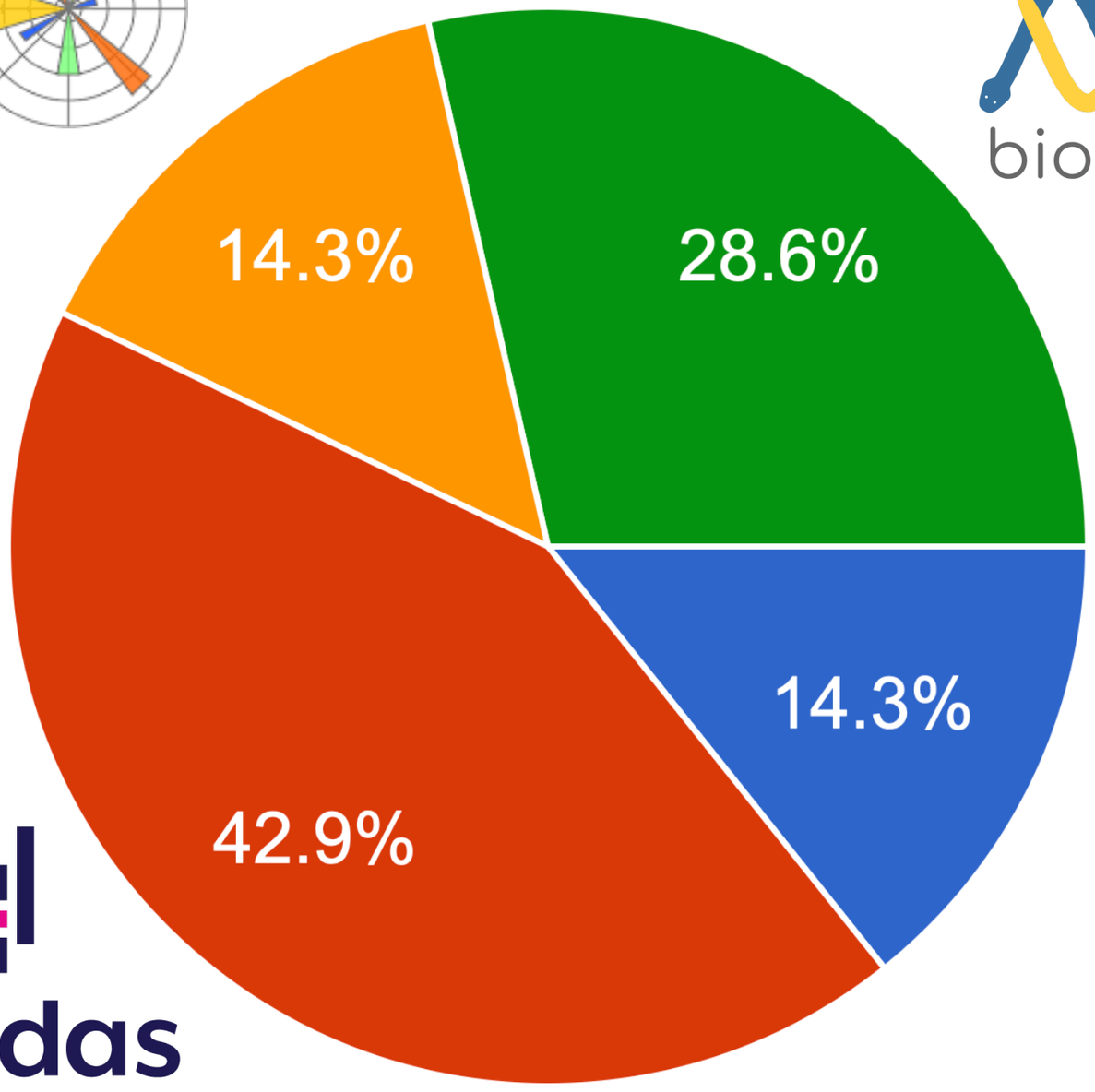
Poll

- NumPy
- Pandas
- Matplotlib
- Biopython
- More snakemake
- Conda



pandas

Series

```
>>> import numpy as np
>>> import pandas as pd
>>> s = pd.Series([1, 3, 5, np.nan, 6, 8])
>>> s
0    1.0
1    3.0
2    5.0
3    NaN
4    6.0
5    8.0
dtype: float64
```

