SOCIETY OF PETROLEUM ENGINE
PORT HARCOURT SECTION

PYTHON FOR RESERVOIR ENGINEERING AND GEOSCIENCES

DESTINY OTTO & YOHANES NUWARA

SLIDE ONE

Courtesy: Edureka

AGENDA

Python

Applications of Python in the oil and gas industry

Getting Started with Python

Numpy, Pandas and Matplotlib

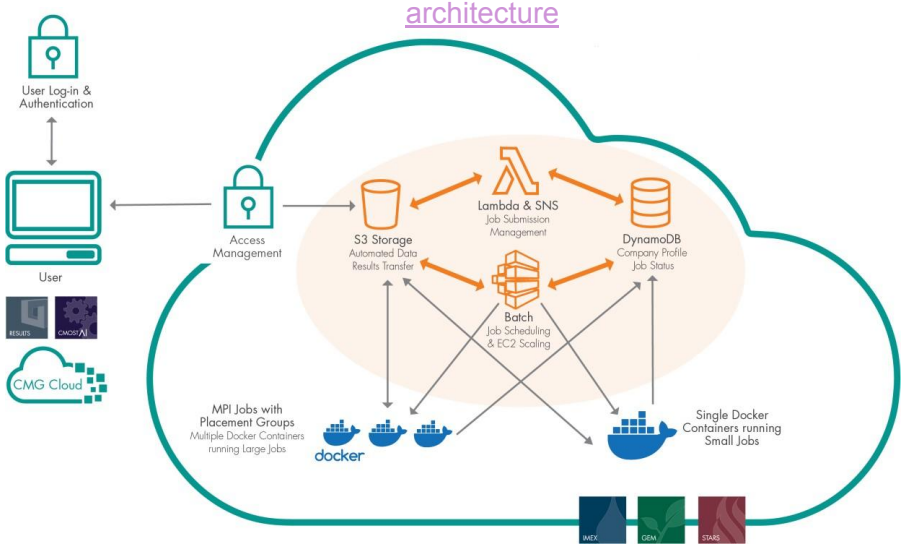
Why Python?

- Python is **open-source**
- Python is **easy** : It is built based on human language (feels like we're "having conversation" with the programs)
- Python is **fast and efficient** : The use of "list comprehension" instead of "for-loop" makes consuming much less lines of codes
- Python is **flexible** : We can build and deploy programs almost anywhere (in local PC
  systems and in the cloud)
- Python is **abundant** : 2 million ++ libraries and packages available to use
- Python is **scientific** : For scientific task, Python is as capable as scientific programming languages like MATLAB, or even better.

The use of Python in Oil and Gas Industries

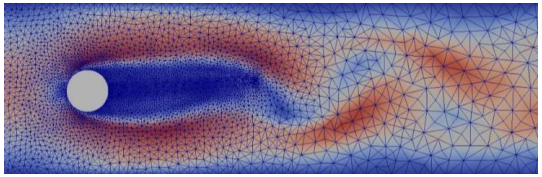
- **CGG GeoSoftware** develops a Python plugin that connects machine learning to their interpretation software
- **Schlumberger** develops Python editor that connects petrophysical calculation to their *Techlog* software
- **Equinor** actively works on Python libraries such as *segyio* for seismic data loader, *libecl* for reservoir simulation data loader, *dlisio*, etc.
- **Tech companies** help oil and gas industries to deploy Python in their cloud service, such as AWS
- **OPM** open-source reservoir simulator runs on Python

AWS and CMG simulation architecture

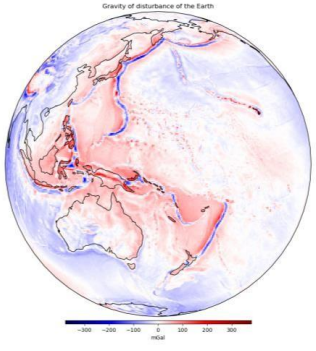


Python as a scientific Language

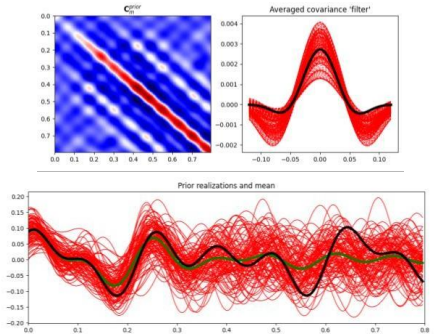
- **Acoustic/elastic wave simulation** based on various methods (finite-difference, pseudo-spectral, spectral) for **computational seismology** (Prof. Heiner Igel from LMU in Germany)
- *FeNiCs* **high performance computing** for **computational fluid dynamics** (KTH in Sweden)
- *Gekko* **numerical solvers and optimization** (Prof. John Hedengren from BYU in USA)
- *Pylops* package for **(mathematical) inversion of seismic data** (Matteo Ravasi from Equinor)
- *Fatiando a Terra* package for **computation in potential methods in geophysics** (Leonardo Uieda from University of Liverpool, UK)



Navier-Stokes simulation in FeNiCs



Gravity processing in Fatiando



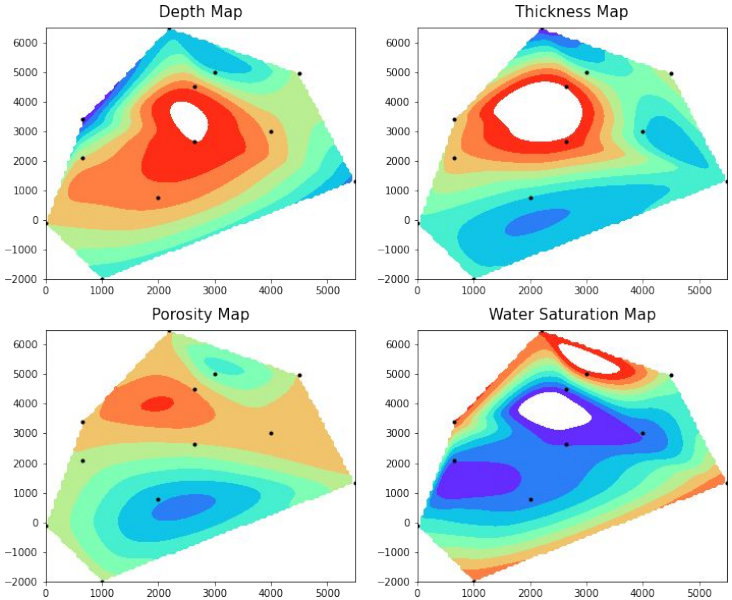
Bayesian inversion in Pylops

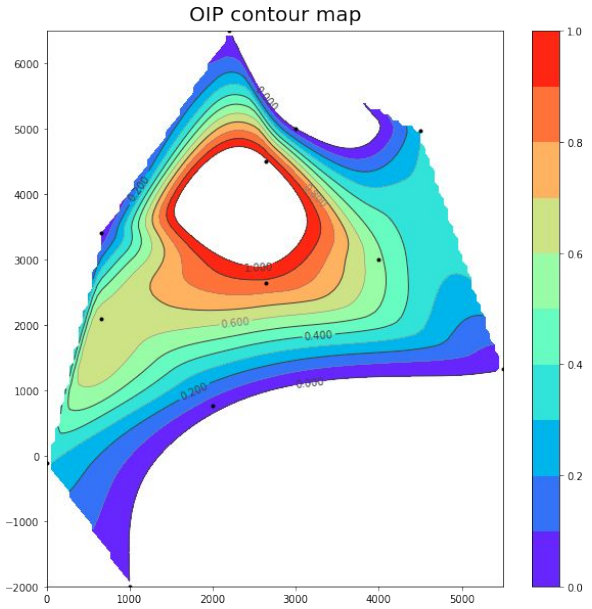
Open Datasets

- Many available open datasets that you can access: SEG Wiki, Geothermal Data Repository (GDR) OpenEi, KGS, MSEEL, and many more
- We have list down open datasets in: https://github.com/yohanesnuwara/open-geoscience- repository
- Python allows interfacing with data stored in websites (cloud computing in Google Colab)

