

# POPS

PROCEDURE ORIENTED PROGRAMMING SYSTEMS



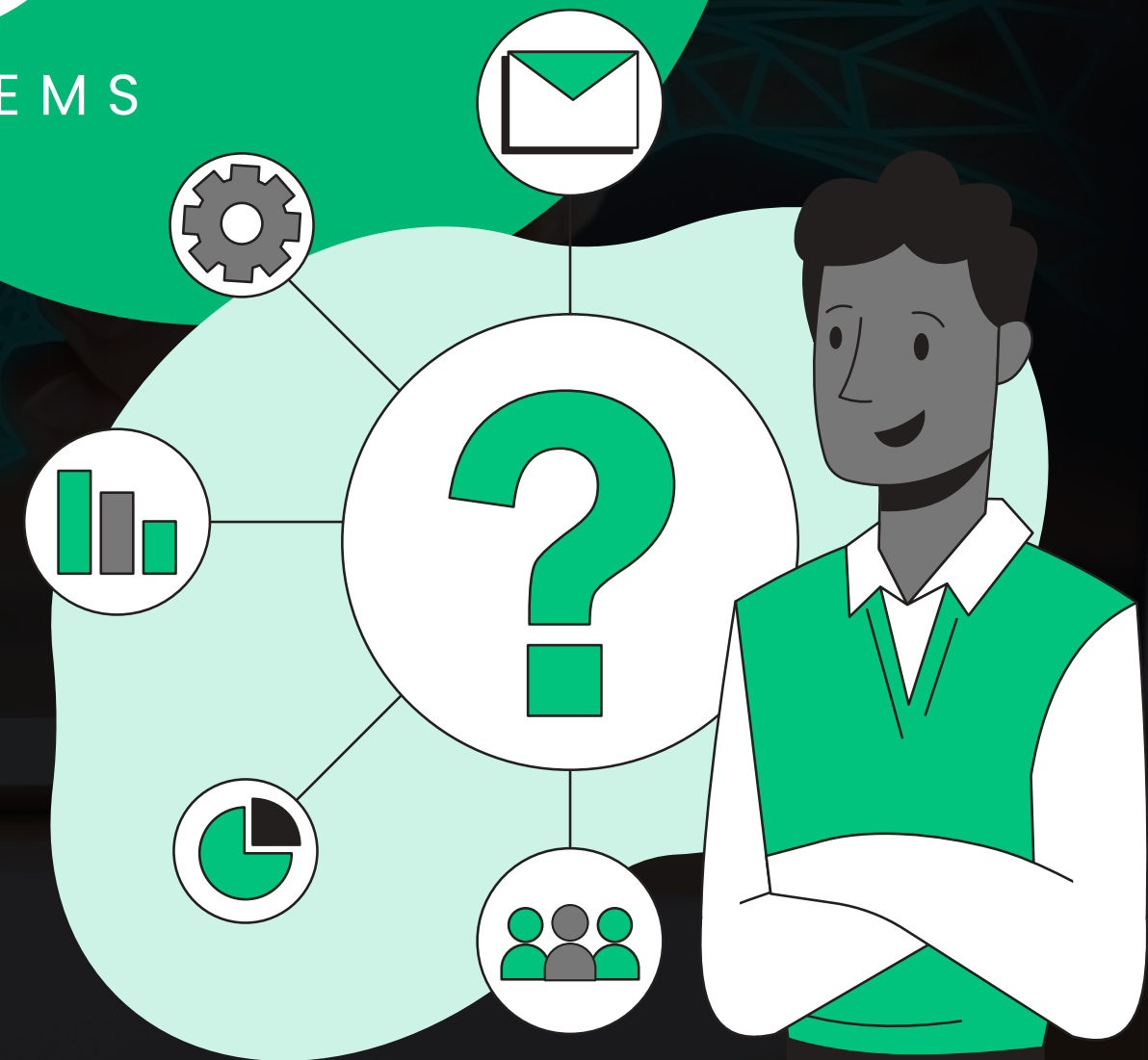


# Characterstics

1. Puts much importance on Things to be Done.
2. Large problems are divided into smaller programs known as functions.
3. Most of the functions share global data.
4. Data move openly around the system from function to function.
5. Functions transfer data from one form to another.

