

Session1:

- Introduction to Computer
- Introduction to Programming Language
- Orange Juice Analogy
- Compilation Flows -> C++ and Java
- Interpretation Flow -> Python
- A very Simple Program in Python (Execute on Terminal)
- Installation - Python, Visual Studio Code
- JupyterLabs, Google CoLab (Online env)
- On the OS (Application SW) vs In the OS (System SW) -> Interview Que
- Everything in Wolrd in MATEHMATICAL

Session2:

- What is main ?

MyApp.cpp

```
#include<iostream>
int main(){
    cout<<"Hello";
    return 0;
}
```

MyApp.java

```
class MyApp{
    public static void main(){
        System.out.println("Hello");
    }
}
```

General Definition -> main is that part of program, from where it begins

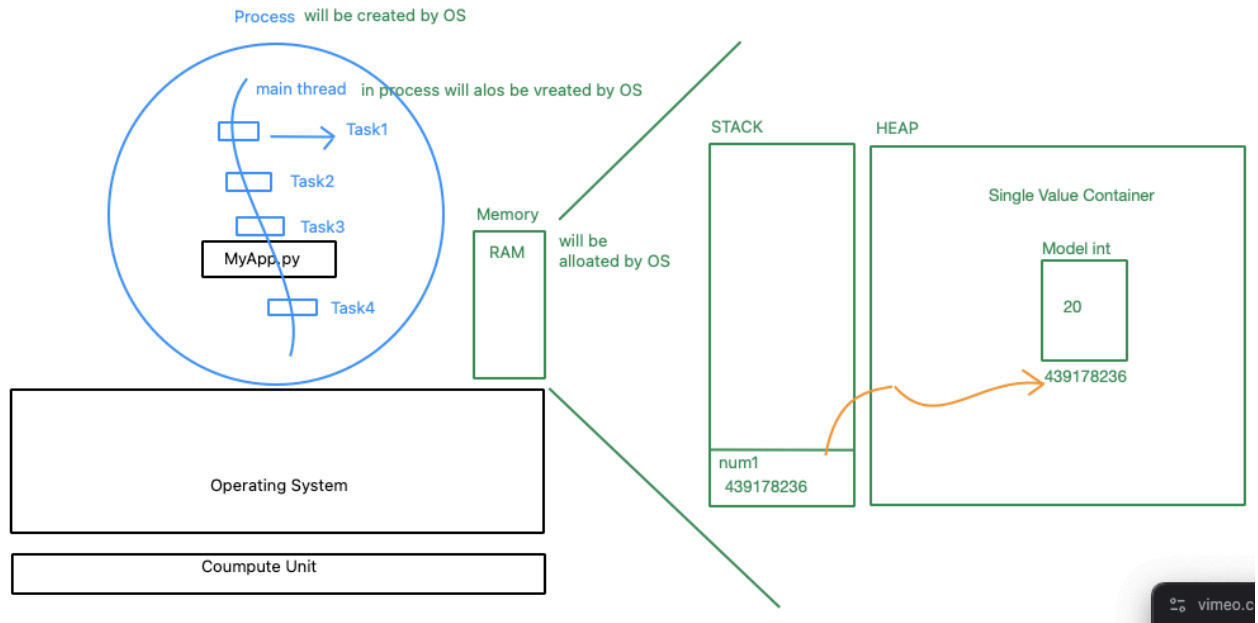
MyApp.py

```
print('hello')
```

In Python, by default you can code without any main

- What is a Process ?
A running app is known as a process
A program in execution is a process

A process has a thread, it has memory associated it
 The very first thread in a process is known as main thread



A simple program in python:

num1 = 10	-> Instruction1/Task1/Job1
num2 = 20	-> Instruction2/Task2/Job2
result = num1 + num2	-> Instruction3/Task3/Job3
print(result)	-> Instruction4/Task4/Job4

In a program, we can have below kind of instructions:

1. Write Statement
2. Read Statement
3. Update Statement
4. Delete Statement
5. Computation Statement, which can result in any of above 4