Volumetrics with python

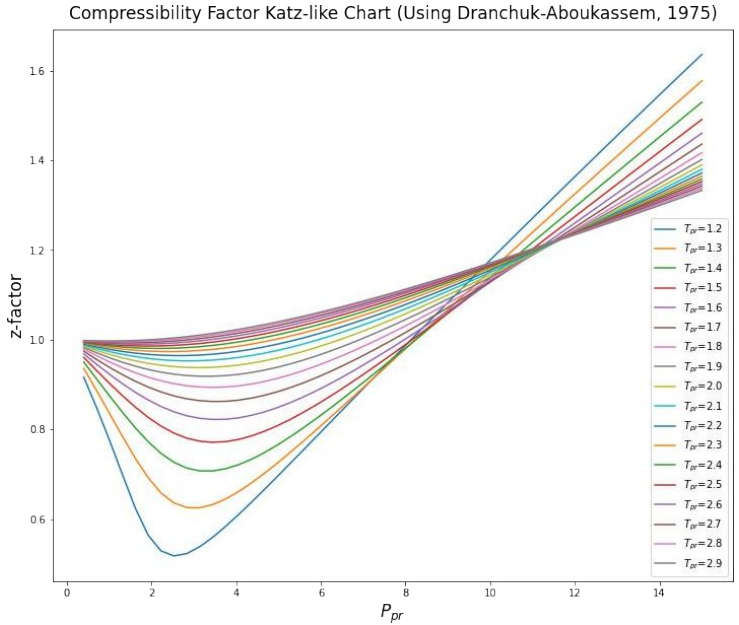
- Producing contour maps of petrophysical properties
- Calculating OOIP and OGIP using Green's theorem and geometric rules; Trapezoidal,
  Pyramidal, and Simpson's 1/3





PVT Calculator with python

- Creating a library of PVT correlations for reservoir gas, oil, and water
- Calculating PVT properties, such as z-factor of gas, formation volume factor (FVF),
  solution gas-oil ratio, isothermal compressibility, and viscosity

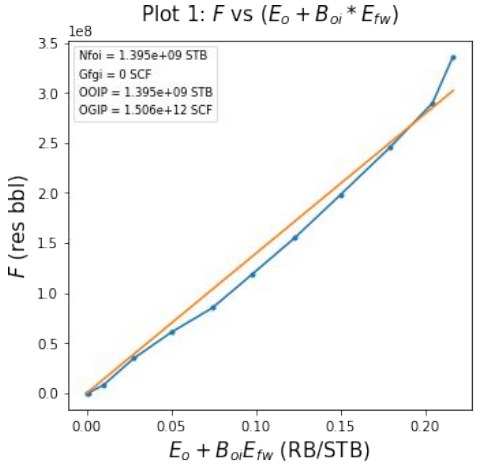


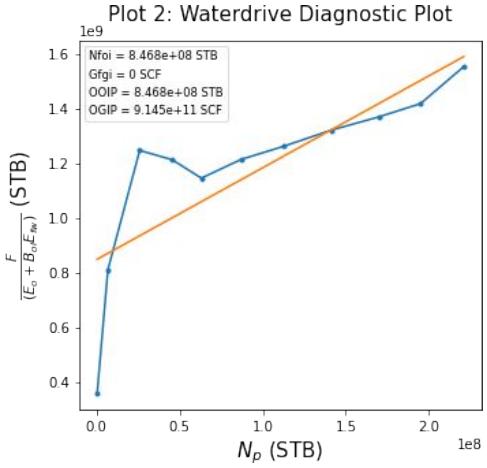

```
=== Gas PVT Correlation Calculator ===

Your Input:
Pressure                    : 1200 psia
Temperature                 : 120 °F
Specific Gravity            : 0.6
H2S Mole Fraction           : 0.07
CO2 Mole Fraction           : 0.1

PVT Output:
z-factor                    : 0.9119286750113232
Density                     : 3.6767036898581713 lb/ft3
FVF                         : 0.012457448652877104 res ft3/scf
Isothermal compressibility  : 836.7169364115891 microsip
Viscosity                   : 0.013328409015399147 cp
```

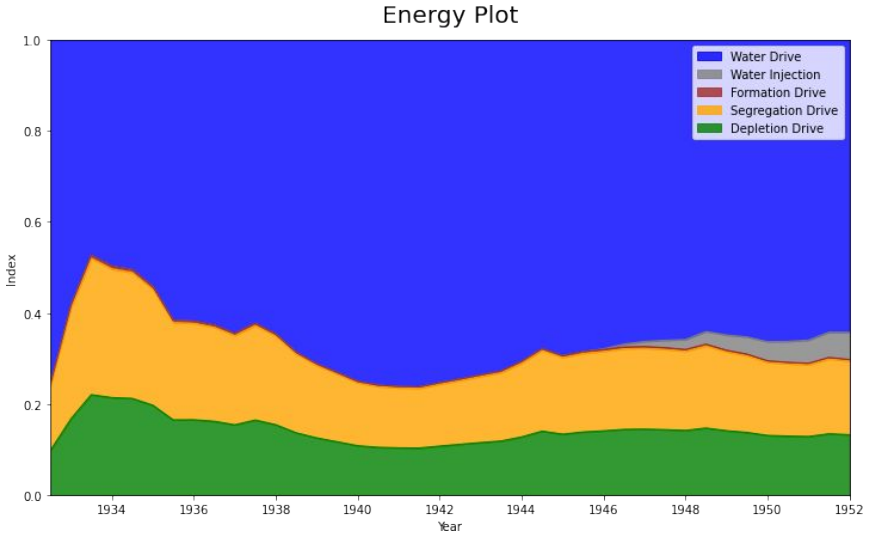
```
=== Oil PVT Correlation Calculator ===

Your Input:
Pressure                    : 1900 psia
Temperature                 : 220 °F
Specific Gravity            : 0.8
Gas-oil ratio @ Bubble-point : 500 scf/STB
Oil gravity                 : 30 API

PVT Output:
Bubble-point Pressure       : 2650.3067919543523 psi
Gas-oil ratio               : 347.44320213156914 scf/STB
FVF                         : 1.2298060072933186 RB/STB
Isothermal compressibility  : 33.100146317783555 microsip
Viscosity                   : 0.7777699805921316 cp
```

Material Balance Analysis with Python

- Calculating original gas (and condensate) in place from dry-gas and gas-condensate reservoirs
- Calculating original oil and gas in place from undersaturated and saturated oil reservoirs; volatile and non-volatile
- Calculating aquifer influx and reservoir drive indices