DataFrame

```
>>> df = pd.DataFrame({
...     "chr": 1,
...     "contig": ['contig1'] * 3 + ['contig2'] * 2 + ['contig3'],
...     "REF": ['A', 'G', 'A', 'C', 'T', 'N'],
...     "ALT": [[], ['A'], ['G', 'C'], ['T'], ['A'], []],
...     "genotypes": [
...         ['0/0', '0/0', '0/1', './.'],
...         ['0/1', '0/0', '1/1', '0/1'],
...         ['0/0', '1/1', '0/1', '2/0'],
...         ['1/1', './.', './.', '0/0'],
...         ['0/1', '0/1', '0/0', './1'],
...         ['0/.', '0/0', '0/0', '0/0']
...     ]
... })
>>> df
   chr   contig REF     ALT            genotypes
0    1  contig1   A      []  [0/0, 0/0, 0/1, ./.]
1    1  contig1   G     [A]  [0/1, 0/0, 1/1, 0/1]
2    1  contig1   A  [G, C]  [0/0, 1/1, 0/1, 2/0]
3    1  contig2   C     [T]  [1/1, ./., ./., 0/0]
4    1  contig2   T     [A]  [0/1, 0/1, 0/0, ./1]
5    1  contig3   N      []  [0/., 0/0, 0/0, 0/0]
```

Column selection

```
>>> df.REF
0    A
1    G
2    A
3    C
4    T
5    N
Name: REF, dtype: object
```

Series

```
>>> df[['REF', 'contig']]
  REF   contig
0   A  contig1
1   G  contig1
2   A  contig1
3   C  contig2
4   T  contig2
5   N  contig3
```

Dataframe

Column assignment

```
>>> df['INFO'] = 'foo'
>>> df
   chr   contig REF     ALT             genotypes INFO
0    1  contig1   A      []  [0/0, 0/0, 0/1, ./.]  foo
1    1  contig1   G     [A]  [0/1, 0/0, 1/1, 0/1]  foo
2    1  contig1   A  [G, C]  [0/0, 1/1, 0/1, 2/0]  foo
3    1  contig2   C     [T]  [1/1, ./., ./., 0/0]  foo
4    1  contig2   T     [A]  [0/1, 0/1, 0/0, ./1]  foo
5    1  contig3   N      []  [0/., 0/0, 0/0, 0/0]  foo
```

```
>>> df['alleles'] = df.apply(lambda r: [r.REF] + r.ALT, axis=1)
>>> df
   chr   contig REF     ALT             genotypes INFO    alleles
0    1  contig1   A      []  [0/0, 0/0, 0/1, ./.]  foo        [A]
1    1  contig1   G     [A]  [0/1, 0/0, 1/1, 0/1]  foo     [G, A]
2    1  contig1   A  [G, C]  [0/0, 1/1, 0/1, 2/0]  foo  [A, G, C]
3    1  contig2   C     [T]  [1/1, ./., ./., 0/0]  foo     [C, T]
4    1  contig2   T     [A]  [0/1, 0/1, 0/0, ./1]  foo     [T, A]
5    1  contig3   N      []  [0/., 0/0, 0/0, 0/0]  foo        [N]
```

Row selection

```
>>> df = pd.DataFrame({
...     'A': [10, 20, 30, 40, 50],
...     'B': [60, 70, 80, 90, 100]
... }, index=['a', 'b', 'c', 'd', 'e'])
>>> df.loc['b':'d']
    A   B
b  20  70
c  30  80
d  40  90
>>> df.iloc[1:4]
    A   B
b  20  70
c  30  80
d  40  90
```

Row assignment

```
>>> df.loc[5, 'REF'] = 'GAGA'
>>> df
   chr   contig   REF     ALT             genotypes INFO    alleles
0    1  contig1     A      []  [0/0, 0/0, 0/1, ./.]  foo        [A]
1    1  contig1     G     [A]  [0/1, 0/0, 1/1, 0/1]  foo     [G, A]
2    1  contig1     A  [G, C]  [0/0, 1/1, 0/1, 2/0]  foo  [A, G, C]
3    1  contig2     C     [T]  [1/1, ./., ./., 0/0]  foo     [C, T]
4    1  contig2     T     [A]  [0/1, 0/1, 0/0, ./1]  foo     [T, A]
5    1  contig3  GAGA      []  [0/., 0/0, 0/0, 0/0]  foo        [N]
```

```
>>> df.loc[6] = dict(chr=1, contig='contig4', REF='A', ALT=['G'])
>>> df
   chr   contig   REF     ALT             genotypes INFO    alleles
0    1  contig1     A      []  [0/0, 0/0, 0/1, ./.]  foo        [A]
1    1  contig1     G     [A]  [0/1, 0/0, 1/1, 0/1]  foo     [G, A]
2    1  contig1     A  [G, C]  [0/0, 1/1, 0/1, 2/0]  foo  [A, G, C]
3    1  contig2     C     [T]  [1/1, ./., ./., 0/0]  foo     [C, T]
4    1  contig2     T     [A]  [0/1, 0/1, 0/0, ./1]  foo     [T, A]
5    1  contig3  GAGA      []  [0/., 0/0, 0/0, 0/0]  foo        [N]
6    1  contig4     A     [G]                   NaN  NaN        NaN
```

