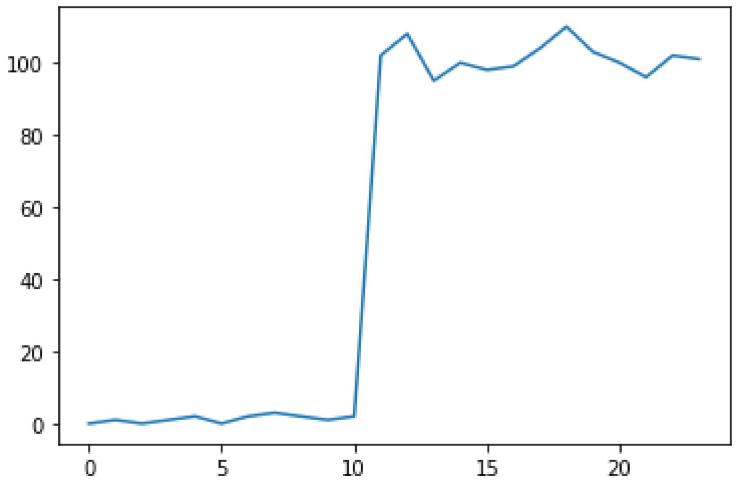
In [ ]:
```
from __future__ import print_function
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import sympy as sym
sym.init_printing(use_latex = "mathjax")
```

In [ ]:
```
ys = np.array([0, 1, 0, 1, 2, 0, 2, 3, 2, 1, 2, 102, 108,  95, 100,  98,  99, 104, 110

fig,ax = plt.subplots()
ax.plot([i for i in range(len(ys))], ys);
```



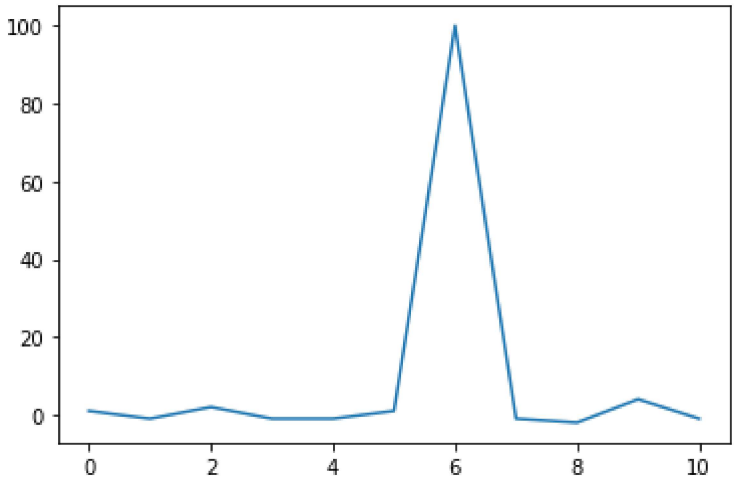
In [ ]:
```
def make_windows(sequence, windowsize):
    positions = len(sequence) - windowsize + 1
    windows = []
    for i in range(positions):
        windows.append(sequence[i:i+windowsize])
    return windows

def print_padded_seq(seq):
    print("[", ",".join(["{:4d}".format(i) for i in seq]), ']')

def print_sliding_windows(seq, windowsize=3):
    windows = make_windows(seq, windowsize)
    for window in windows:
        print(",".join(["{:4d}".format(i) for i in window]))
```

In [ ]:
```
series = [0, 1, 0, 2, 1, 0, 1, 101, 100, 98, 102, 101]
windowsize = 2

print_padded_seq(series)

print_sliding_windows(series, windowsize=windowsize)
```

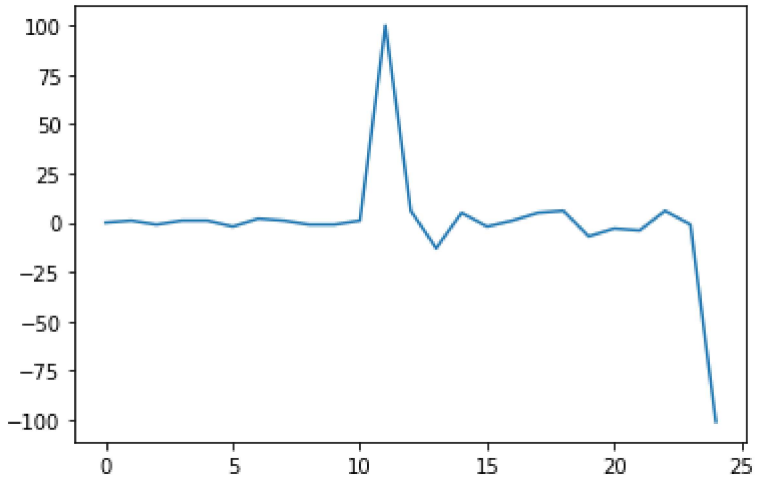
```
[   0,   1,   0,   2,   1,   0,   1, 101, 100,  98, 102, 101 ]
  0,   1
  1,   0
  0,   2
  2,   1
  1,   0
  0,   1
  1, 101
101, 100
100,  98
 98, 102
102, 101
```

In [ ]:
```
convolutions = []
kernel = np.array([-1,1])
for w in make_windows(series, windowsize=2):
    w = np.array(w)
    convolved = np.dot(w,kernel)
    convolutions.append(convolved)
print(convolutions)
plt.plot(convolutions);
```

```
[1, -1, 2, -1, -1, 1, 100, -1, -2, 4, -1]
```



In [ ]:
```
convolved = np.convolve([1, -1], ys)

fig,ax = plt.subplots()
ax.plot([i for i, _ in enumerate(convolved)], convolved);
```



Question: Why does the graph move down at the end?

Numpy's convolve function will append a 0 to the end of the array in order to complete the convolution on the array. Since the last value in the array is around 100, appending a 0 and performing the operation will result in a very large negative slope. When graphed, this results in a large downward spike.