una collezione dizionario è possibile creare coppie chiave/valore, un po' come accade con i dizionari reali, nei quali a una parola è associata una definizione. Un dizionario consente tempi di ricerca incredibilmente veloci e rende molto più facile l'ordinamento dei dati.
- **Stack (o pila):** la maggior parte dei linguaggi di programmazione supporta direttamente gli stack, mentre Python no, sebbene esista un sistema alternativo per farlo. Uno *stack* è una sequenza LIFO (last in/first out, il primo elemento a uscire è l'ultimo che è entrato). È un po' come una pila di frittelle: a mano a mano che se ne aggiunge una, quella in cima è quella che verrà presa per prima. Lo stack è una collezione molto importante, che in Python può essere simulata mediante una lista.
- **Code:** una *coda* è una collezione FIFO (first in/first out, il primo elemento a uscire è il primo che è stato inserito). Serve a tenere traccia degli elementi che necessitano di un qualche tipo di elaborazione. Una coda è un po' come quando si fa la fila in banca: entrate nella fila, aspettate il vostro turno e alla fine venite chiamati per conferire con un operatore di sportello.
- **Deque:** una *deque* (double-ended queue, coda con due estremità) è una struttura simile a una coda, che consente di aggiungere o rimuovere elementi da entrambi i capi, ma non dal centro. La deque può essere utilizzata come una coda, come uno stack o come un qualsiasi altro tipo di collezione a cui si possano aggiungere e da cui si possano rimuovere elementi in modo ordinato, a differenza di quanto accade con le liste, le tuple e i dizionari, che consentono accesso e gestione randomizzati.

Di tutte le sequenze, le liste sono le più facili da capire e le più direttamente correlate a oggetti del mondo reale. Lavorare con le liste vi aiuterà a imparare a lavorare con altri tipi di sequenze, le quali garantiscono funzionalità maggiori e una migliore flessibilità. I dati vengono memorizzati nelle liste più o meno come si potrebbe fare su un foglio di carta: un elemento viene dopo l'altro. Una lista ha un inizio, un centro e una fine. Come si può vedere qui di seguito, gli elementi sono numerati. (Sebbene nella vita reale sia raro numerare gli elementi di una lista, Python esegue questa operazione automaticamente.) Per vedere come si lavora con le liste, avviate un Jupyter Notebook e digitate questo codice:

```
ListA = [0, 1, 2, 3]
ListB = [4, 5, 6, 7]
ListA.extend(ListB)
ListA
```

Quando digitate l'ultima riga di codice, l'output visualizzato è `[0, 1, 2, 3, 4, 5, 6, 7]`. La funzione `extend()` aggiunge a `ListA` gli elementi di `ListB`. Oltre a estendere le liste, è anche possibile aggiungervi degli elementi utilizzando la funzione `append()`. Digitate `ListA.append(-5)` e premete Invio: quando poi digiterete `ListA` e premerete Invio, potrete notare che Python ha aggiunto −5 al termine della lista. A volte è necessario rimuovere degli