# OOPS

OBJECT ORIENTED PROGRAMMING SYSTEMS





# OOPS

Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects.



# Characteristics



1. Emphasis on data rather than procedure
2. Programs are divided into entities known as objects
3. Data Structures are designed such that they characterize objects
4. Functions that operate on data of an object are tied together in data structures
5. Data is hidden and cannot be accessed by external functions
6. Objects communicate with each other through functions



# Pillars of OOPS


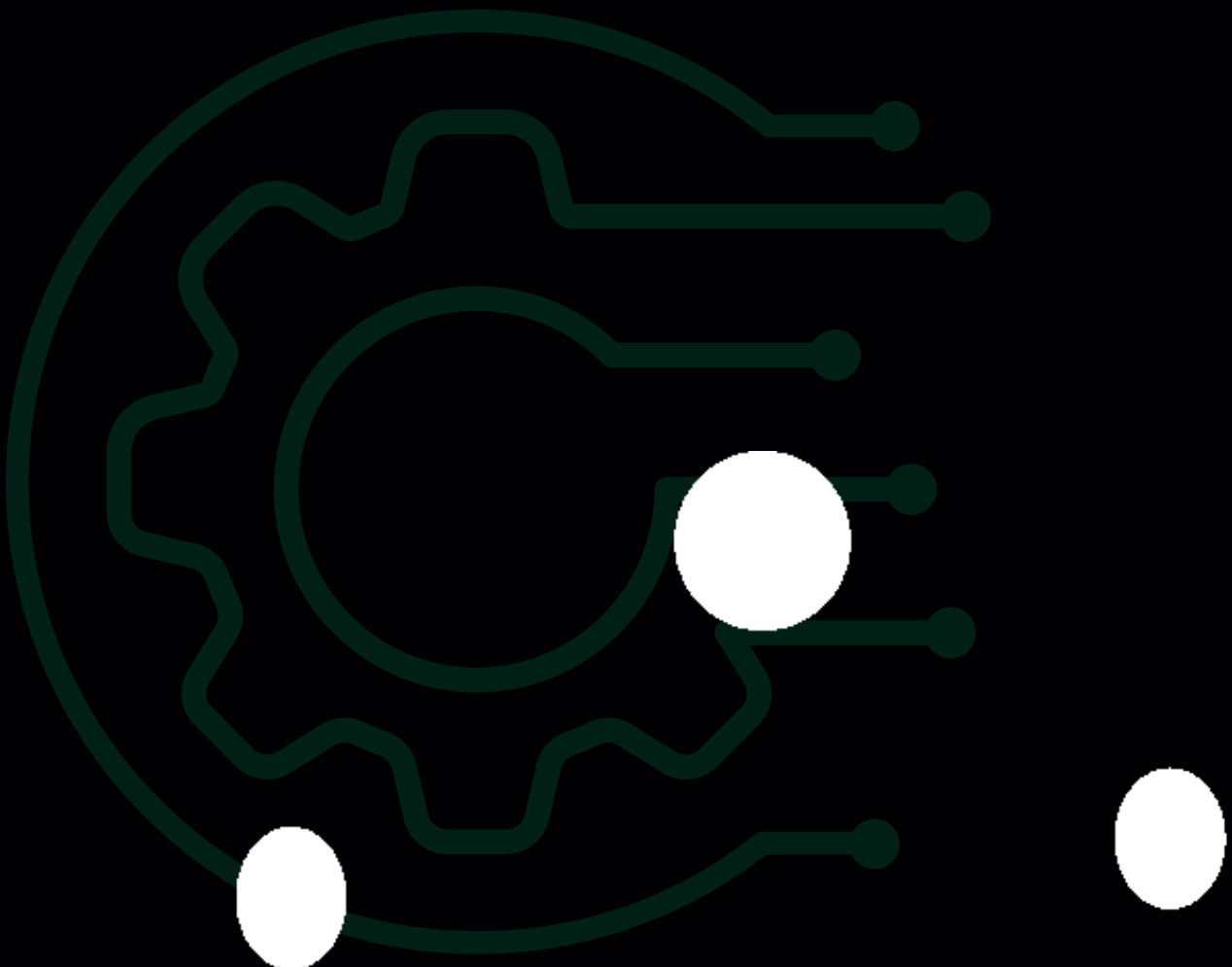


1. Abstraction
