PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

# Python
# Data structures

Robert Haase

April 2022

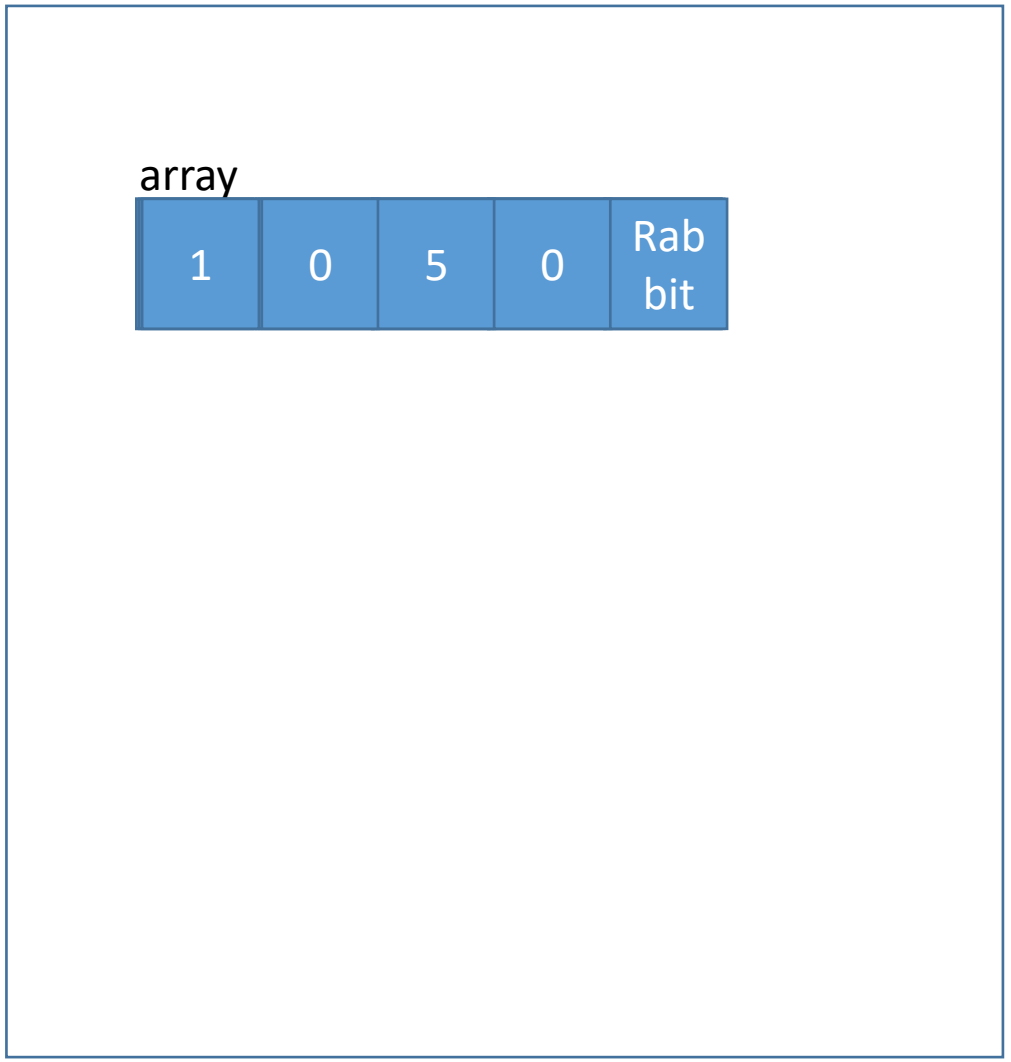
# Lists

- Lists are variables, where you can store multiple values

Give me a “0”, five times!

```
array = [0] * 5
```

Computer memory

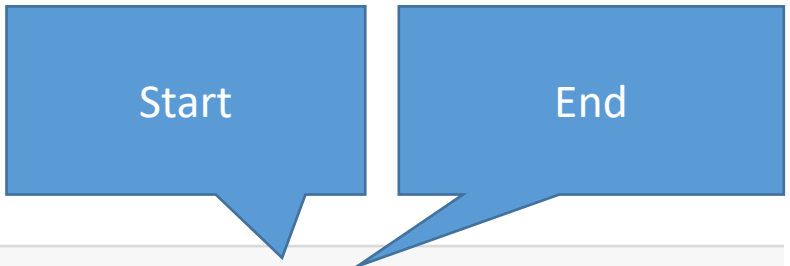


# Subsets

```
# Arrays
numbers = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
print(numbers)
```

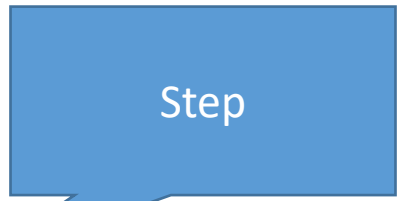
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

- Creating subsets of lists




```
subset = numbers[2:4]
print(subset)
```

```
[2, 3]
```




```
subset_with_gaps = arr[1:8:2]
print(subset_with_gaps)
```

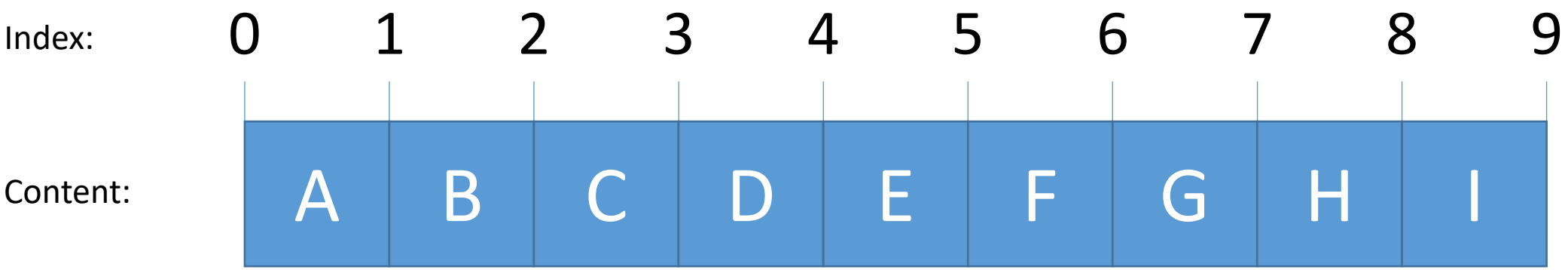
```
[1, 3, 5, 7]
```

**`data[start:stop:step]`**

# Indexing, cropping, subsets

- “Indexing” is addressing certain elements in lists. The first element is “0” away from the start.