Analogy: Pendrive

- MODEL VIEW CONTROLLER Architecture
- MVC Architecture

Model -> Deals with Data (Data can be containerised) -> Space Complexity

Data Structure

A container holds data, just the way we put sugar in a container in the kitchen

View - represents Interfaces

Controller - represents logic

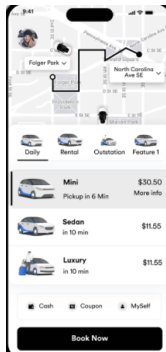
Uber Case Study to understand MVC Architecture

Model

- source location
- destination location
- type of cab

Purpose: To Store Data

View -> Interface



Purpose-> to show data from MODEL and to capture data from user into MODEL

Controller -> Logic (Maths/Reasoning/Logical SkillSet) -> Time Complexity

Algorithm

- > Search the shortest distance with least traffic
- > Closest Driver, also second closest if first cancels
- > Least Fare
- > Book Sedan at price of Mini

Purpose -> Get Data from MODEL -> Do Computation -> Show on UI

OR Capture from UI -> Do Computation -> Put in Model

- Create, Update, Delete and read in a program
- TIME is Top Notch Thing
Any innovation in world aims to solve a problem, which is 100 percent linked with TIME

Code for the Day:

```
# Create Statement
# MODEL: Container -> Single Value
num1 = 10

# num1: is a reference variable created in STACK of RAM
# 10 gets stored in a Container of type int in HEAP of RAM

# Read Statement
print('num1 is:', num1)
print('type of num1 is:', type(num1))
print('num1 hashcode is:', id(num1))

# Update Statement
num1 = 20

# Read Statement -> VIEW
print('num1 now is:', num1)
print('type of num1 now is:', type(num1))
print('num1 hashcode now is:', id(num1))

# Delete Statement
# Explicit (del statement) or Implicit (Automatic)
del num1

print('num1 now is:', num1)
print('type of num1 now is:', type(num1))
print('num1 hashcode now is:', id(num1))
```

Session3:

MODEL VIEW CONTROLLER

Model -> Deals with Data

View -> Represents Interface

Controller -> Logical Reasoning

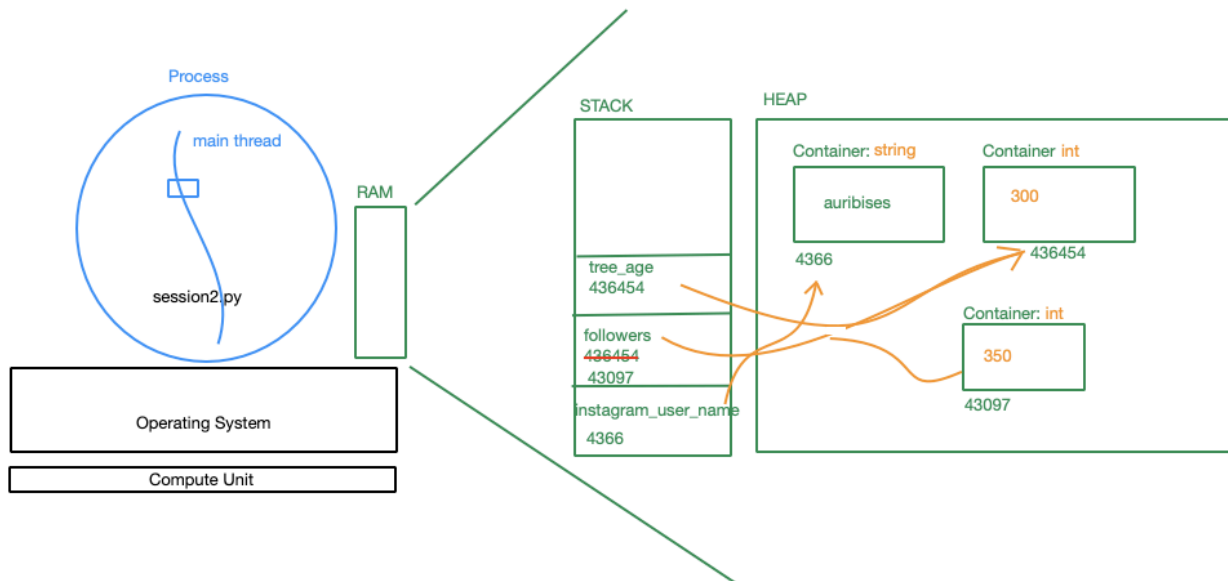
Data Structure -> Space Complexity

User Interface -> User Experience (UX)