2. Encapsulation
3. Inheritance
4. Polymorphism

# CLASS



Class is a user-defined data type which defines its properties and its functions. Class is the only logical representation of the data. For example, Human being is a class. The body parts of a human being are its properties, and the actions performed by the body parts are known as functions.





# OBJECT

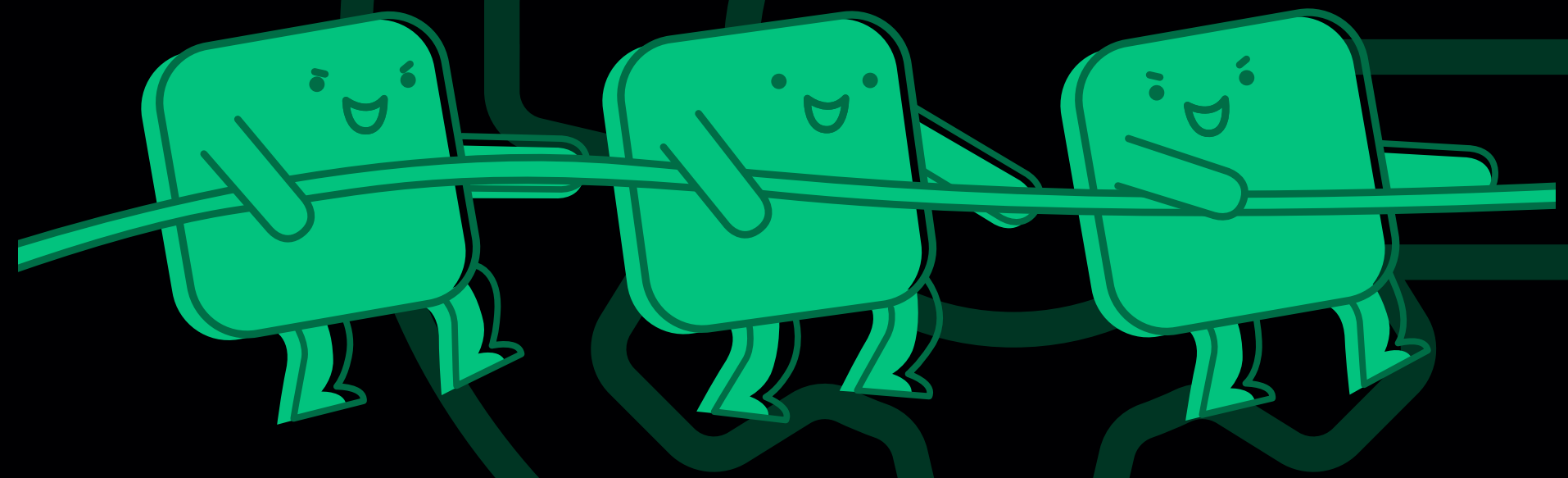
Object is a run-time entity. It is an instance of the class. An object can represent a person, place or any other item. An object can operate on both data members and member functions.