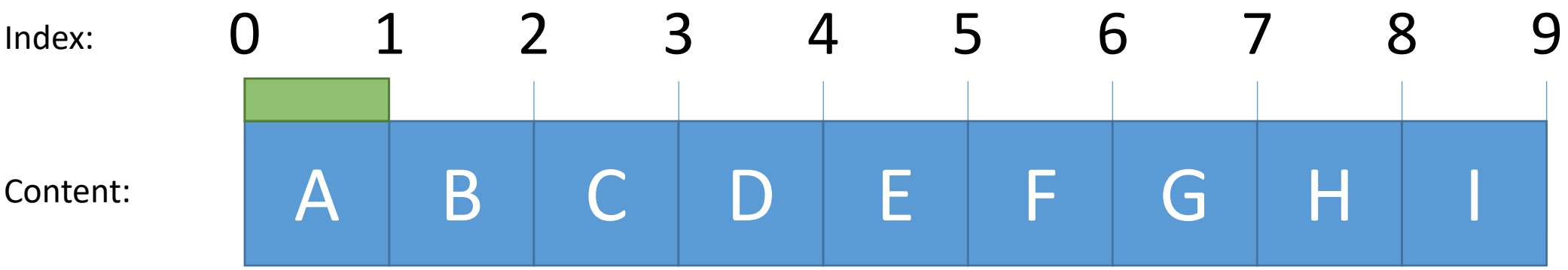
```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



# Indexing, cropping, subsets

- “Indexing” is addressing certain elements in lists. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



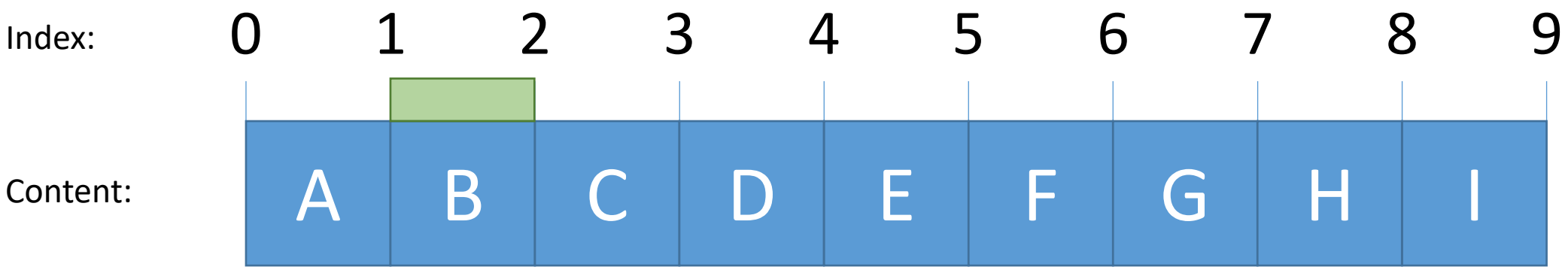

```
data[0]
```

```
'A'
```

# Indexing, cropping, subsets

- “Indexing” is addressing certain elements in lists. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[0]
```

```
'A'
```

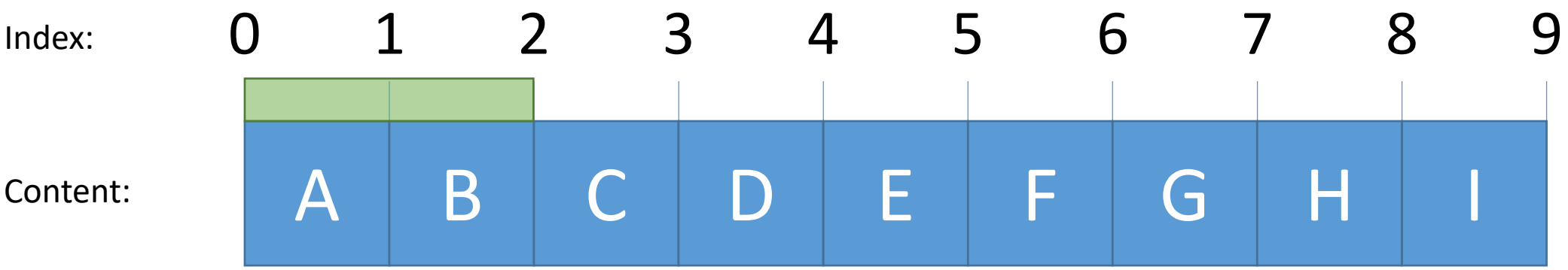
```
data[1]
```

```
'B'
```

# Indexing, cropping, subsets

- “Indexing” is addressing certain elements in lists. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[0]
```

```
'A'
```

```
data[1]
```

```
'B'
```

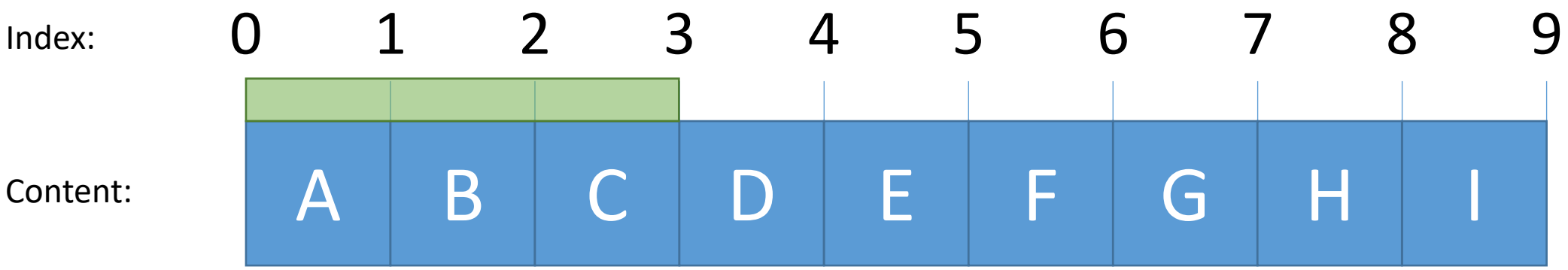
```
data[0:2]
```

```
['A', 'B']
```

# Indexing, cropping, subsets

- “Indexing” is addressing certain elements in lists. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[0]
```
'A'

```
data[1]
```
'B'

```
data[0:2]
```
['A', 'B']

```
data[0:3]
```
['A', 'B', 'C']

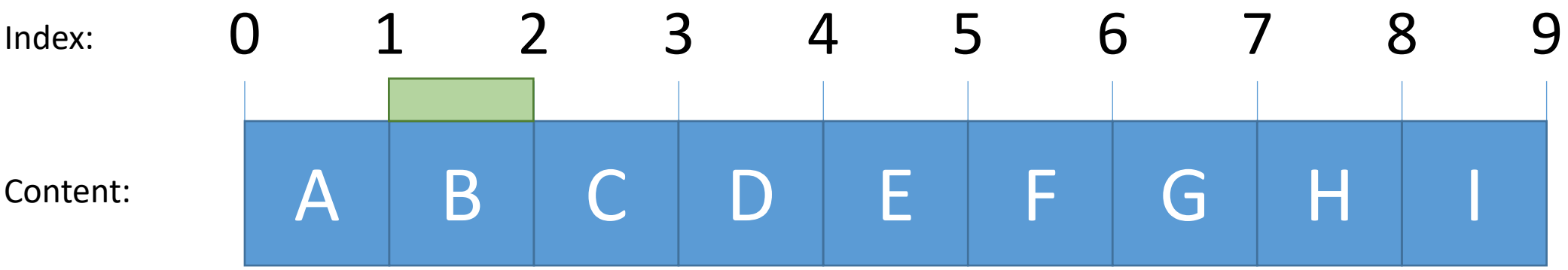
```
data[1:2]
```
['B']

```
len(data)
```
9

# Indexing, cropping, subsets

- “Indexing” is addressing certain elements in lists. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[0]
```
'A'

```
data[1]
```
'B'