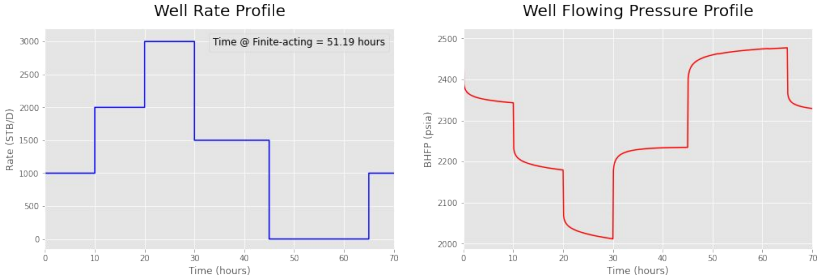


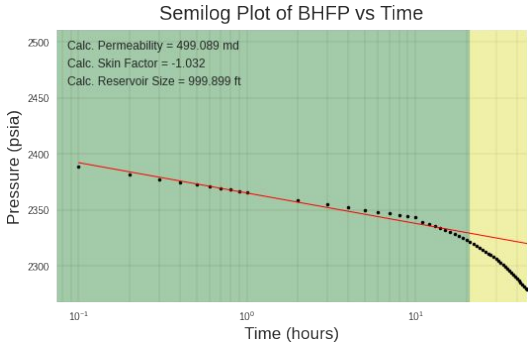


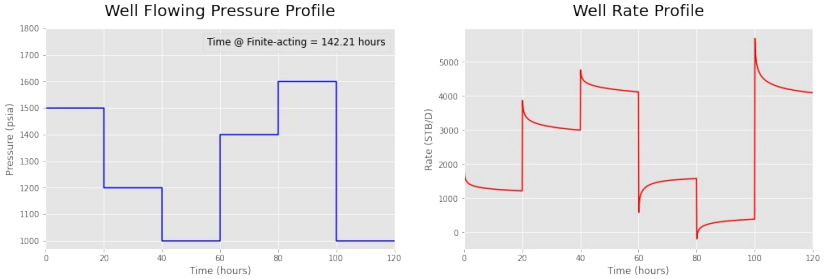


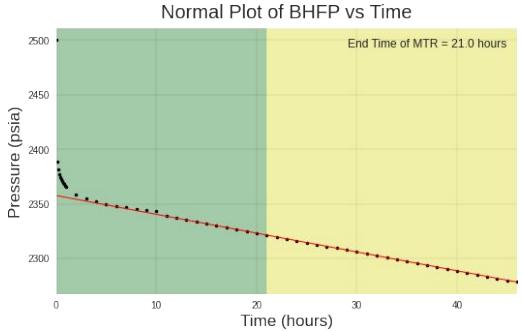
Well testing with Python

- Modeling well rate and pressure transient response from a simulated multi-rate pressure and rate tests; series of drawdowns and shut-ins
- Analyzing drawdown, buildup, and constant pressure tests





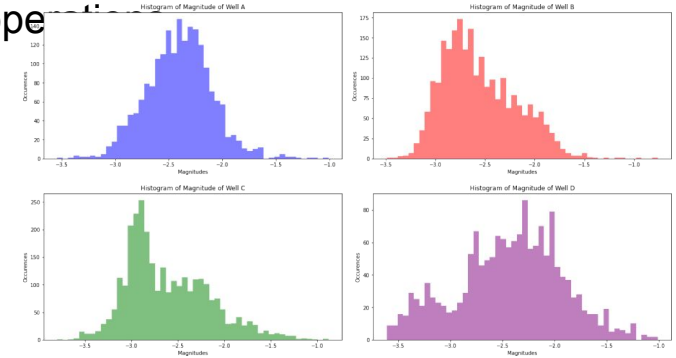


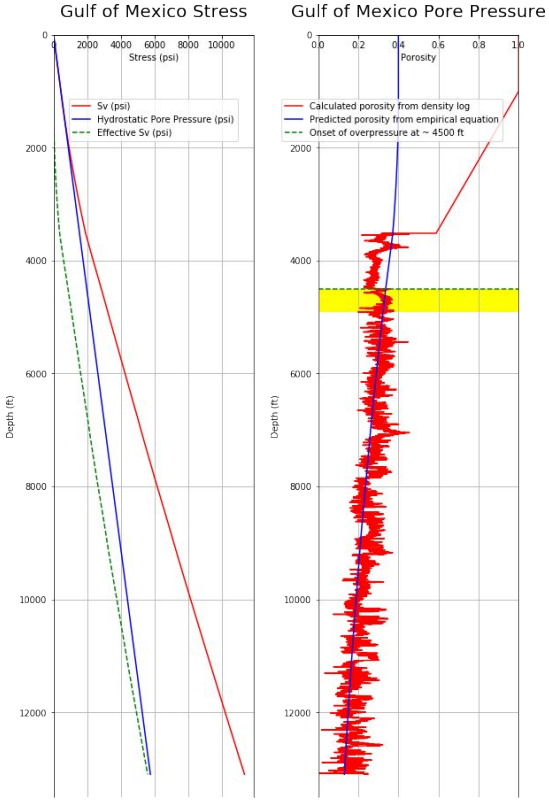


Geomechanics with Python

- Pore pressure estimation from geophysical logs (sonic and density) logs
- Stress bounds (Sv, SHmax, Shmin) estimation using stress diagram
- Mohr-Coulomb analysis of fault data (dip and azimuth) from image logs
- Measuring probability of induced seismicity from hydraulic fracturing ope

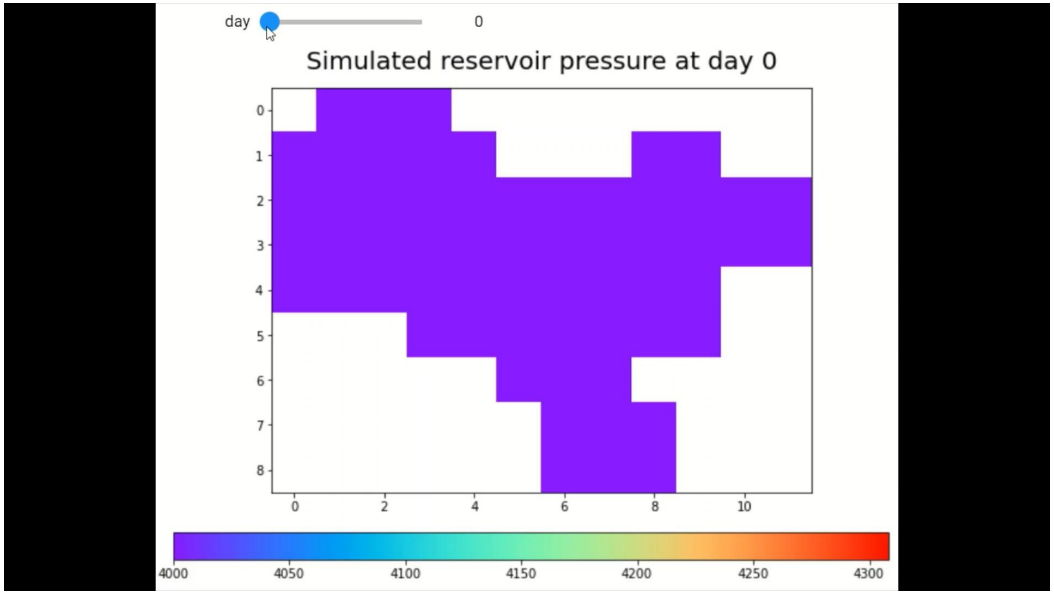
Taken from my project reservoir-geomechanics in GitHub





Reservoir Simulation with Python

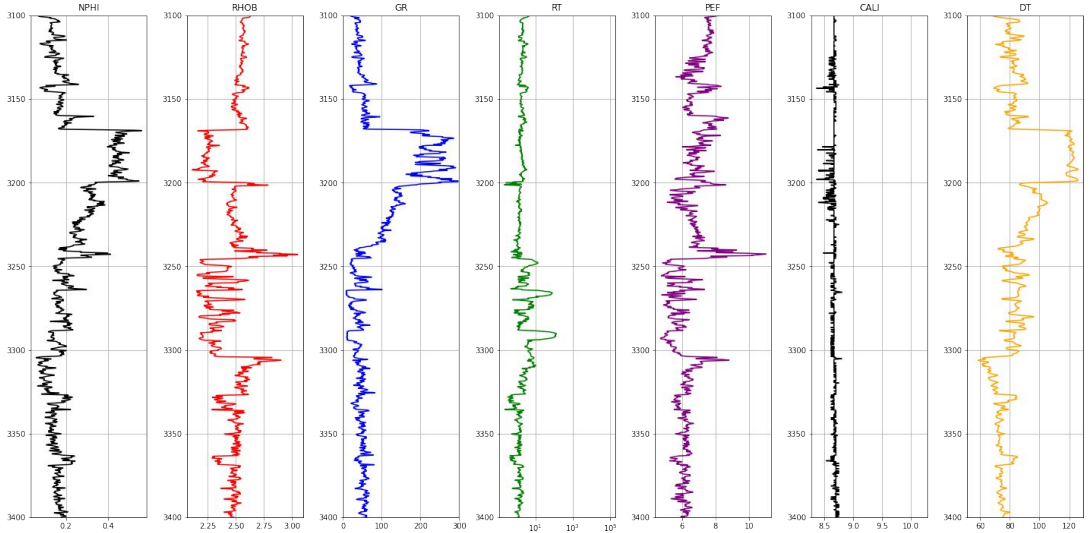
- Build a simple reservoir simulator with Python
- Solve single-phase simulation (incompressible, slightly compressible, and compressible); and multi-phase simulation

