

Else Statement and Loops

Else Statement and Loops

You can use an else statement after a `for` or `while` loop. The code in this `else` statement is executed if the loop completed without being terminated.

```
[3]: for i in range(3):  
      print(i)  
      else:  
          print('Loop completed')
```

```
0  
1  
2  
Loop completed
```

The only time the `else` part will not be executed is if you `break` out of a loop:

```
[4]: for i in range(5):  
      print(i)  
  
      if i == 3:  
          break  
      else:  
          print('Loop completed')
```

```
0  
1  
2  
3
```

Worked Example

A common use for this structure is if you're searching for an object. Consider this example where we are trying to find a 'fish' in a list:

```
[5]: animals = ['zebra', 'cow', 'crow', 'eel']  
  
for animal in animals:  
    if animal == 'fish':
```

```
        print('We caught a fish!')
        break
else:
    print('We did not catch a fish.')
```

We did not catch a fish.

```
[6]: animals = ['human', 'bear', 'fish', 'squid', 'crab']

for animal in animals:
    if animal == 'fish':
        print('We caught a fish!')
        break
else:
    print('We did not catch a fish.')
```

We caught a fish!

Of course, finding a particular object in a list is quicker and simpler using:

```
[7]: animals = animals = ['human', 'bear', 'fish', 'squid', 'crab']

if 'fish' in animals:
    print('We caught a fish!')
else:
    print('We did not catch a fish.')
```

We caught a fish!

but for more complex procedures this may not be an option.