```
data[0:2]
```
['A', 'B']

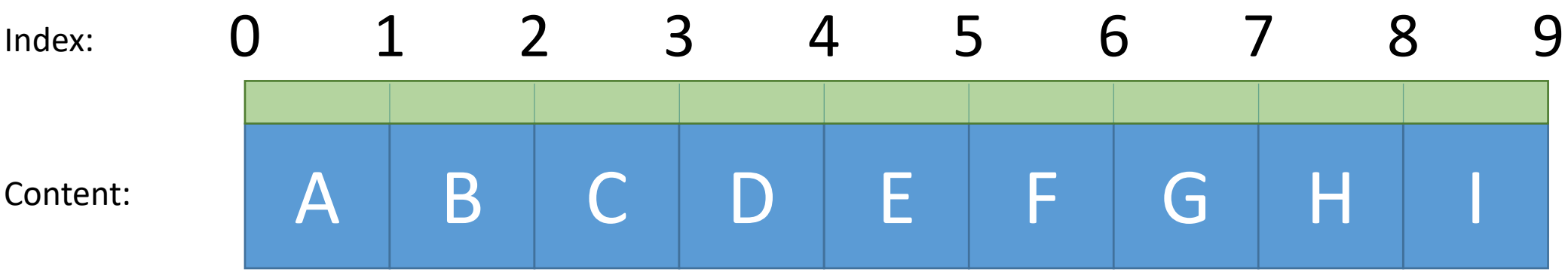
```
data[0:3]
```
['A', 'B', 'C']

```
data[1:2]
```
['B']

# Indexing, cropping, subsets

- “Indexing” is addressing certain elements in lists. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[0]
```
'A'

```
data[1]
```
'B'

```
data[0:2]
```
['A', 'B']

```
data[0:3]
```
['A', 'B', 'C']

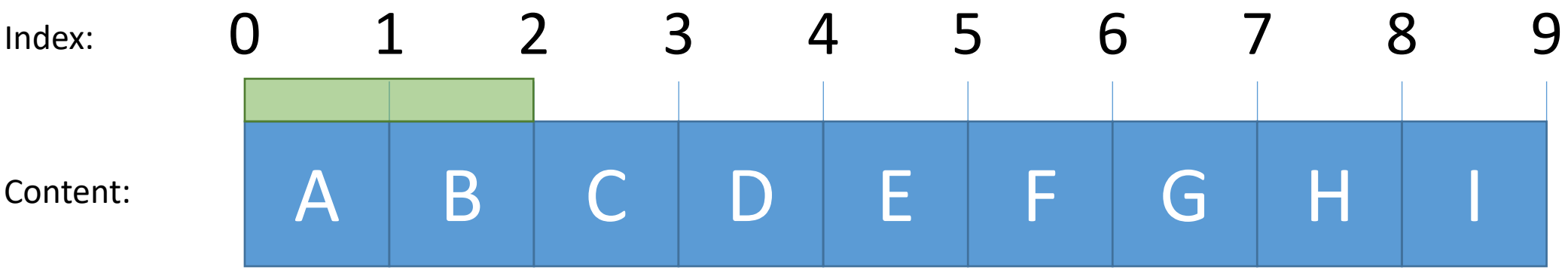
```
data[1:2]
```
['B']

```
len(data)
```
9

# Indexing, cropping, subsets

- You  can leave start and end out when specifying index ranges

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



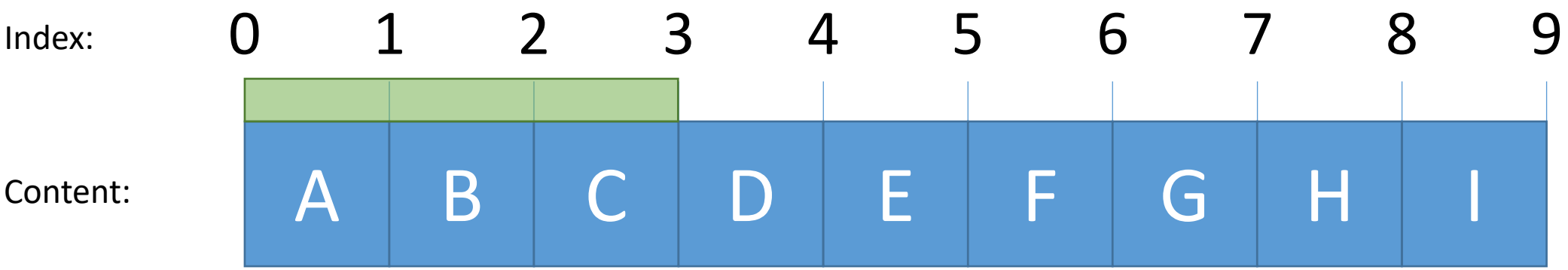

```
data[:2]
```

```
['A', 'B']
```

# Indexing, cropping, subsets

- You  can leave start and end out when specifying index ranges

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[:2]
```

```
['A', 'B']
```

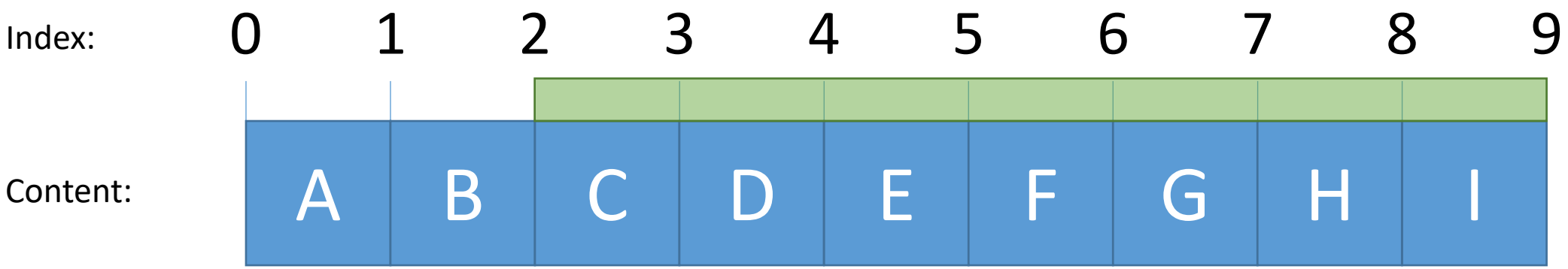
```
data[:3]
```

```
['A', 'B', 'C']
```

# Indexing, cropping, subsets

- You  can leave start and end out when specifying index ranges

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[:2]
```

```
['A', 'B']
```

```
data[:3]
```

```
['A', 'B', 'C']
```

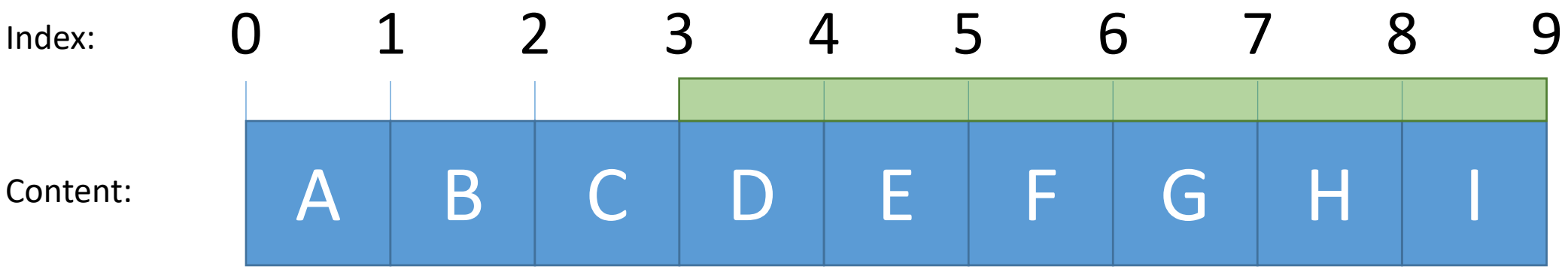
```
data[2:]
```

```
['C', 'D', 'E', 'F', 'G', 'H', 'I']
```