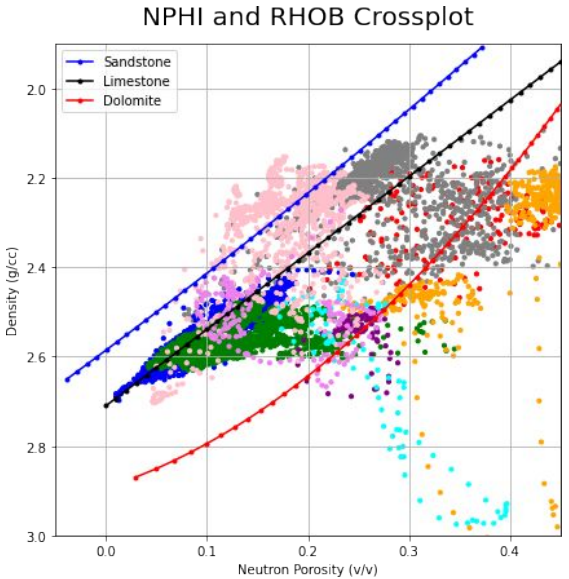


Taken from my project PyReSim in GitHub

Formation Evaluation with Python

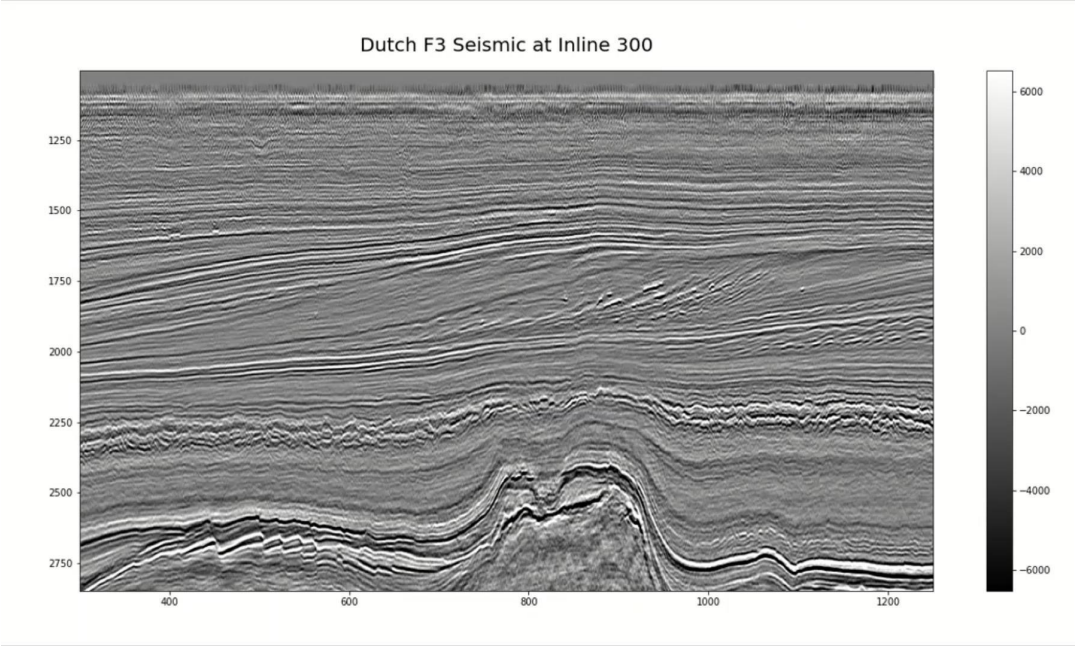
- Well-log visualization
- Exploratory data analysis (EDA) of well logs
- Petrophysical computations for total porosity, shale volume, water saturation, and permeability





Seismic Attributes with Python

- Compute various seismic attributes (RMS, envelope, phase envelope, instantaneous frequency, instantaneous phase, apparent polarity)
- Utilizing effective computation that does not consume memory space and RAM



Taken from my project computational-geophysics in GitHub

Potential Implementations in Engineering and Geosciences

- **Seismic post- and pre-stack inversion** – *geoscience*
- **Interpretation of bond logs (CBL, VDL, USIT)** – *completion + geoscience*
- **Kill sheets for kick tolerance calculation** – *drilling engineering*
- **Offset study and trajectory design** – *drilling engineering*
- **Wellbore geomechanics (borehole instabilities using Kirsch modeling)** – *drilling engineering + geoscience*
- **Mud weight design considering the mud window** – *drilling engineering + geoscience*
- **Geosteering (real-time acquisition of formation logging)** – *drilling engineering + geoscience*
- **Drill bit selection** – *drilling engineering + machine learning*
- **Nodal analysis** – *production engineering*
- **Selection of artificial lift system** – *production engineering (completion)*
- **Frac job** – *production engineering (completion)*
- **Calculations in formation stimulation and acidizing** – *production engineering*
- **Never ending list !!!**

