Introduction to OOPS

1. Why OOPS `Object Oriented Programming`

OOPS was developed because of limitations were discovered in earlier approaches to programming.

To appreciate what `OOPS`, Let's understand the limitations of earlier approach.

📝`Procedural/Functional/Structured Programming Limitations:`

1. `Poor Representation`: Does not model real world problem very well.
2. `Data Access Dependency`: If a new data is added, all the functions needs to be modified to access the Data.
3. `Global data`: it is accessible to all the functions.
4. `Code Ambiguity`: No clear boundaries and well definition of code.
5. `No Modularity`: Functional programs can become monolithic and difficult to maintain as they grow in complexity.

2. Multiple Students Example - Messy Code

How will you model a program having 100s of students, and each students have their own properties, behaviours and something to hide?

🧑‍💻`Challenges in Modeling a Program with Hundreds of Students:`

1. `Individual Properties`: Each student possesses unique attributes such as name, age, and grade.
2. `Diverse Behaviors`: Students exhibit various behaviors like studying, attending classes, and submitting assignments.
3. `Privacy Concerns`: Certain information about students may need to be concealed or restricted.

To address these complexities, a structured approach like **Object-Oriented Programming** becomes crucial for `maintaining clarity`, `modularity`, and `ease of management` in the code.

3. What is OOP?

1. Programming is used to solve real-world problems, how can we model real-world systems with programming languages.

2. A Programming Style, involves dividing a program into pieces of objects that can communicate with each other.
3. Objects based coding style, in which each object (aka, real-world entity) has its own attributes and behaviour.
4. Fundamental Idea is to combine into single unit, both data and behaviour, that will promote Modularity.
5. OOP promotes modularity by encapsulating data and behavior within objects. This modular approach enhances code reusability and maintainability, as objects can be reused in different parts of the program.
6. OOP is `LIFE` (We'll understand this on-the-go)

4. Objects and Classes

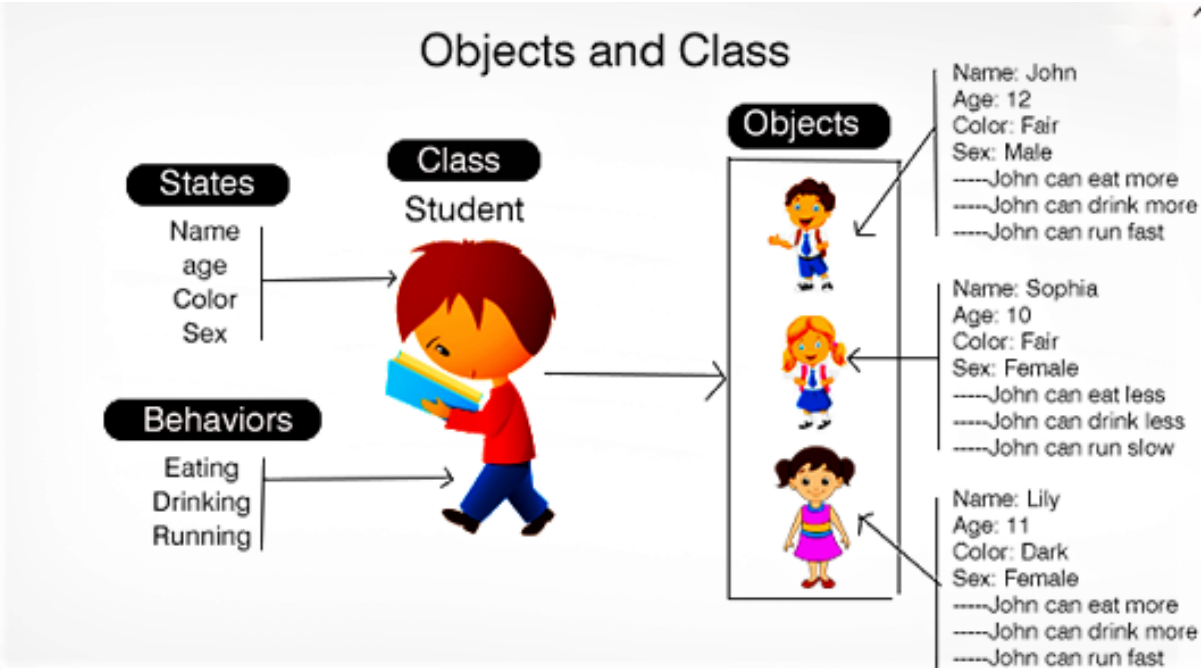
1. Real world `entities` like cars, person, students, building etc., they all have some `state` and `behaviour`.
2. For e.g., a Student named John, is an real world entity, in programming, he is an `object`.
3. What defines, how would an object look like?, there must be a Blueprint i.e., `Class`.
4. Hence, Object is an `instance of a Class`.

5. Attributes and Behaviour and Identity

1. Attributes are `state` / `properties` of an object.
2. Behaviour is `methods` / `functions` that an object can perform.
3. Identity is `unique name` of each instance of a class.

So…

1. Object Oriented Programming (OOP) is a programming paradigm focused on implementing real-world objects.
2. The identification of code objects similar to real-life objects and structuring code using classes and objects signifies the use of OOP principles.
3. Classes and objects serve as the fundamental building blocks of the OOP concept.
4. Major Object Oriented languages include C++, Java, and Javascript.



Characteristics of Object

A **State**
Represent the data of an object.

B **Behavior**
Represents the behavior of an object such as deposit, withdraw, etc.

C **Identity**
It is used internally by the JVM to identify each object uniquely.