# Indexing, cropping, subsets

- You  can leave start and end out when specifying index ranges

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[:2]
```

```
['A', 'B']
```

```
data[:3]
```

```
['A', 'B', 'C']
```

```
data[2:]
```

```
['C', 'D', 'E', 'F', 'G', 'H', 'I']
```

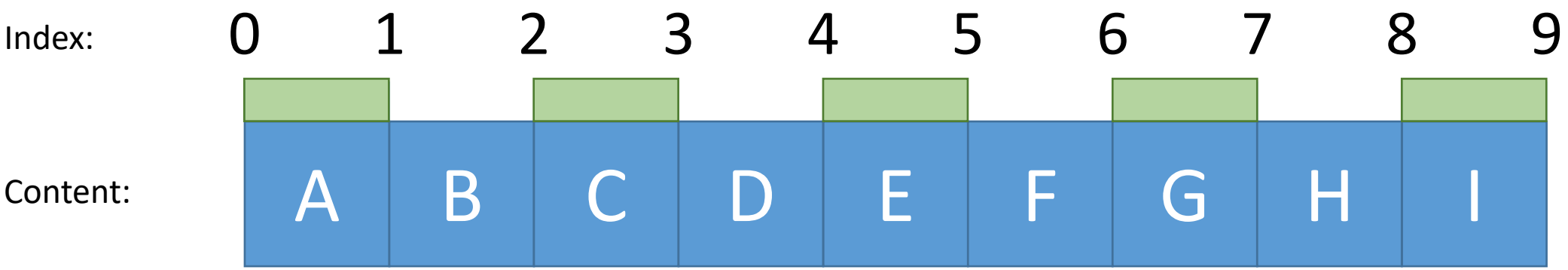
```
data[3:]
```

```
['D', 'E', 'F', 'G', 'H', 'I']
```

# Indexing, cropping, subsets

- The step-size allows skipping elements

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



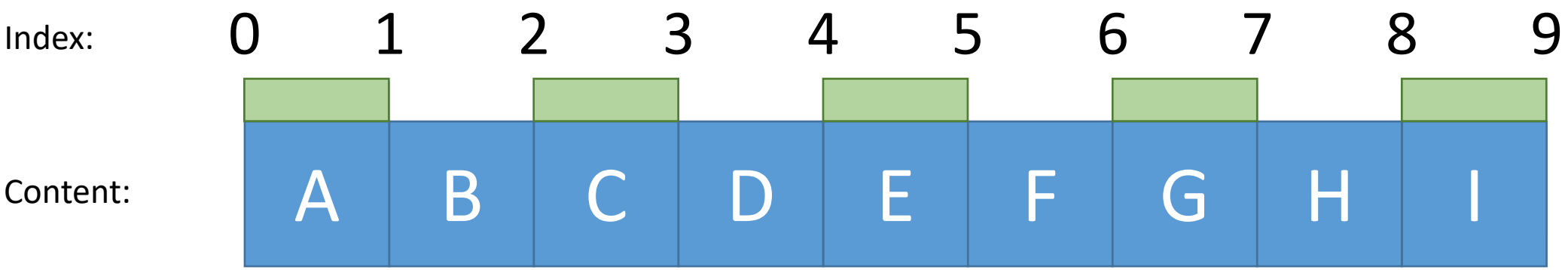

```
data[0:10:2]
```

```
['A', 'C', 'E', 'G', 'I']
```

# Indexing, cropping, subsets

- The step-size allows skipping elements

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[0:10:2]
```

```
['A', 'C', 'E', 'G', 'I']
```

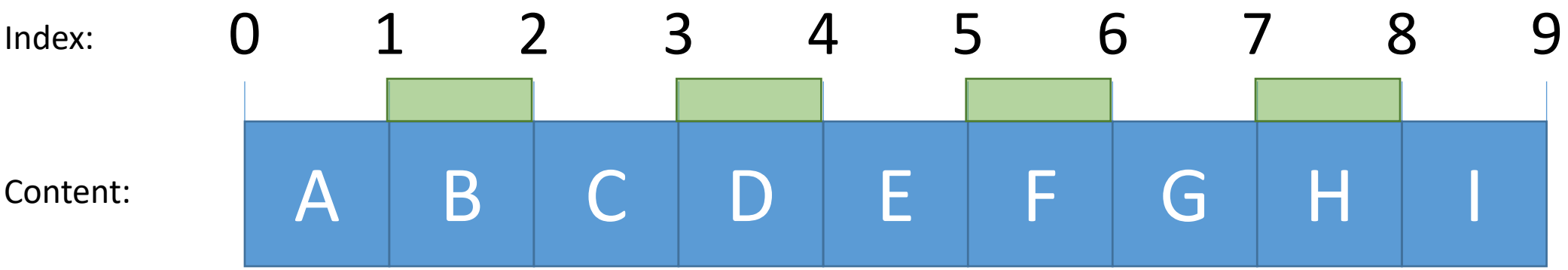
```
data[::2]
```

```
['A', 'C', 'E', 'G', 'I']
```

# Indexing, cropping, subsets

- The step-size allows skipping elements

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[0:10:2]
```

```
['A', 'C', 'E', 'G', 'I']
```

```
data[::2]
```

```
['A', 'C', 'E', 'G', 'I']
```

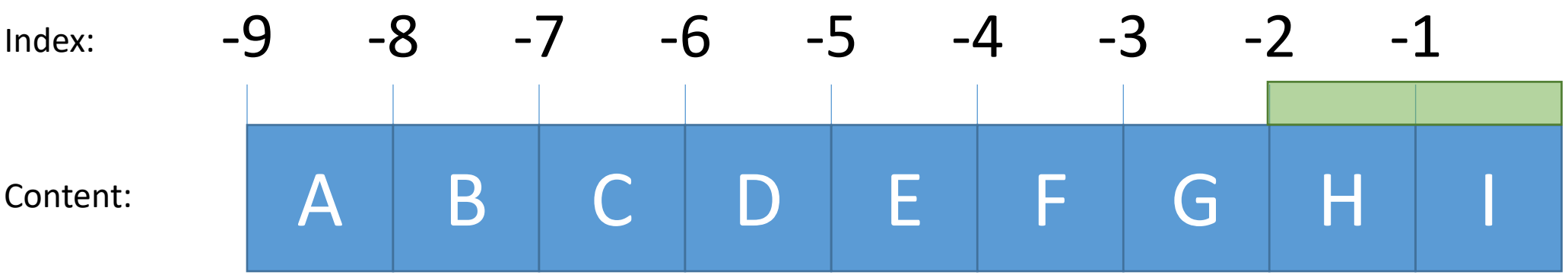
```
data[1::2]
```