GETTING STARTED WITH PYTHON

SEQUENCES IN PYTHON

Sequences

Sequence operations

Lists

Tuples

Strings

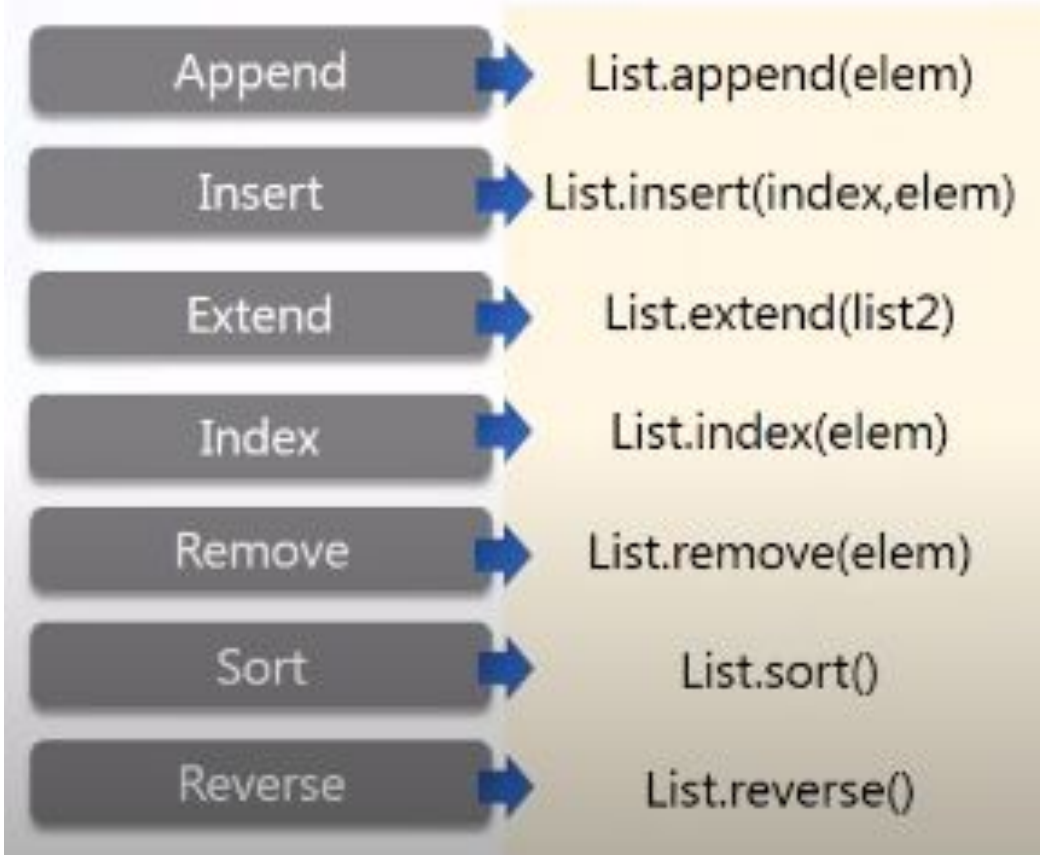
Sets

Dictionaries

LISTS

A list is a mutable ordered sequence of elements

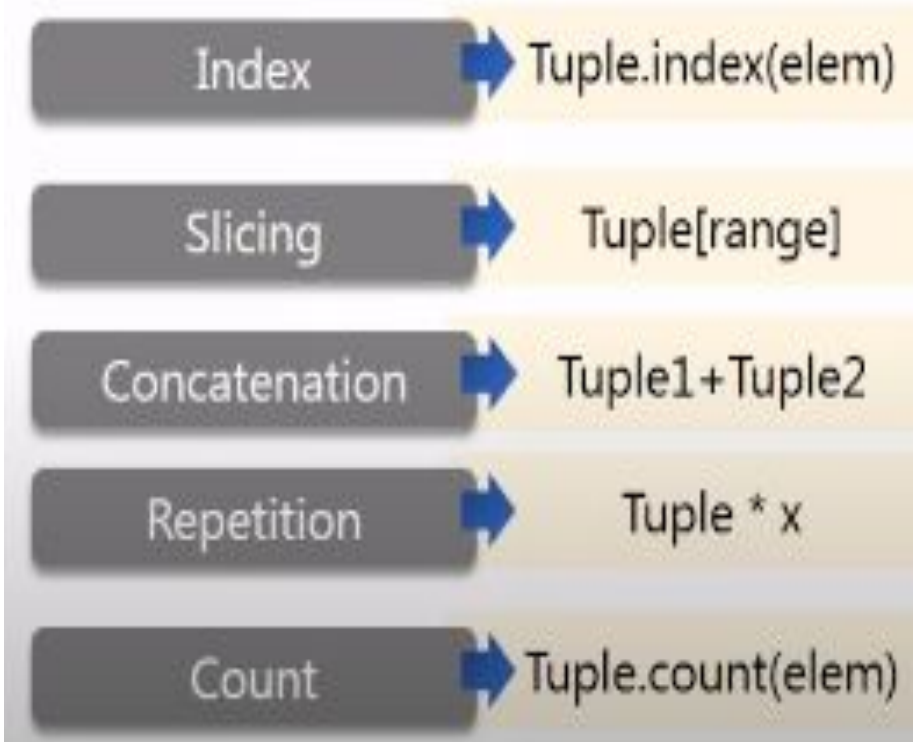
A = [x, y]



TUPLES

Tuples are immutable ordered sequence of elements

A = (x, y)



STRINGS

A sequence of characters. They are immutable.

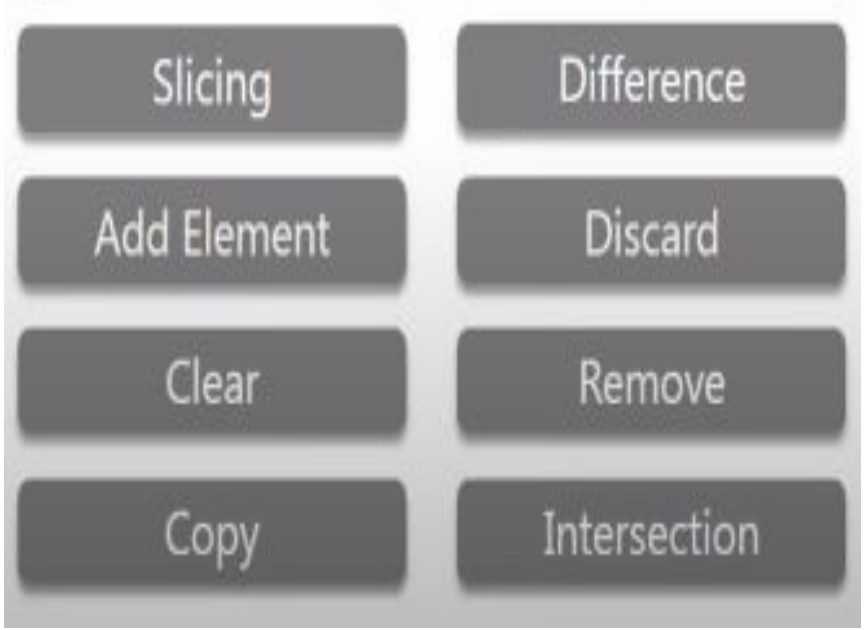
a = “abc”

SETS

A set is a collection that is unordered and unindexed

A = {a, b}

set(X)

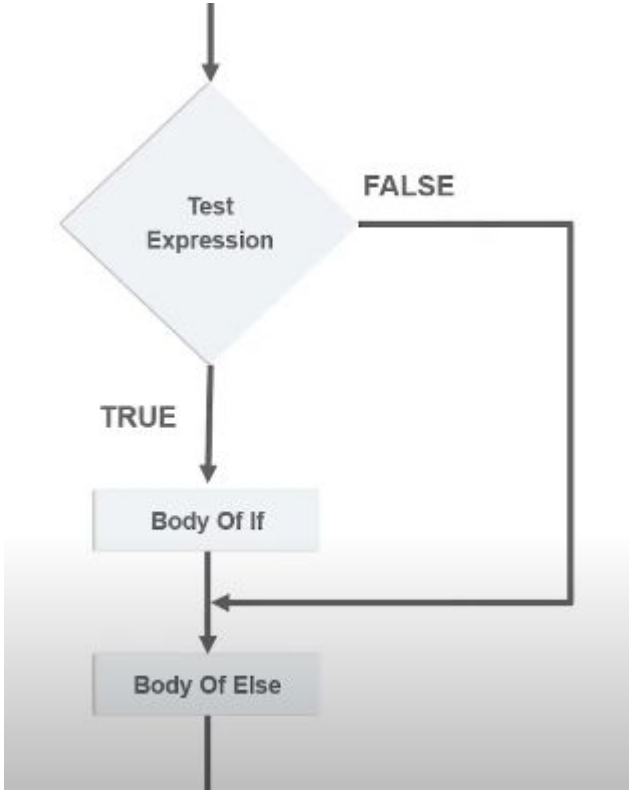


DICTIONARIES

A dictionary is an unordered sequence which is changeable and indexed.