Masks

```
>>> df
   chr   contig REF     ALT             genotypes
0    1  contig1   A      []  [0/0, 0/0, 0/1, ./.]
1    1  contig1   G     [A]  [0/1, 0/0, 1/1, 0/1]
2    1  contig1   A  [G, C]  [0/0, 1/1, 0/1, 2/0]
3    1  contig2   C     [T]  [1/1, ./., ./., 0/0]
4    1  contig2   T     [A]  [0/1, 0/1, 0/0, ./1]
5    1  contig3   N      []  [0/., 0/0, 0/0, 0/0]
```

```
>>> df['REF'] != 'A'
0    False
1     True
2    False
3     True
4     True
5     True
Name: REF, dtype: bool
```

```
>>> (df['REF'] != 'A') | df.apply(lambda row: 'G' in row.ALT, axis=1)
0    False
1     True
2     True
3     True
4     True
5     True
dtype: bool
```

Subsetting

```
>>> df[df.REF != 'N']
   chr    contig REF      ALT                genotypes
0    1  contig1   A       []  [0/0, 0/0, 0/1, ./.]
1    1  contig1   G      [A]  [0/1, 0/0, 1/1, 0/1]
2    1  contig1   A  [G, C]  [0/0, 1/1, 0/1, 2/0]
3    1  contig2   C      [T]  [1/1, ./., ./., 0/0]
4    1  contig2   T      [A]  [0/1, 0/1, 0/0, ./1]
```

Reassign

```
>>> df.chr = 2
>>> df
   chr   contig REF     ALT             genotypes
0    2  contig1   X      []  [0/0, 0/0, 0/1, ./.]
1    2  contig1   G     [A]  [0/1, 0/0, 1/1, 0/1]
2    2  contig1   X  [G, C]  [0/0, 1/1, 0/1, 2/0]
3    2  contig2   C     [T]  [1/1, ./., ./., 0/0]
4    2  contig2   T     [A]  [0/1, 0/1, 0/0, ./1]
5    2  contig3   N      []  [0/., 0/0, 0/0, 0/0]
```

```
>>> df.loc[df['REF'] == 'A', 'REF'] = 'X'
>>> df
   chr   contig REF     ALT             genotypes
0    1  contig1   X      []  [0/0, 0/0, 0/1, ./.]
1    1  contig1   G     [A]  [0/1, 0/0, 1/1, 0/1]
2    1  contig1   X  [G, C]  [0/0, 1/1, 0/1, 2/0]
3    1  contig2   C     [T]  [1/1, ./., ./., 0/0]
4    1  contig2   T     [A]  [0/1, 0/1, 0/0, ./1]
5    1  contig3   N      []  [0/., 0/0, 0/0, 0/0]
```

Importing and exporting

```
>>> df.to_csv('sites.csv')
>>> df2 = pd.read_csv('sites.csv', index_col=0)
>>> df2
   chr    contig REF         ALT                        genotypes
0    2  contig1   X          []  ['0/0', '0/0', '0/1', './.']
1    2  contig1   G       ['A']  ['0/1', '0/0', '1/1', '0/1']
2    2  contig1   X  ['G', 'C']  ['0/0', '1/1', '0/1', '2/0']
3    2  contig2   C       ['T']  ['1/1', './.', './.', '0/0']
4    2  contig2   T       ['A']  ['0/1', '0/1', '0/0', './1']
5    2  contig3   N          []  ['0/.', '0/0', '0/0', '0/0']
```