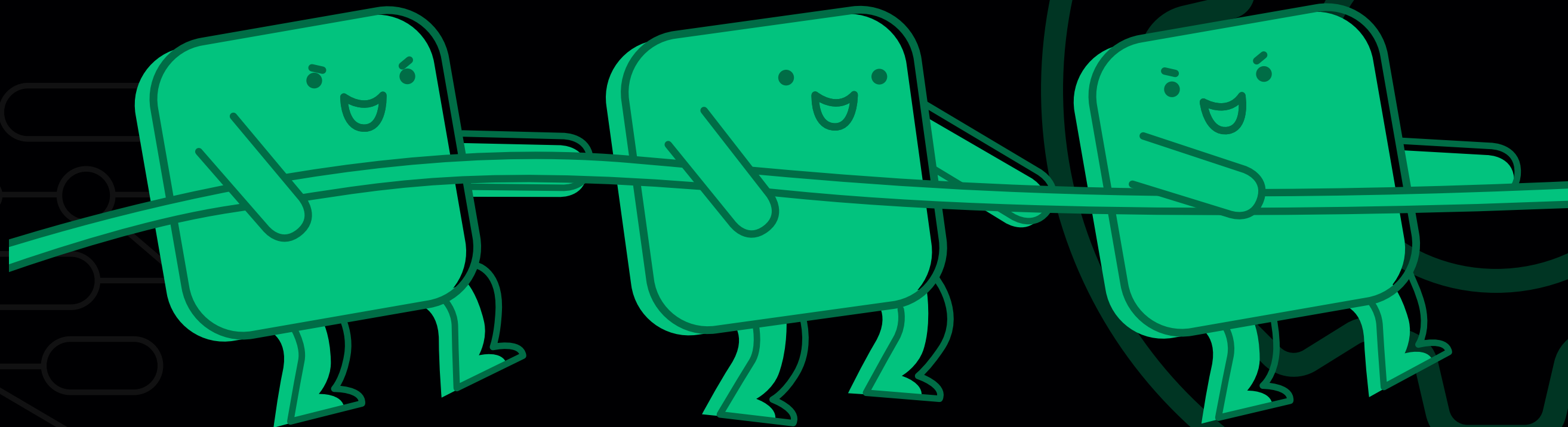
# Polymorphism

Polymorphism is the ability to present the same interface for differing underlying forms (data types). With polymorphism, each of these classes will have different underlying data. Precisely, Poly means 'many' and morphism means 'forms'.