CONDITIONAL STATEMENTS

If Statement



Equals to: x == y

Not Equals to: x != y

Less than: x < y

Less than or equal to: x<= y

More than: x > y

More than or equal to: x >= y

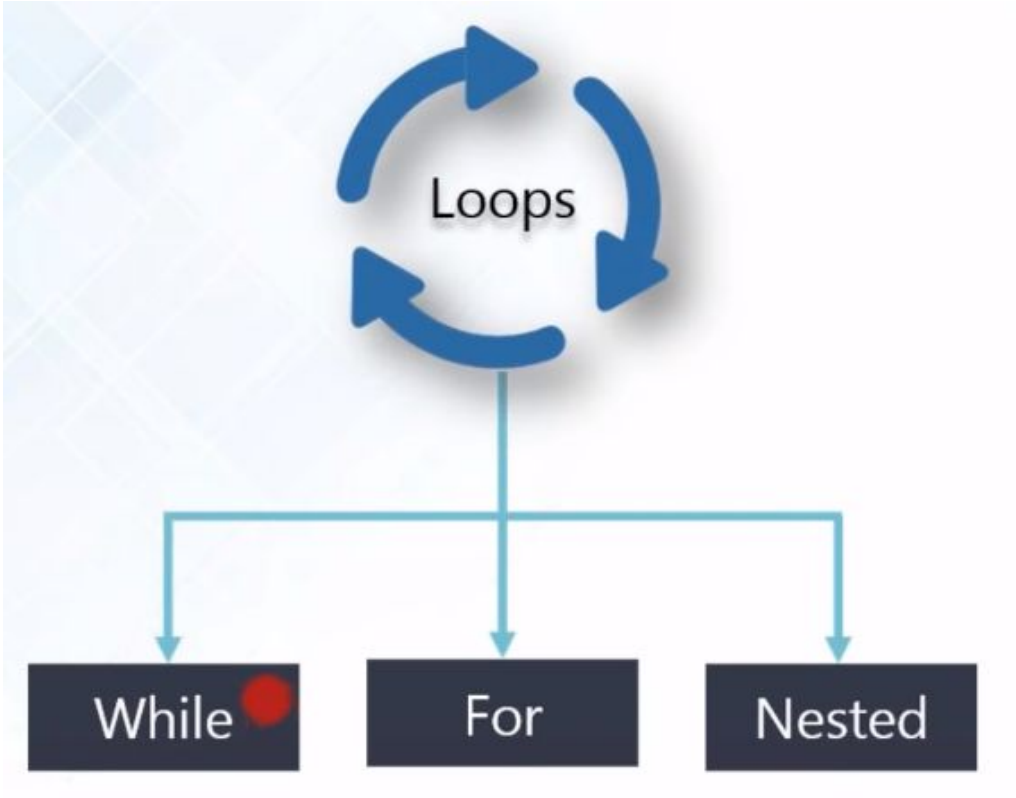
LOOPS

Loops

While loops

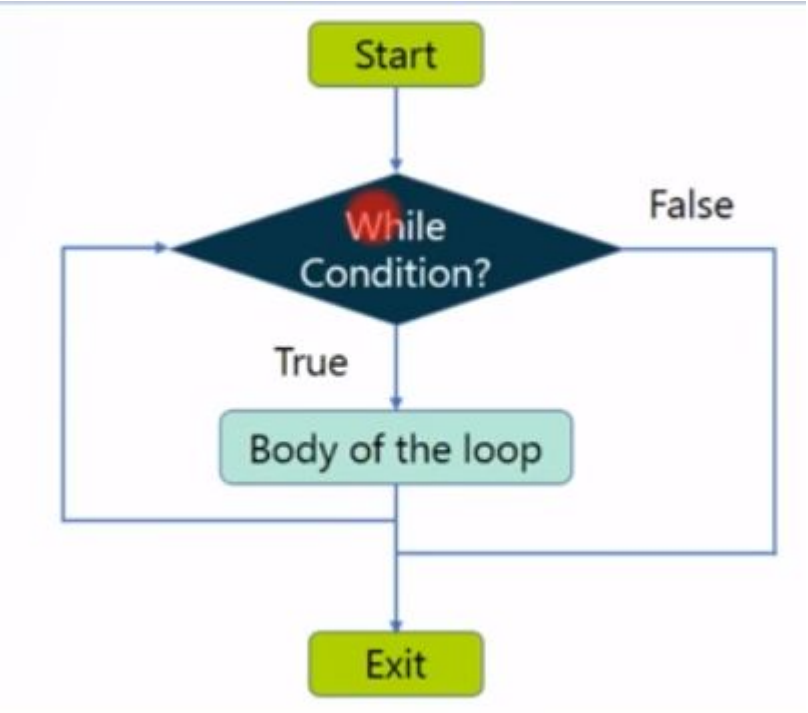
For loops

Nested loops



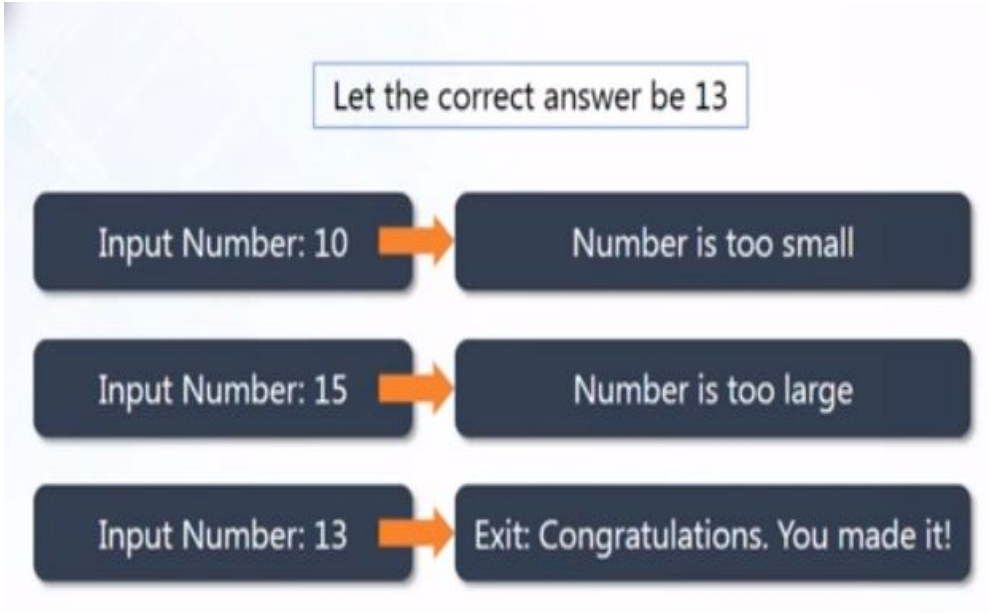
While loop

Used to iterate over a block of code as long as the conditions holds true



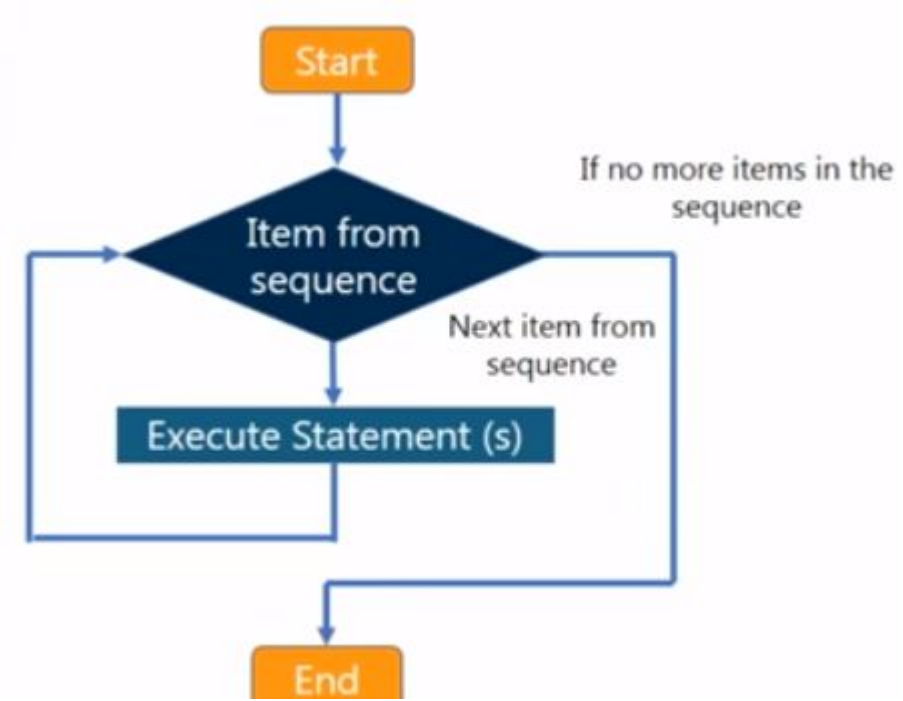
While loop

```
i = 1

While i < 6:

        print(i)

        i += 1
```