Viewing data

```
>>> df.head(n=3)
   chr    contig REF     ALT                genotypes
0    2  contig1   X      []  [0/0, 0/0, 0/1, ./.]
1    2  contig1   G     [A]  [0/1, 0/0, 1/1, 0/1]
2    2  contig1   X  [G, C]  [0/0, 1/1, 0/1, 2/0]
>>> df.tail(n=3)
   chr    contig REF  ALT                genotypes
3    2  contig2   C  [T]  [1/1, ./., ./., 0/0]
4    2  contig2   T  [A]  [0/1, 0/1, 0/0, ./1]
5    2  contig3   N   []  [0/., 0/0, 0/0, 0/0]
```

Operations

```
>>> df.chr.mean()
2.0

>>> df.chr.var()
0.0
```

Missing data

```
>>> None == None
True
>>> np.nan == np.nan
False
>>> pd.NA == pd.NA
<NA>
```

```
>>> pd.Series([1, 2, None], dtype='Int64')
0       1
1       2
2    <NA>
dtype: Int64
>>> pd.Series([1, 2, None])
0    1.0
1    2.0
2    NaN
dtype: float64
```

```
>>> list(map(type, [None, np.nan, pd.NA]))
[<class 'NoneType'>, <class 'float'>, <class 'pandas._libs.missing.NAType'>]
```

```
>>> pd.NA | True
True
>>> np.nan | True
Traceback (most recent call last):
  File "/Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/_pydevd_bundle/pydevd_exec2.py", line 3, in Exec
    exec(exp, global_vars, local_vars)
  File "<input>", line 1, in <module>
TypeError: unsupported operand type(s) for |: 'float' and 'bool'
```

Grouping

```
>>> df = pd.DataFrame({
...     'Category': ['Electronics', 'Furniture', 'Electronics', 'Furniture', 'Electronics'],
...     'Sales': [200, 150, 340, 124, 450]
... })
>>> df
      Category  Sales
0  Electronics    200
1    Furniture    150
2  Electronics    340
3    Furniture    124
4  Electronics    450
>>> df.groupby('Category').sum()
             Sales
Category
Electronics    990
Furniture      274
```

Merging

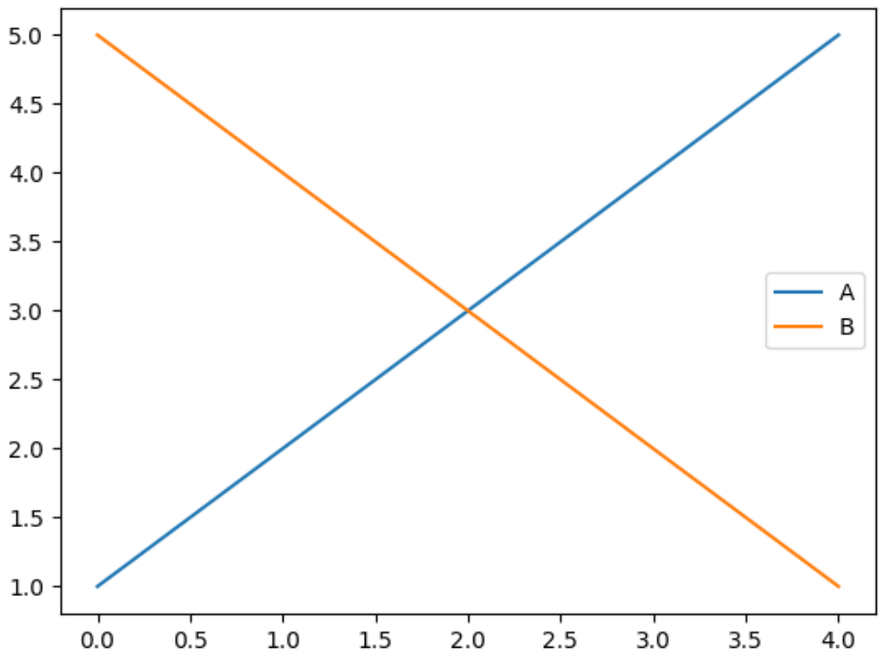
```
>>> df1 = pd.DataFrame({
...     'CustomerID': [1, 2, 3, 4],
...     'CustomerName': ['Alice', 'Bob', 'Charlie', 'David']
... })
>>> df2 = pd.DataFrame({
...     'CustomerID': [2, 4, 3, 5],
...     'PurchaseAmount': [100, 200, 150, 300]
... })
>>> pd.merge(df1, df2, on='CustomerID', how='inner')
   CustomerID CustomerName  PurchaseAmount
0           2          Bob             100
1           3      Charlie             150
2           4        David             200
```