# Types of Polymorphism

1. Compile Time Polymorphism (Static)
2. Runtime Polymorphism (Dynamic)

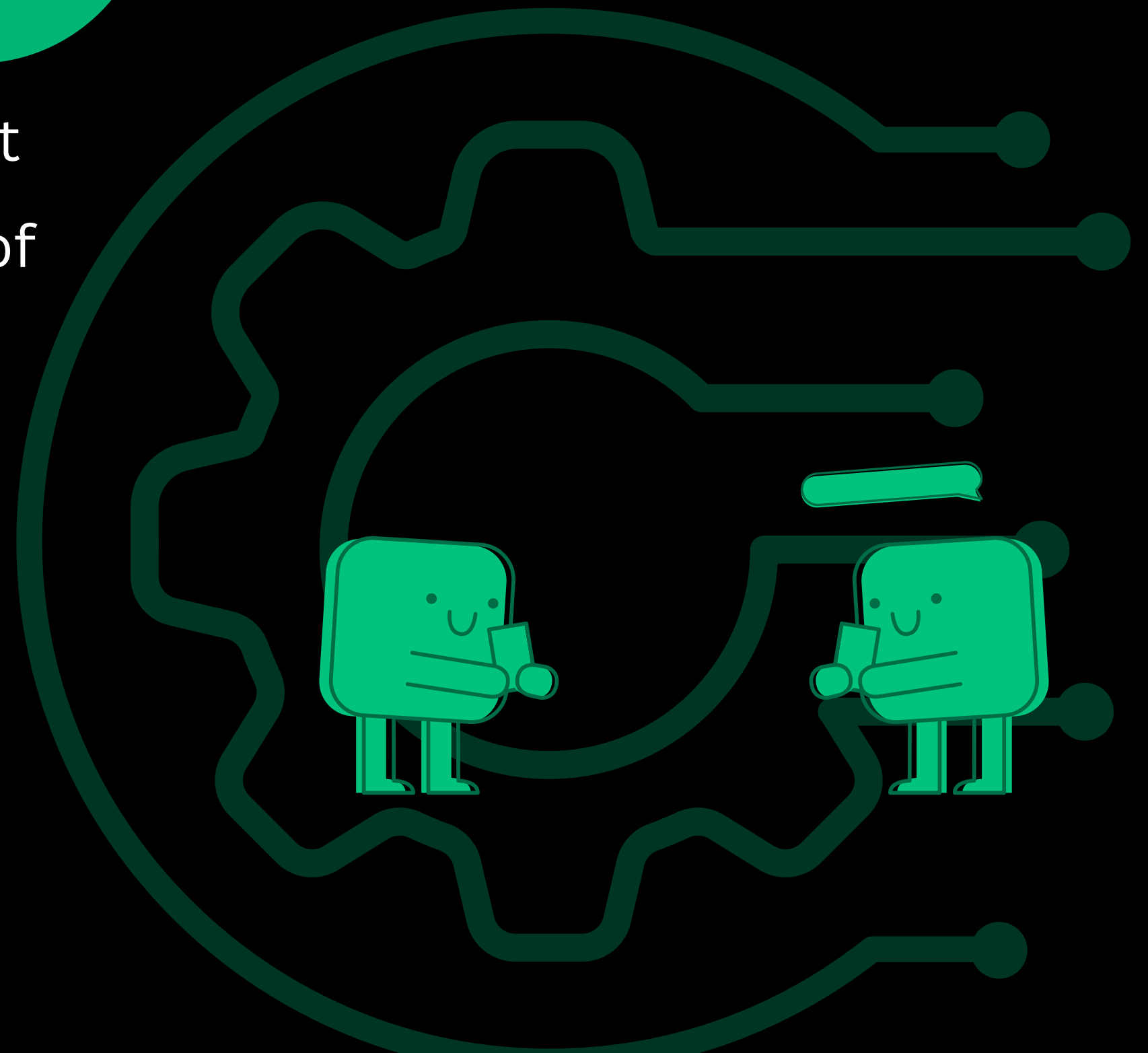




# Inheritance

Inheritance is a process in which one object acquires all the properties and behaviors of its parent object automatically. In such a way, you can reuse, extend or modify the attributes and behaviors which are defined in other classes.

In Java, the class which inherits the members of another class is called derived class and the class.

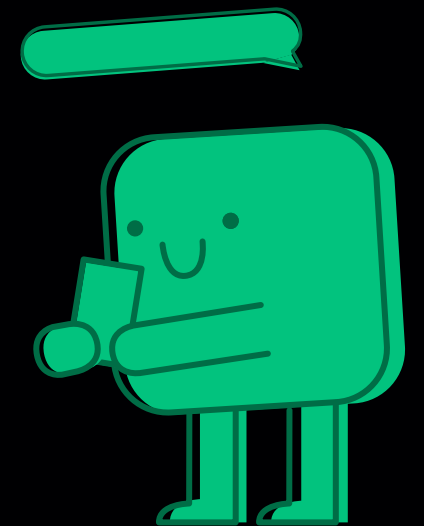
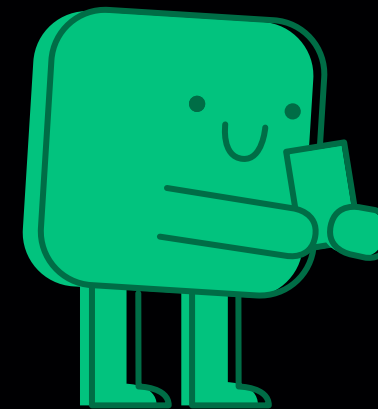


# Types of Inheritance

1. Single inheritance : When one class inherits another class, it is known as single level inheritance

2. Hierarchical inheritance : Hierarchical inheritance is defined as the process of deriving more than one class from a base class.