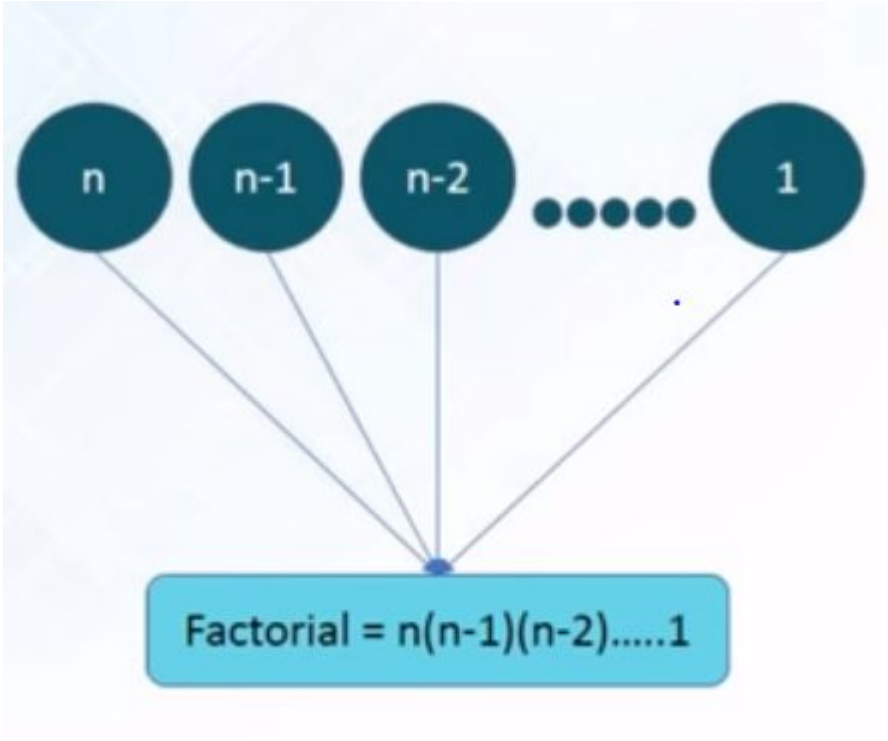
For loop

Used for iterating over a sequence



For loops

```
for x in y:

    print(x)
```



Nested loops

While (i < 100):

    j = 2

    while (j <= (i/j)):

Syntax:

```
for iterating_var in sequence:
    for iterating_var in sequence:
        statements
    statements
```

Syntax:

```
while expression:
    while expression:
        statements
    statements
```

Functions

Sme built in functions

abs()

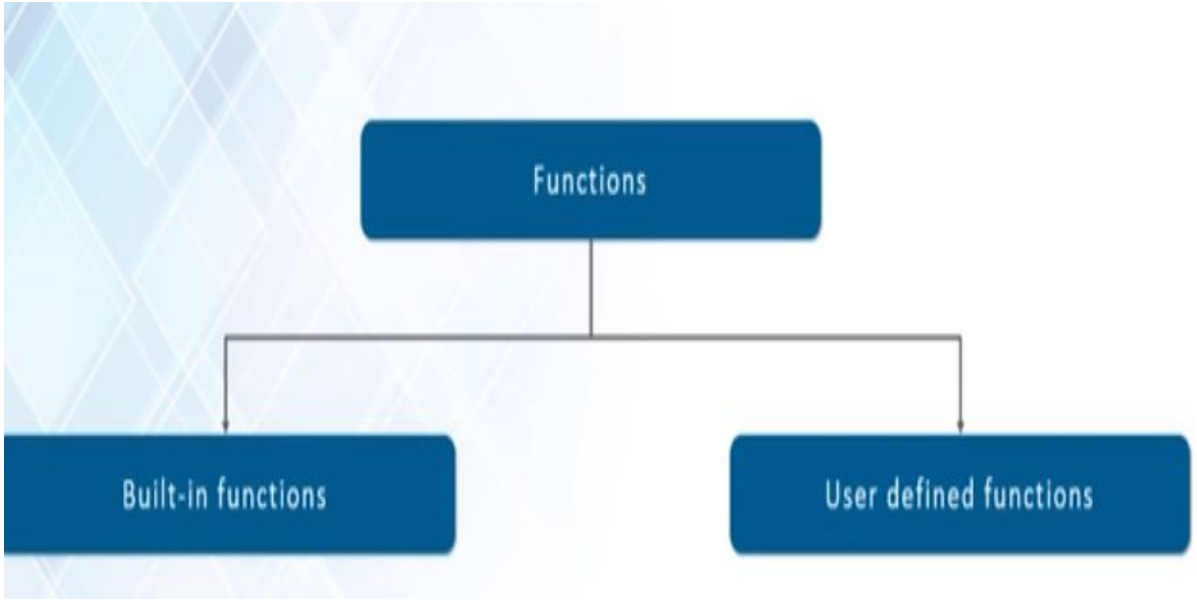
all()

ascii()

bool()

enumerate()

format()(



Functions

getattr()

id()

len()

map()

min()

pow()

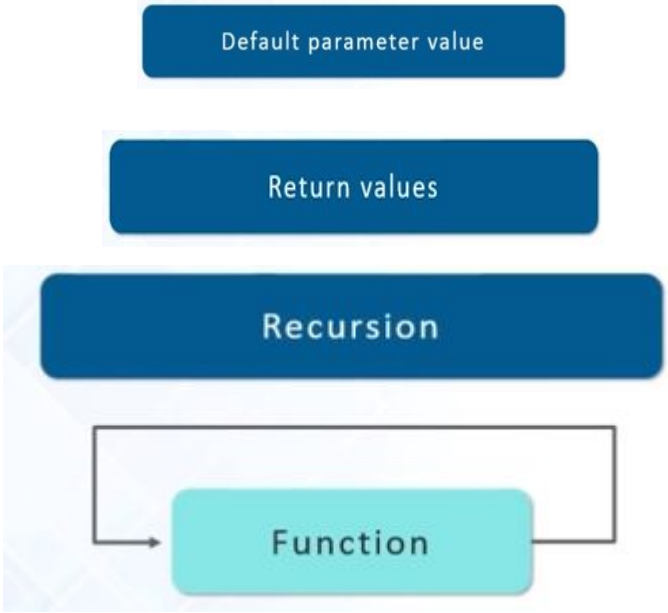
print()

setattr()

sorted()

Lambda functions

Functions



Creating a function

```
def my_function():
  print("Hello from a function")
```

Calling a function

```
def my_function():
  print("Hello from a function")

my_function()
```

