## Getting started with lists

A list is an ordered and changeable sequence of elements. It can hold integers, characters, floats, strings, and even objects.

### Creating lists

A list is created with `[ ]`, elements separated by commas.

```
x = [1, 2, 3]
```

### List functions and methods

- `append(x)` adds an element `x` to the end of the list.
- `sort()` sorts the elements in the list.
- `reverse()` reverses the order of elements in the list.
- `len(list)` returns the number of elements in the list.

```
x.append(4)  # Add 4 to the end
x.sort()     # Sort the list
x.reverse()  # Reverse the list
len(x)       # Return the length of the list
```

### Selecting list elements

Python lists are zero-indexed (the first element has index 0). For ranges, the last index is excluded but the list is not.

```
x = ['a', 'b', 'c', 'd', 'e']
x[0]      # Get the first element
x[0:3]    # Get elements from index 0 to 2
x[1:]     # Get all elements from index 1
x[:3]     # Get all elements up to index 3
```

### Concatenating lists

```
x = ['a', 'b', 'c']
y = ['d', 'e', 'f']
x + y       # Concatenate two lists
```

### Getting started with dictionaries

A dictionary stores data using a key-value pair. That is, a list of elements which are indexed by Python, dictionary are indexed by key. The names of which must be unique.

```
d = {'a': 1, 'b': 2, 'c': 3}  # Create a dictionary
```

### Dictionary functions and methods

- `keys()` returns a view object that contains the keys.
- `values()` returns a view object that contains the values.

```
d.keys()  # Get keys as a list
```

### Selecting dictionary elements

```
d['a']  # Get the value of the dictionary
```

## NumPy arrays

NumPy is a python package for scientific computing. It provides multidimensional array objects and efficient operations on them. Let’s learn how to use the Python NumPy library.

```
x = np.array([1, 2, 3])
```

## Operators

### Arithmetic operators

- `+`  // Add two numbers
- `-`  // Subtract two numbers
- `*`  // Multiply two numbers
- `/`  // Divide a number by another

### Assignment operators

- `=`  // Assign a value to a variable

### Numeric comparison operators

- `<`  // Less than
- `>`  // Greater than
- `<=`  // Less than or equal to
- `>=`  // Greater than or equal to

### Logical operators

- `and`  // Logical and
- `or`  // Logical or
- `not`  // Logical not

# Getting started with characters and strings

- `len(str)`  // Returns the length of a string.
- `str[0]`  // Returns the first character of a string.
- `str[-1]`  // Returns the last character of a string.

### Combining and splitting strings

- `str + str`  // Concatenate two strings.
- `str.replace(old, new)`  // Replace all occurrences of `old` with `new` in `str`.

### Mutating strings

- `str[0] = 'C'`  // Replace the first character of `str` with `'C'`.

### Getting started with DataFrames

A DataFrame is an efficient and powerful package for data analysis and manipulation in Python. To import the package, you can use `import pandas as pd`. A pandas DataFrame is a structure that contains multi-dimensional data in one object.

```
df = pd.DataFrame({'a': [1, 2, 3], 'b': [4, 5, 6]})
```

### Creating DataFrames

- `pd.DataFrame(data, columns)`  // Creates a DataFrame from a dictionary

```
data = {'a': [1, 2, 3], 'b': [4, 5, 6]}
df = pd.DataFrame(data, columns=['a', 'b'])
```

### Selecting DataFrame Elements

```
df['a']  # Select a column
df.iloc[2]  # Select second row
```

### Manipulating DataFrames

- `df['a'] = df['a'] + 1`  // Increment the values in column `a`
- `df.drop('a', axis=1)`  // Drop column `a`
- `df['a'].sum()`  // Sum of values in column `a`
- `df['a'].mean()`  // Mean of values in column `a`
- `df['a'].max()`  // Maximum value in column `a`
- `df['a'].min()`  // Minimum value in column `a`
- `df['a'].count()`  // Number of non-null values in column `a`
- `df['a'].std()`  // Standard deviation of values in column `a`

## Functions and methods

- `np.mean(x)`  // Calculate the mean of elements in a NumPy array `x`.
- `np.std(x)`  // Calculate the standard deviation of elements in a NumPy array `x`.

```
mean = np.mean(x)
std = np.std(x)
```

## Getting started with Python Basics

Python is the most popular programming language in data science. It is easy to learn and comes with a wide array of powerful libraries for data analysis. This cheat sheet provides beginners and intermediate users a guide to starting using python, that is to jump-start your journey with python. If you want more notes and Python cheat sheets, check out the following cheat sheet.

## Importing packages

Python packages are a collection of useful tools developed by the opensource community. They extend the capabilities of the python language. To install a new package, for example, pandas, you can go to the package’s prompt.

```
import pandas
```

## The working directory

The working directory is the default file path that python notes or saves the data. An example of the working directory is `/data/`. If you navigate to this directory, you will see the data.

```
import os
os.getcwd()
```

## How to use this cheat sheet

- **Python basics**
- **Operators**
- **Getting started with lists**
- **Getting started with characters and strings**
- **Getting started with DataFrames**
- **Accessing help and getting object types**
- **Importing packages**
- **The working directory**
- **NumPy arrays**