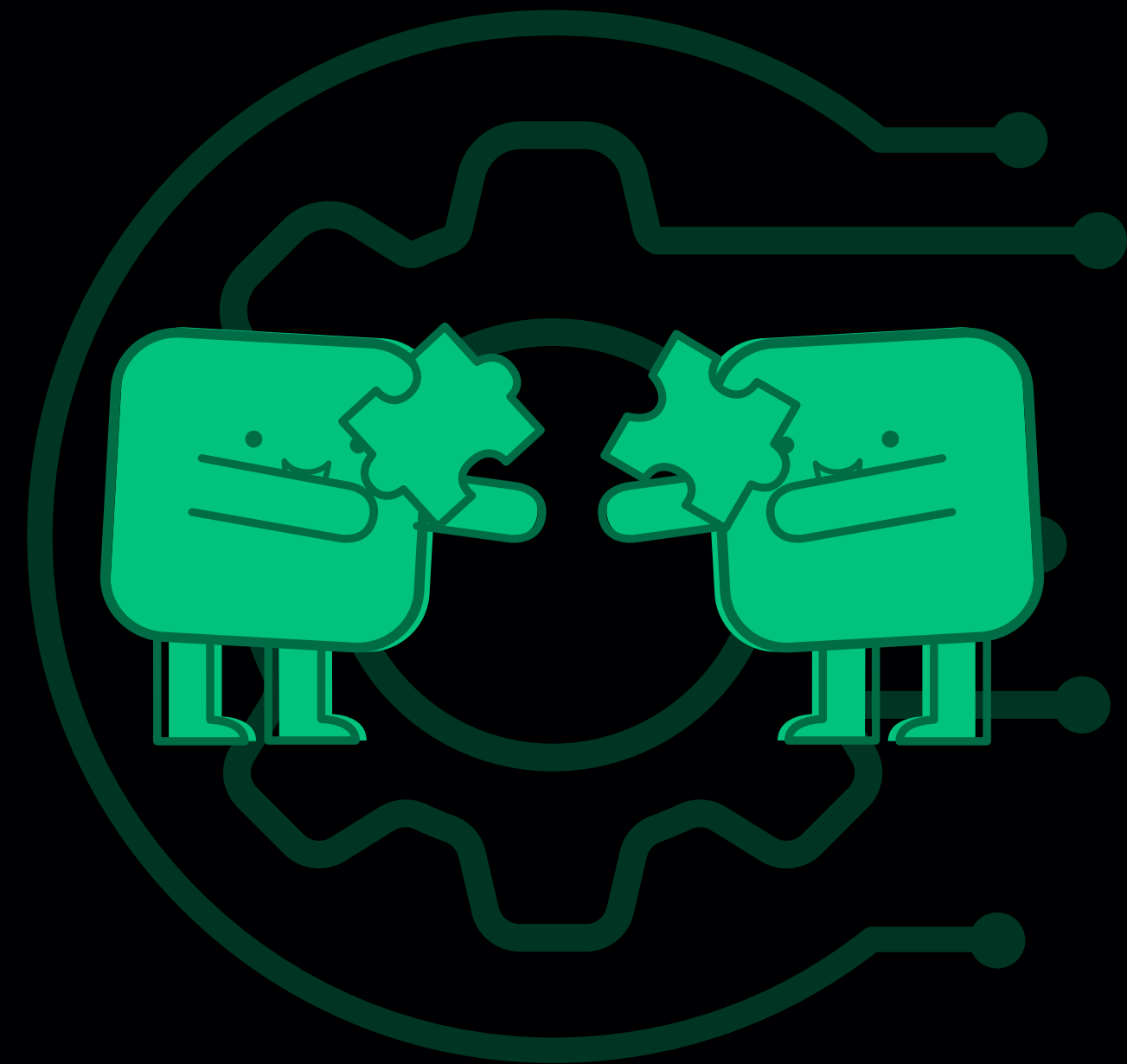
3. Multilevel inheritance : Multilevel inheritance is a process of deriving a class from another derived class.





