```
'Output': Number is 8
     Number is 7
     Number is 6
     Number is 5
     Number is 4
     Number is 3
     Number is 2
     Number is 1
```

The for Loop

The for loop repeats the statements within its code block until a terminating condition is reached. It is different from the while loop in that it knows exactly how many times the iteration should occur. The for loop is controlled by an iterable expression (i.e., expressions in which elements can be accessed sequentially). The for statement is visualized by the flowchart in Figure 9-4.

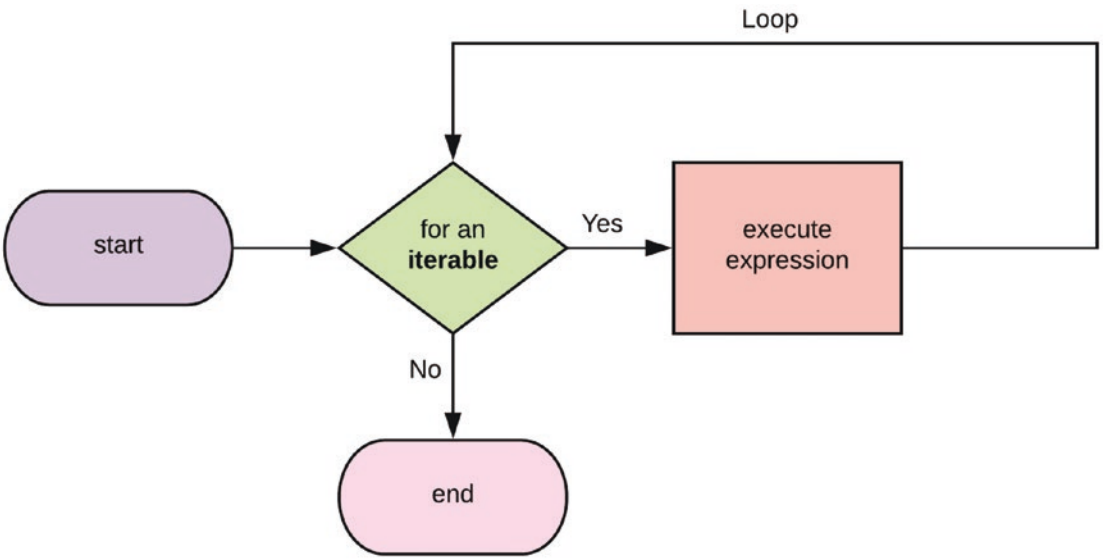

Figure 9-4. *Flowchart of the for loop*

The syntax for the for loop is as follows:

```
for item in iterable:
    statement
```

Note that in the for loop syntax is not the same as the membership logical operator earlier discussed.

Here is a program example:

```
a = [2, 4, 6, 8, 10]
for elem in a:
    print(elem**2)
```

```
'Output': 4
    16
    36
    64
    100
```

To loop for a specific number of time, use the range() function.

```
for idx in range(5):
    print('The index is', idx)
```

```
'Output': The index is 0
     The index is 1
     The index is 2
     The index is 3
     The index is 4
```

List Comprehensions

Using list comprehension, we can succinctly rewrite a for loop that iteratively builds a new list using an elegant syntax. Assuming we want to build a new list using a for loop, we will write it as

```
new_list = []
for item in iterable:
    new_list.append(expression)
```