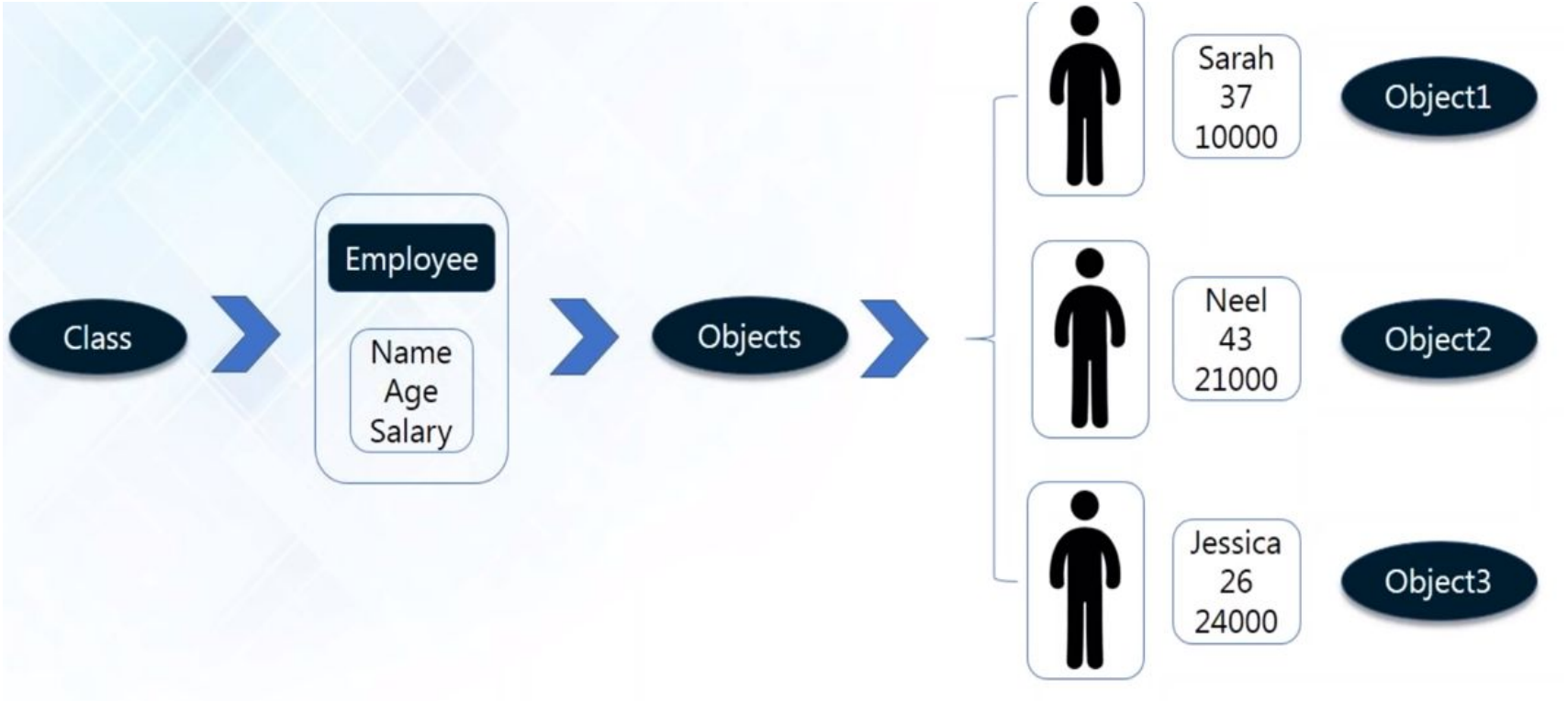
Classes

Classes and objects

Inheritance

Abstract classes

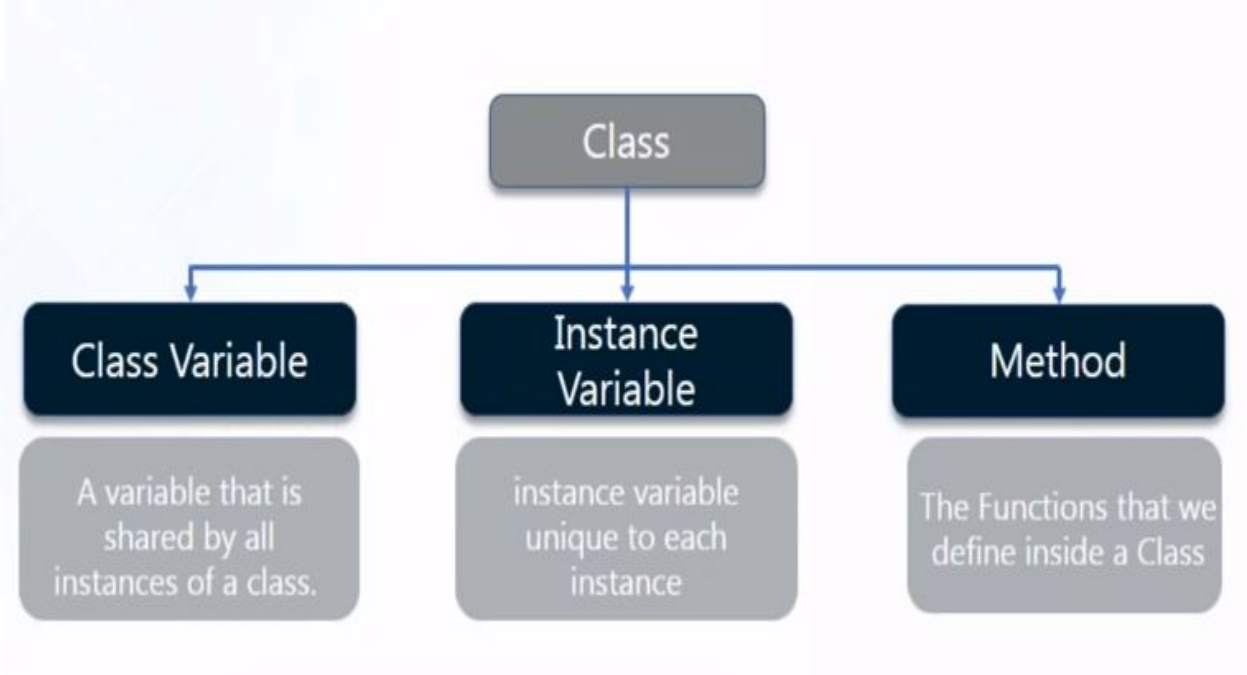
Classes


Class
Employee
Name
Age
Salary
Objects
Sarah
37
10000
Object1
Neel
43
21000
Object2
Jessica
26
24000
Object3

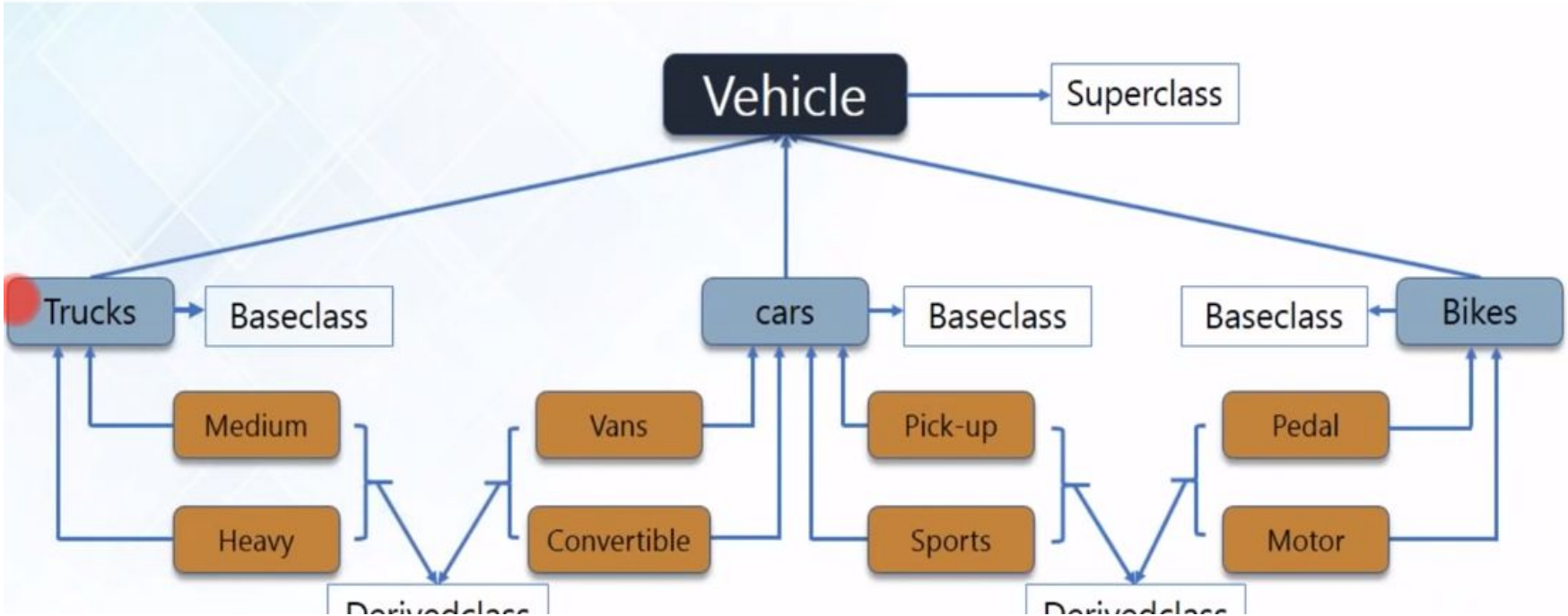
Elements in a Class

Syntax:

```
class Class_Name:
    statement-1
    .
    .
    .
    statement-N
```



Class Inheritance



Thankyou