```
['B', 'D', 'F', 'H']
```

# Indexing, cropping, subsets

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



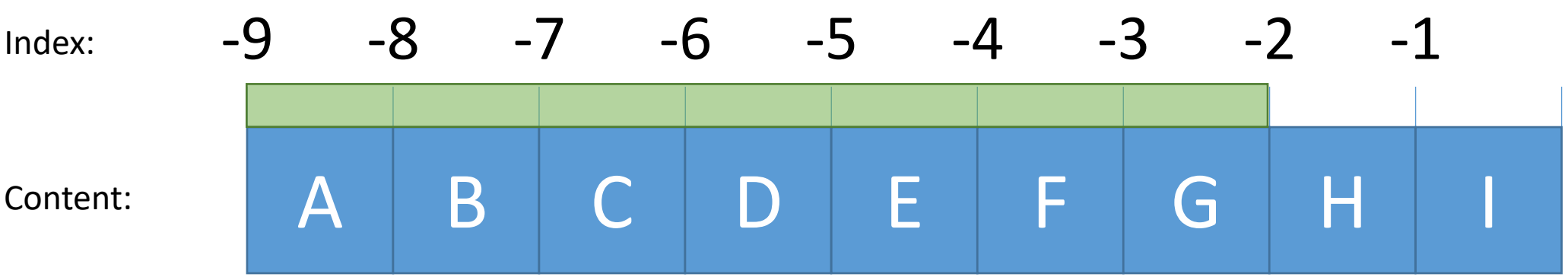

```
data[-2:]
```

```
['H', 'I']
```

# Indexing, cropping, subsets

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[-2:]
```

```
['H', 'I']
```

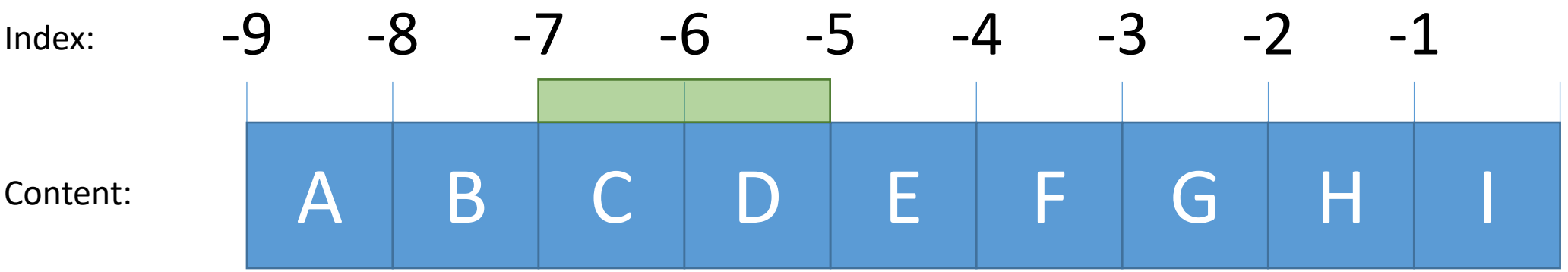
```
data[:-2]
```

```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

# Indexing, cropping, subsets

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[-2:]
```

```
['H', 'I']
```

```
data[:-2]
```

```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

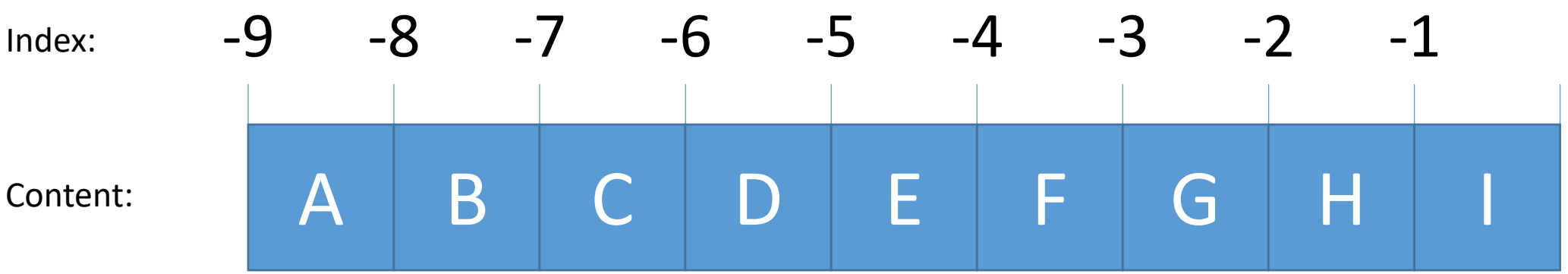
```
data[-7:-5]
```

```
['C', 'D']
```

# Indexing, cropping, subsets

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[-2:]
```

```
['H', 'I']
```

```
data[:-2]
```

```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

```
data[-7:-5]
```

```
['C', 'D']
```

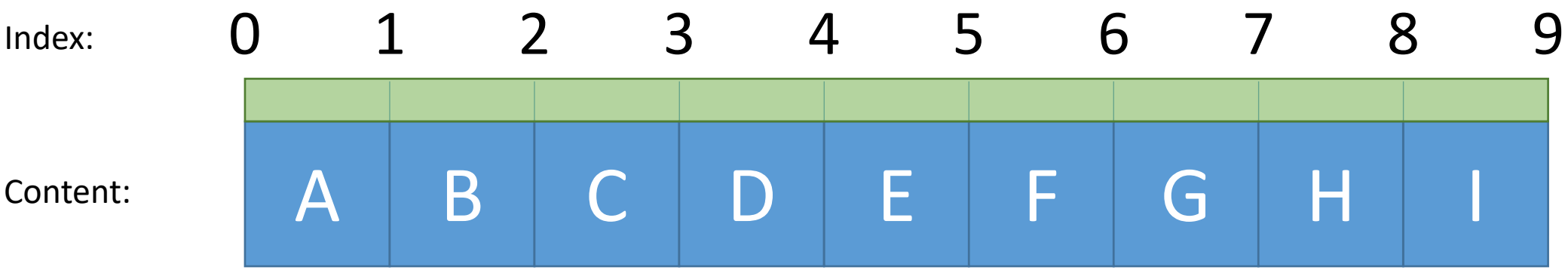
```
data[-5:-7]
```

```
[]
```

# Indexing, cropping, subsets

- Negative stepping also works

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



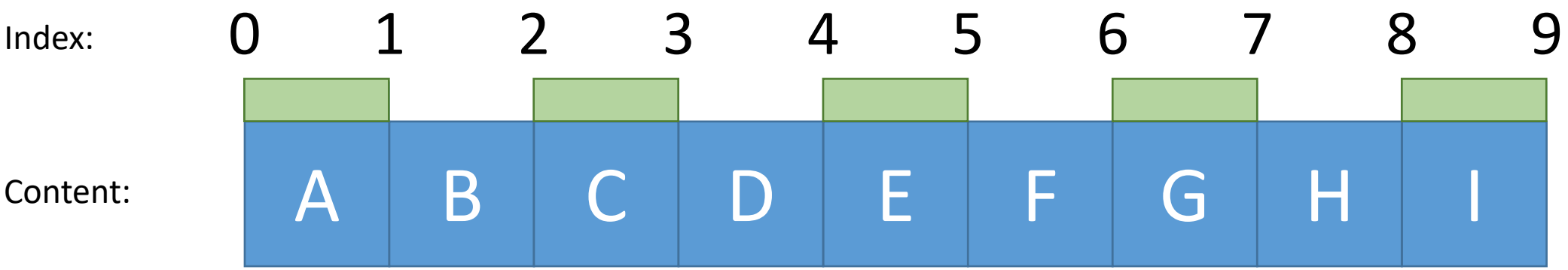

```
data[::-1]
```

```
['I', 'H', 'G', 'F', 'E', 'D', 'C', 'B', 'A']
```

# Indexing, cropping, subsets

- Negative stepping also works

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```




```
data[::-1]
```

```
['I', 'H', 'G', 'F', 'E', 'D', 'C', 'B', 'A']
```

```
data[::-2]
```

```
['I', 'G', 'E', 'C', 'A']
```

# Arrays in Python

- Modifying lists entries

```
numbers = [0, 1, 2, 3, 4]

# write in one array element
numbers[1] = 5

print(numbers)
```

```
[0, 5, 2, 3, 4]
```

Note: The first element has index 0!

- Creating lists of defined size

What?

How many?