Indices

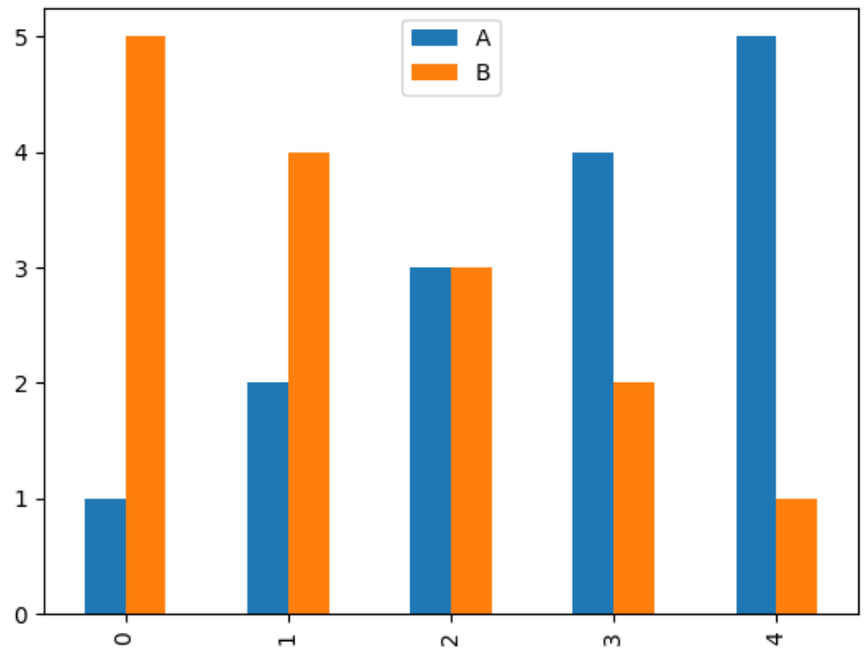
```
>>> df = pd.DataFrame({
...     'A': [10, 20, 30, 40, 50],
...     'B': [60, 70, 80, 90, 100]
... }, index=pd.MultiIndex.from_tuples([('a', 1), ('b', 1), ('c', 1), ('d', 3), ('b', 2)]))
>>> df.loc['b']
    A    B
1  20   70
2  50  100
>>> df.loc[('b', 1)]
A    20
B    70
Name: (b, 1), dtype: int64
>>> df.xs(1, level=1)
    A   B
a  10  60
b  20  70
c  30  80
```

Visualization

```
>>> import matplotlib.pyplot as plt
>>> pd.DataFrame({
...     'A': [1, 2, 3, 4, 5],
...     'B': [5, 4, 3, 2, 1]
... }).plot()
<Axes: >
>>> plt.show()
```




```
>>> pd.DataFrame({
...     'A': [1, 2, 3, 4, 5],
...     'B': [5, 4, 3, 2, 1]
... }).plot(kind='bar')
<Axes: >
>>> plt.show()
```



Exercises

```
>>> pd.options.mode.copy_on_write = False
>>> df = pd.DataFrame({"foo": [1, 2, 3], "bar": [4, 5, 6]})
>>> subset = df["foo"]
>>> subset.iloc[0] = 100
>>> df
   foo  bar
0  100    4
1    2    5
2    3    6
```

Copy-on-write

```
>>> pd.options.mode.copy_on_write = False
>>> df = pd.DataFrame({"foo": [1, 2, 3], "bar": [4, 5, 6]})
>>> subset = df["foo"]
>>> subset.iloc[0] = 100
>>> df
   foo  bar
0  100    4
1    2    5
2    3    6
```

```
>>> pd.options.mode.copy_on_write = True
>>> df = pd.DataFrame({"foo": [1, 2, 3], "bar": [4, 5, 6]})
>>> subset = df["foo"]
>>> subset.iloc[0] = 100
>>> df
   foo  bar
0    1    4
1    2    5
2    3    6
```

Exercise

- Load https://github.com/Sendrowski/BirchesScandinavia/blob/master/resources/birch_samples.csv
- Only retain *Betula pendula* samples
- Calculate average location (latitude, longitude) per population
- Create scatter plot of location (latitude, longitude) using Pandas
- Label by population name (location)
- Superimpose in plot using Pandas