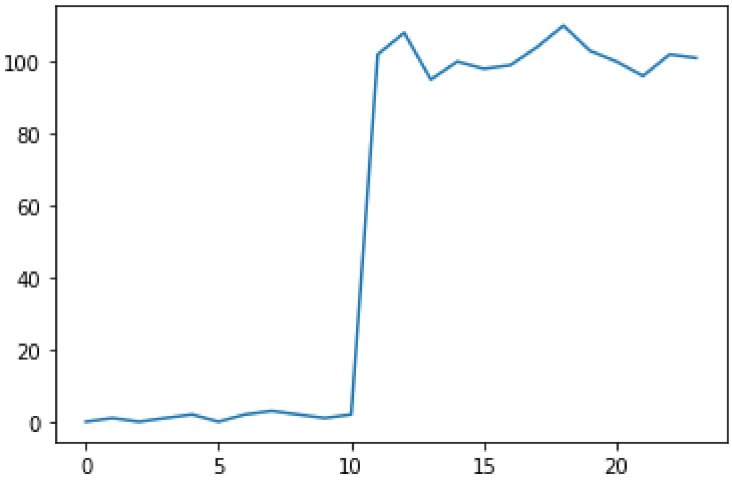
Now, let's generate a fake one-dimensional signal:

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);
# check(1)
```



Next, let's look at small chunks of our fake signal:

In [ ]:
```
chunks = np.split(ys, len(ys)//2)
print(chunks)
# check(2)
```

```
[array([0, 1]), array([0, 1]), array([2, 0]), array([2, 3]), array([2, 1]), array([
2, 102]), array([108,  95]), array([100,  98]), array([ 99, 104]), array([110, 103]),
array([100,  96]), array([102, 101])]
```

Question: Which one of these chunks would you say is the most "interesting"?

The most "interesting" chunk is the array [2, 102].

Question If we always divide up the signal as we did above, will we always find something "interesting"?

Not necessarily; if the array increases slowly (i.e. small differences in every chunk), there wouldn't be a single chunk with a large difference.

Convolutions

Derivatives and convolutions are one technique to help us tackle the above problem.

First, you'll need to generate windows into the signal. Write a function that can generate windows with a user-supplied windowsize, and print them out.

An example signal with 3 window sizes is shown below. Your output does not need to replicate the formatting shown, but they should produce the same windows. E.g., given an input signal of

`[10,20,30]` and a `windowsize=2`, your function should return `[[10,20], [20,30]]`.

A windowsize of 1:

```
signal:
        0   1   0   2   1   0   1 101 100  98 102 101
 0:     0
 1: ____    1
 2: ________    0
 3: ____________    2
 4: ________________    1
 5: ____________________    0
 6: ________________________    1
 7: ____________________________  101
 8: ________________________________  100
 9: ____________________________________   98
10: ________________________________________  102
11: ____________________________________________  101

     ::::::::::::::::::::::::::::::::::::::::::::::::

i:    0 |  i + windowsize:     1  |  window:  [   0]
i:    1 |  i + windowsize:     2  |  window:  [   1]
i:    2 |  i + windowsize:     3  |  window:  [   0]
i:    3 |  i + windowsize:     4  |  window:  [   2]
i:    4 |  i + windowsize:     5  |  window:  [   1]
i:    5 |  i + windowsize:     6  |  window:  [   0]
i:    6 |  i + windowsize:     7  |  window:  [   1]
i:    7 |  i + windowsize:     8  |  window:  [ 101]
i:    8 |  i + windowsize:     9  |  window:  [ 100]
i:    9 |  i + windowsize:    10  |  window:  [  98]
i:   10 |  i + windowsize:    11  |  window:  [ 102]
i:   11 |  i + windowsize:    12  |  window:  [ 101]
```

A windowsize of 2:

```
signal:
        0   1   0   2   1   0   1 101 100  98 102 101
 0:     0   1
 1: ____    1   0
 2: ________    0   2
 3: ____________    2   1
 4: ________________    1   0
 5: ____________________    0   1
 6: ________________________    1 101
 7: ____________________________  101 100
 8: ________________________________  100  98
 9: ____________________________________   98 102
10: ________________________________________  102 101

     ::::::::::::::::::::::::::::::::::::::::::::::::
```

```
i:    0 |  i + windowsize:     2  |  window:  [   0,   1]
i:    1 |  i + windowsize:     3  |  window:  [   1,   0]
i:    2 |  i + windowsize:     4  |  window:  [   0,   2]
i:    3 |  i + windowsize:     5  |  window:  [   2,   1]
i:    4 |  i + windowsize:     6  |  window:  [   1,   0]
i:    5 |  i + windowsize:     7  |  window:  [   0,   1]
i:    6 |  i + windowsize:     8  |  window:  [   1, 101]
i:    7 |  i + windowsize:     9  |  window:  [ 101, 100]
i:    8 |  i + windowsize:    10  |  window:  [ 100,  98]
i:    9 |  i + windowsize:    11  |  window:  [  98, 102]
i:   10 |  i + windowsize:    12  |  window:  [ 102, 101]
```

A windowsize of 3

```
signal:
        0   1   0   2   1   0   1 101 100  98 102 101
 0:     0   1   0
 1: ____     1   0   2
 2: ________     0   2   1
 3: ____________     2   1   0
 4: ________________    1   0   1
 5: ____________________    0   1 101
 6: ________________________    1 101 100
 7: ____________________________  101 100  98
 8: ________________________________  100  98 102
 9: ____________________________________   98 102 101

     ::::::::::::::::::::::::::::::::::::::::::::::::::

i:    0 |  i + windowsize:     3  |  window:  [   0,   1,   0]
i:    1 |  i + windowsize:     4  |  window:  [   1,   0,   2]
i:    2 |  i + windowsize:     5  |  window:  [   0,   2,   1]
i:    3 |  i + windowsize:     6  |  window:  [   2,   1,   0]
i:    4 |  i + windowsize:     7  |  window:  [   1,   0,   1]
i:    5 |  i + windowsize:     8  |  window:  [   0,   1, 101]
i:    6 |  i + windowsize:     9  |  window:  [   1, 101, 100]
i:    7 |  i + windowsize:    10  |  window:  [ 101, 100,  98]
i:    8 |  i + windowsize:    11  |  window:  [ 100,  98, 102]
i:    9 |  i + windowsize:    12  |  window:  [  98, 102, 101]
```

In [ ]:
```python
def make_windows(sequence, windowsize):

    windowlist = []
    i = 0
    while i < len(sequence):
        window = []
        for j in range(windowsize):
            if i + j < len(sequence):
                window.append(sequence[i+j])
        i += windowsize
        windowlist.append(window)

    print(windowlist)
```

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

make_windows(sequence=series, windowsize=1)
make_windows(sequence=series, windowsize=2)
make_windows(sequence=series, windowsize=3)
```

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