```
zeros = [0] * 10
print(zeros)
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

- Concatenating lists

```
ones = [1, 1, 1]
twos = [2, 2, 2, 2]

# concatenate arrays
numbers = ones + twos

print(numbers)
```

```
[1, 1, 1, 2, 2, 2, 2]
```

+ means appending

# Lists versus Tuples

- Lists can be modified

```
measurements = [5.5, 6.3, 7.2, 8.0, 8.8]
```

```
measurements[1] = 25
```

```
measurements.append(10.2)
```

```
measurements
```

```
]: [5.5, 25, 7.2, 8.0, 8.8, 10.2]
```

- Tuples not

Note: round brackets

```
immutable = (4, 3, 7.8)
```

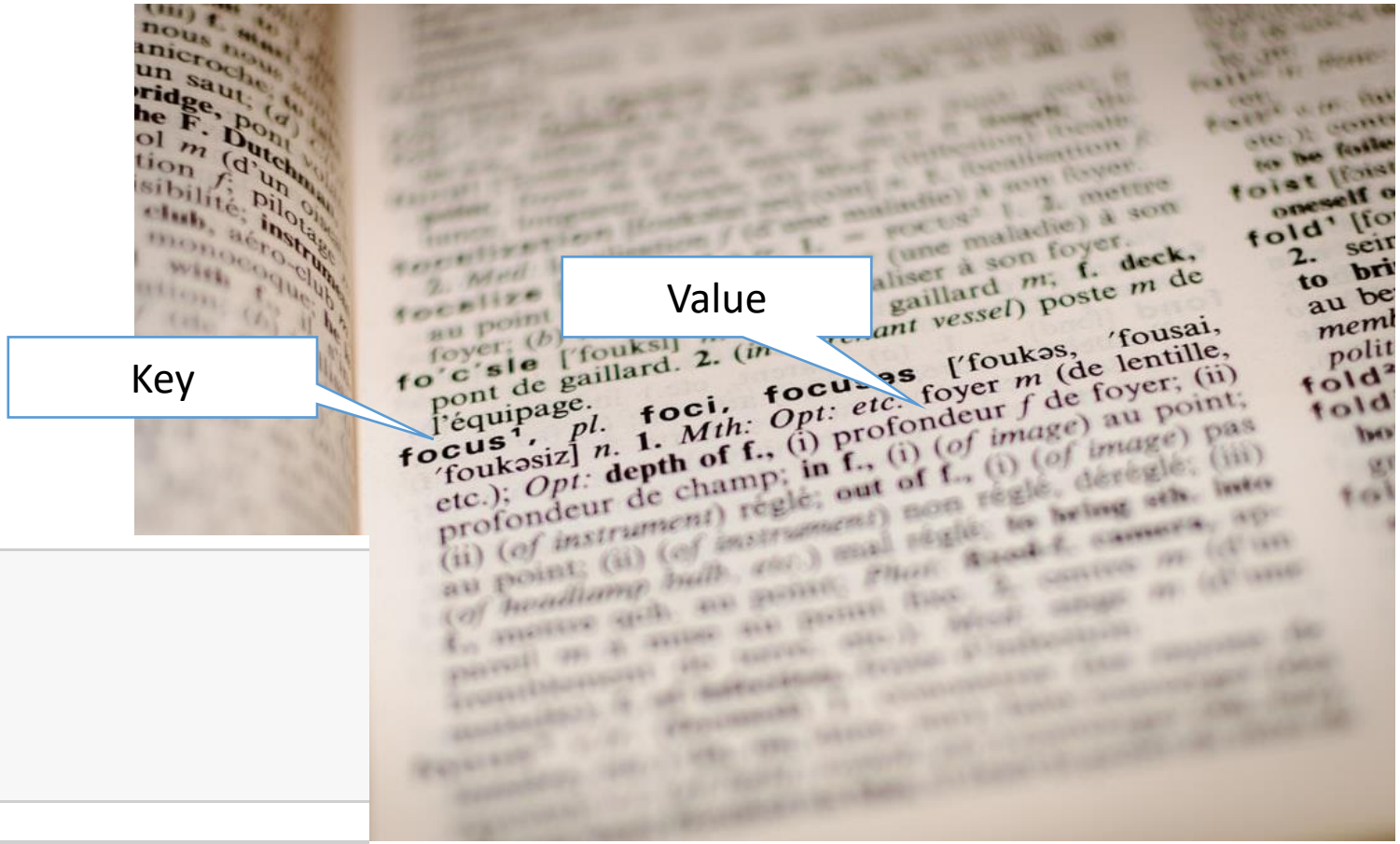
```
immutable[1] = 5
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-49-a01b13633c23> in <module>
----> 1 immutable[1] = 5

TypeError: 'tuple' object does not support item assignment
```

# Dictionaries

- Dictionary: a list of key-value pairs




Key Value

```
german_english_dictionary = {
    'Vorlesung':'Lecture',
    'Gleichung':'Equation'
}
```

```
german_english_dictionary
```

```
]: {'Vorlesung': 'Lecture', 'Gleichung': 'Equation'}
```



Image source: Romain Vignes

# Dictionaries

- Dictionary: a list of key-value pairs

```
german_english_dictionary = {
    'Vorlesung':'Lecture',
    'Gleichung':'Equation'
}
```

- Look up something in the dictionary: it's an array with named entries!

```
german_english_dictionary['Vorlesung']
```

```
'Lecture'
```

# Tables

- Tables can be dictionaries with lists as values

```
measurements_week = {
    'Monday':[2.3, 3.1, 5.6],
    'Tuesday':[1.8, 7.0, 4.3],
    'Wednesday':[4.5, 1.5, 3.2],
    'Thursday':[1.9, 2.0, 6.4],
    'Friday':[4.4, 2.3, 5.4]
}
```

```
measurements_week
```

```
]: {'Monday': [2.3, 3.1, 5.6],
    'Tuesday': [1.8, 7.0, 4.3],
    'Wednesday': [4.5, 1.5, 3.2],
    'Thursday': [1.9, 2.0, 6.4],
    'Friday': [4.4, 2.3, 5.4]}
```

- Retrieve a column

```
measurements_week['monday']
```

```
]: [2.3, 3.1, 5.6]
```

# Tables

- Sneak preview: By the mid of the course, we will work with Pandas DataFrames, *fancy* Tables.