Algorithm -> Time Complexity

MODEL -> Storage Container -> Data Structure

- Single Value Container
- Multi Value Container
 - Homogeneous MVC
 - Hetrogeneous MVC



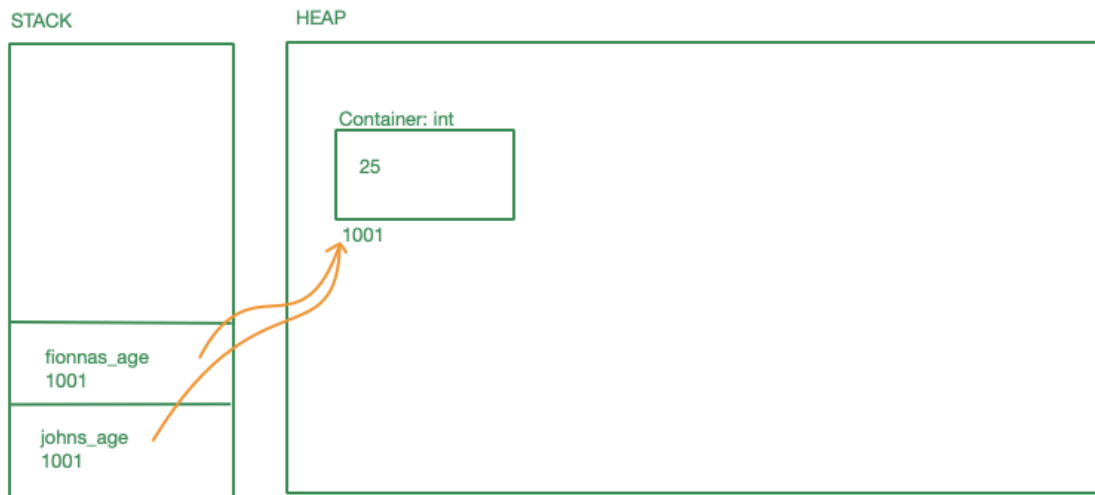
Functions

$$f(x) = 2x + 1$$

$$f(x,y) = 2x + y + 1$$

$x \rightarrow 1$, result $\rightarrow 3$

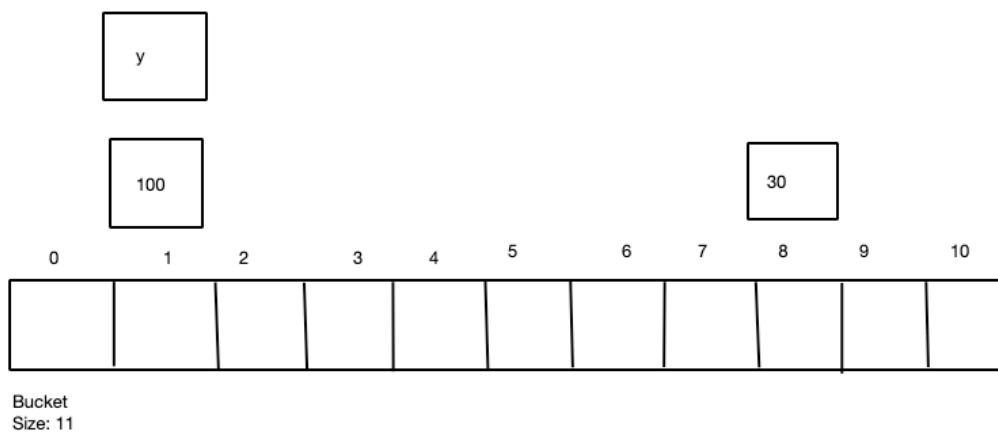
$x \rightarrow 1$ and $y \rightarrow 2$, result $\rightarrow 5$



Hashing -> Logical Way how to store data (Algorithm)
 HashCode is an outcome of Hashing

Hashing Function
 Bucket Size: 11

$h(X) = X \% \text{size}$
 $X = 100$
 $h(100) = 1$ (hashcode)
 $h(30) = 8$ (hashcode)
 $h(200) = 2$



Multi Value Container: Tuple