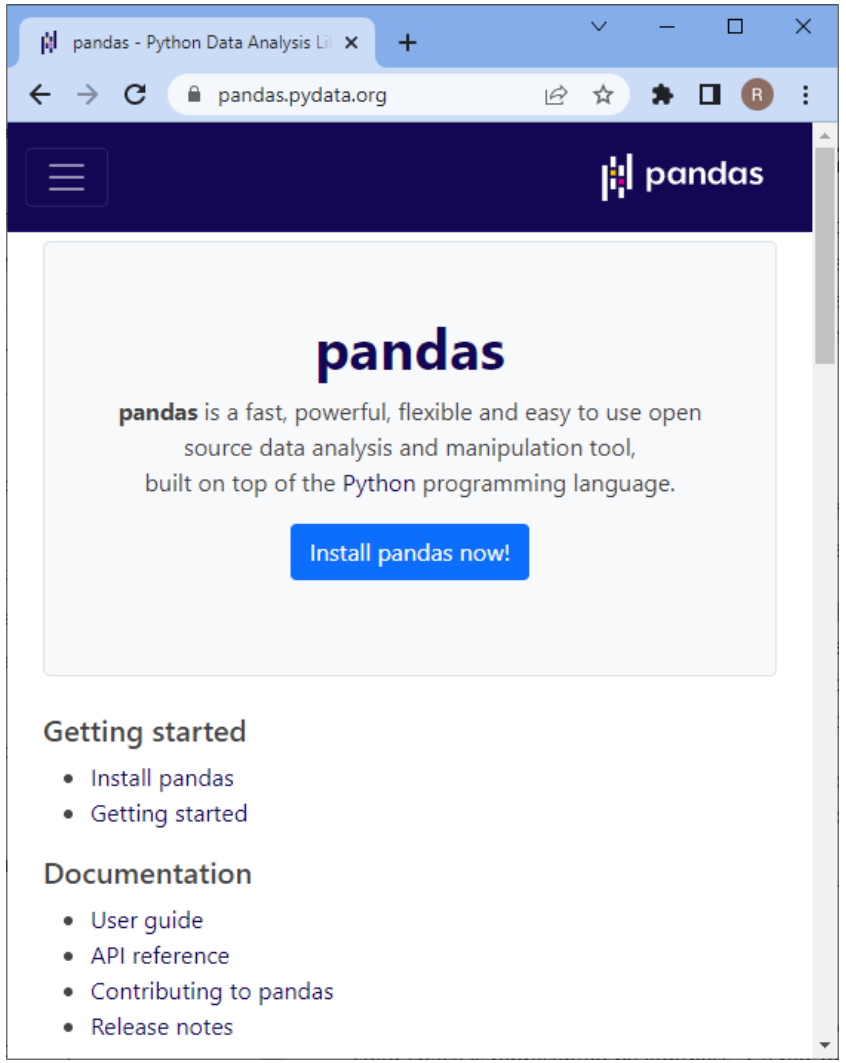
- `conda install pandas`

- Among many other features, Pandas allows to visualize tables nicely in Jupyter notebooks.

```
import pandas

pandas.DataFrame(measurements_week)
```

| | Monday | Tuesday | Wednesday | Thursday | Friday |
|---|---|---|---|---|---|
| 0 | 2.3 | 1.8 | 4.5 | 1.9 | 4.4 |
| 1 | 3.1 | 7.0 | 1.5 | 2.0 | 2.3 |
| 2 | 5.6 | 4.3 | 3.2 | 6.4 | 5.4 |

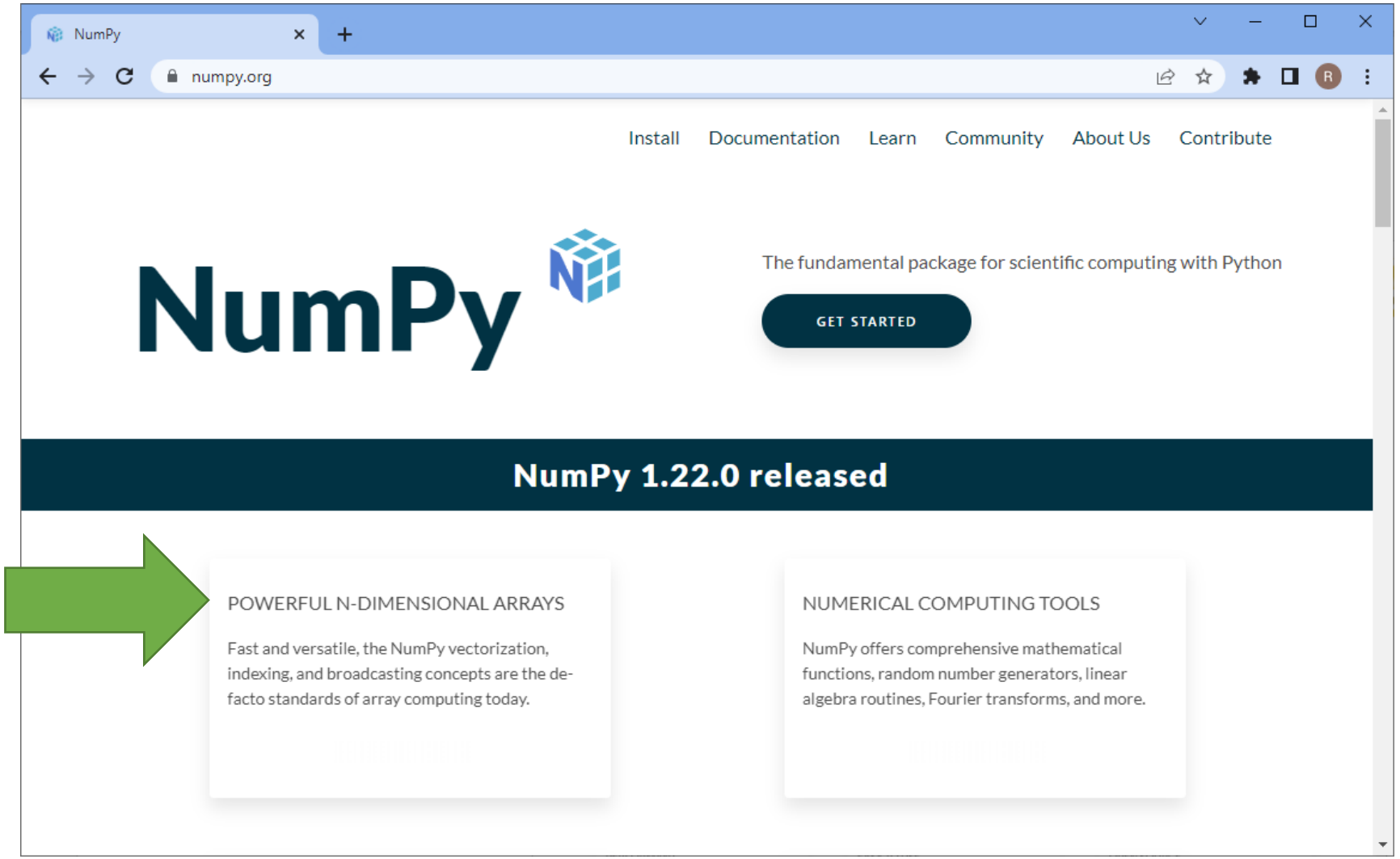



https://pandas.pydata.org/

# numpy

- The fundamental package for scientific computing with python.

- `conda install numpy`






https://numpy.org/

# numpy

- Simplifying mathematical operations on n-dimensional arrays

- Python arrays of arrays (lists of lists)

```
# multidimensional arrays
matrix = [
    [1, 2, 3],
    [2, 3, 4],
    [3, 4, 5]
]

print(matrix)
```

```
[[1, 2, 3], [2, 3, 4], [3, 4, 5]]
```

```
result = matrix * 2
print(result)
```

```
[[1, 2, 3], [2, 3, 4], [3, 4, 5], [1, 2, 3], [2, 3, 4], [3, 4, 5]]
```

- numpy arrays

Tell python that you want to use a library called numpy

```
import numpy as np

np_matrix = np.asarray(matrix)

print(np_matrix)
```

If “numpy” is to long, you can give an alias “np”

```
[[1 2 3]
 [2 3 4]
 [3 4 5]]
```

```
np_result = np_matrix * 2
print(np_result)
```

```
[[ 2  4  6]
 [ 4  6  8]
 [ 6  8 10]]
```

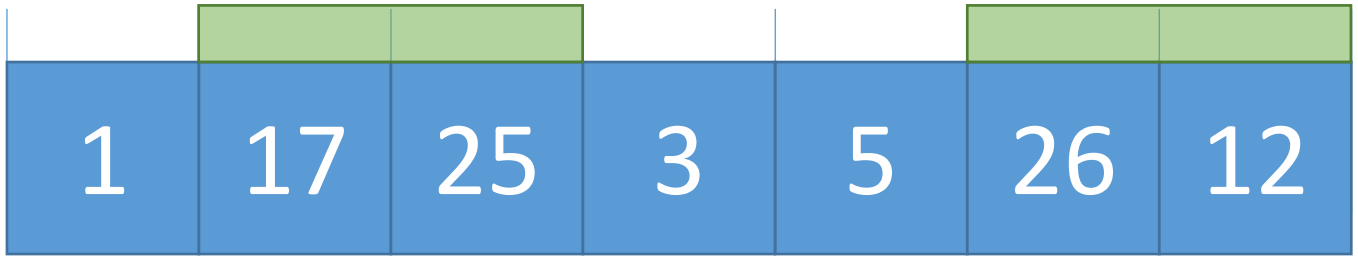
# Masking

- “Masking” is addressing certain elements in numpy arrays, e.g. depending on their content

```
import numpy
measurements = numpy.asarray([1, 17, 25, 3, 5, 26, 12])
measurements
```

```
array([ 1, 17, 25,  3,  5, 26, 12])
```

Content:




```
mask = measurements > 10
mask
```

```
array([False,  True,  True, False, False,  True,  True])
```

```
measurements[mask]
```

```
array([17, 25, 26, 12])
```

# Basic descriptive statistics using numpy

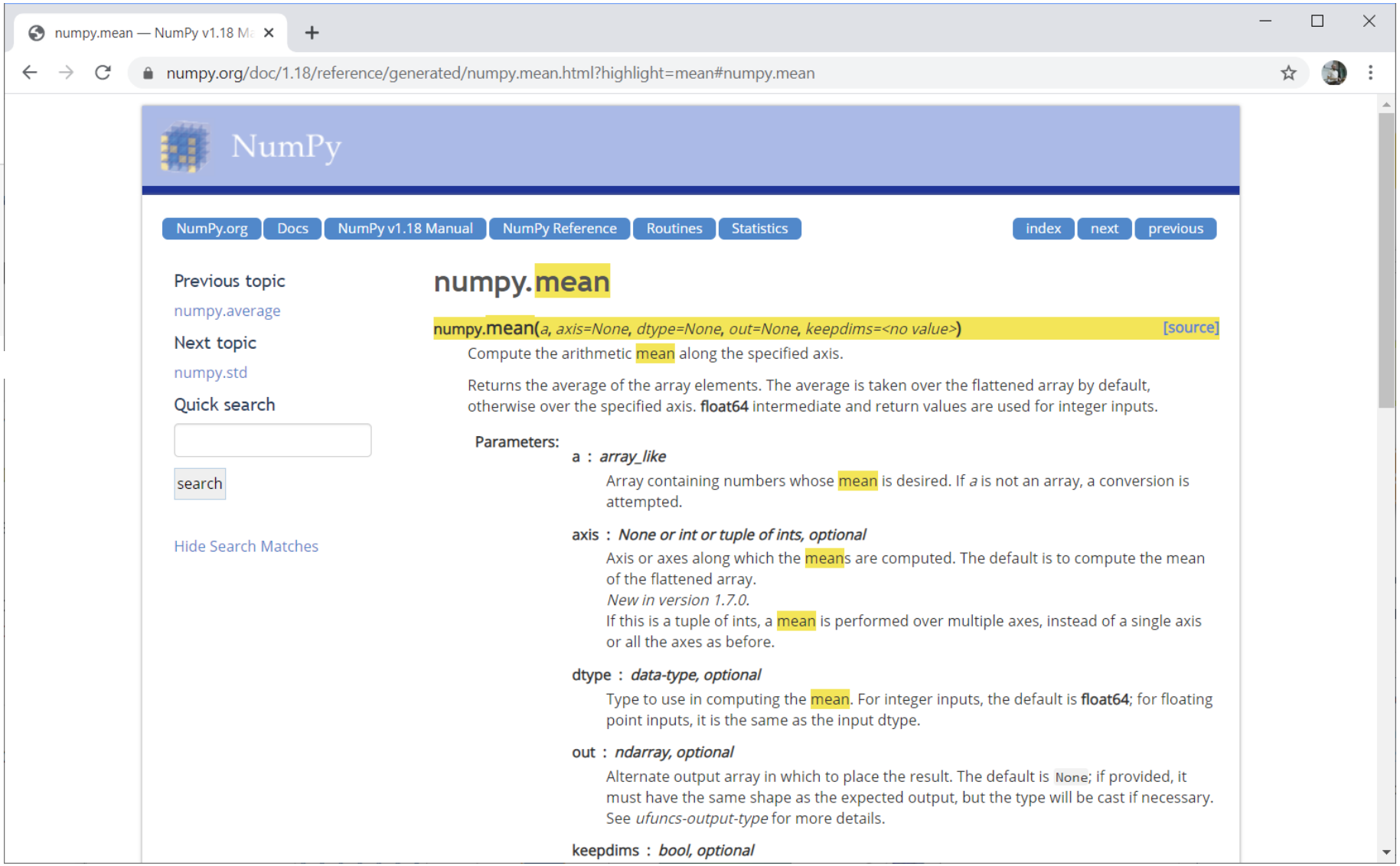
- Basic descriptive statistics

```
import numpy as np

measurements = [1, 4, 6, 7, 2]

mean = np.mean(measurements)
print("Mean: " + str(mean))
```

```
Mean: 4.0
```



# Troubleshooting

If your program throws error messages:

- Don't panic.
- *"There are two ways to write error-free programs; only the third one works."*

  Alan J. Perlis, Yale University

- Read where the error happened.
  - You may see your fault immediately, when looking at the right point.
- Read what appears to be wrong.
  - If you know, what's missing, you may see it, even if it's missing in a slightly different place.
  - Sometimes, something related is missing

```
print(round(4.5)
```

```
  File "<ipython-input-15-09a9be4a90c5>", line 1
    print(round(4.5)
                    ^
SyntaxError: unexpected EOF while parsing
```

# Summary

Take home messages

- Lists can be accessed like this:

```
data[start:stop:step]
```

- Strings are lists of characters
- Tuples are immutable lists
- Dictionaries are lists with named elements
- Columns in tables are lists
- Images are multi-dimensional lists

- Learning how to deal with lists in Python is key.