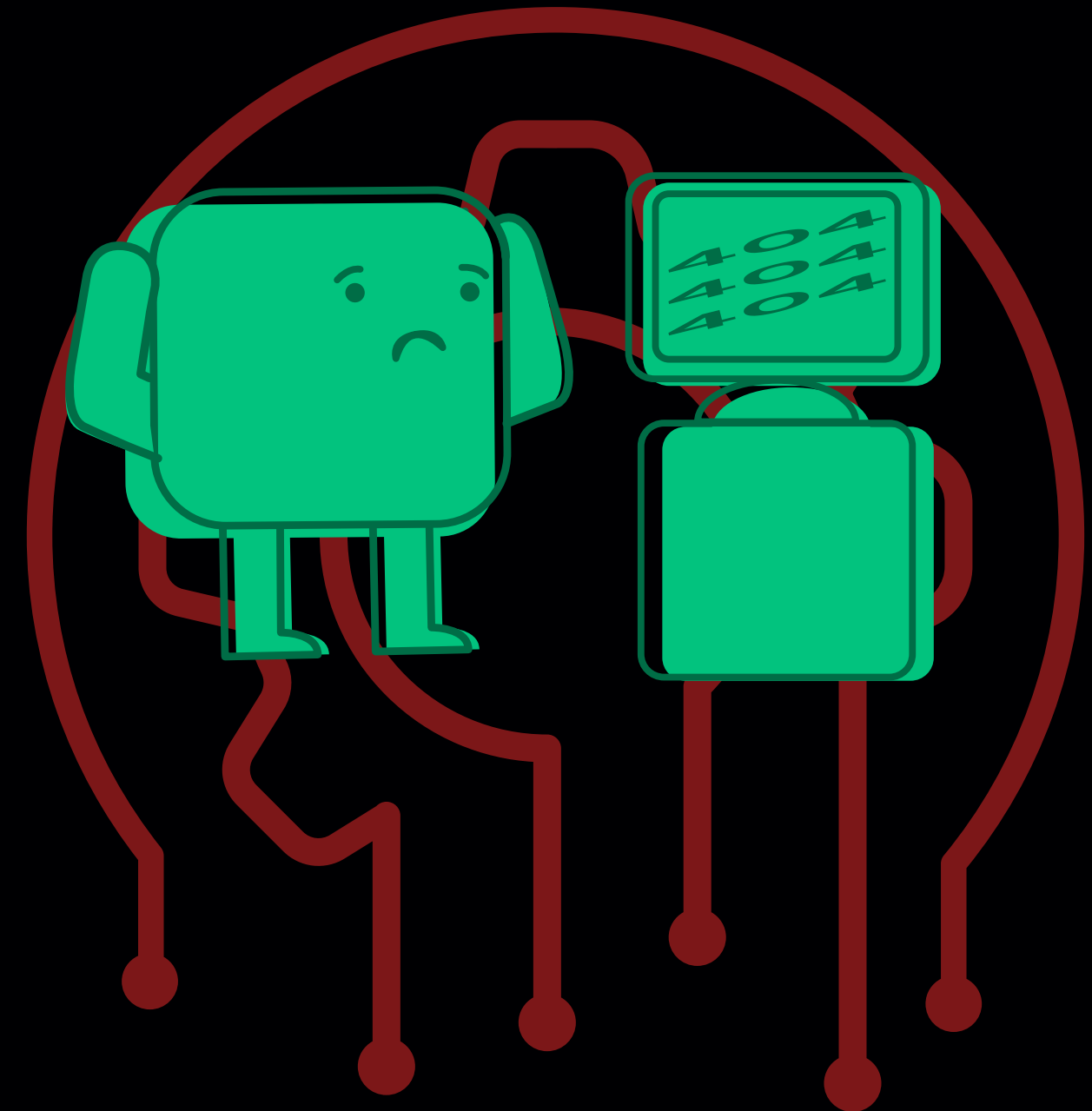
# Encapsulation

Encapsulation is the process of combining data and functions into a single unit called class. In Encapsulation, the data is not accessed directly; it is accessed through the functions present inside the class. In simpler words, attributes of the class are kept private and public getter and setter methods are provided to manipulate these attributes. Thus, encapsulation makes the concept of data hiding possible.



# Abstraction

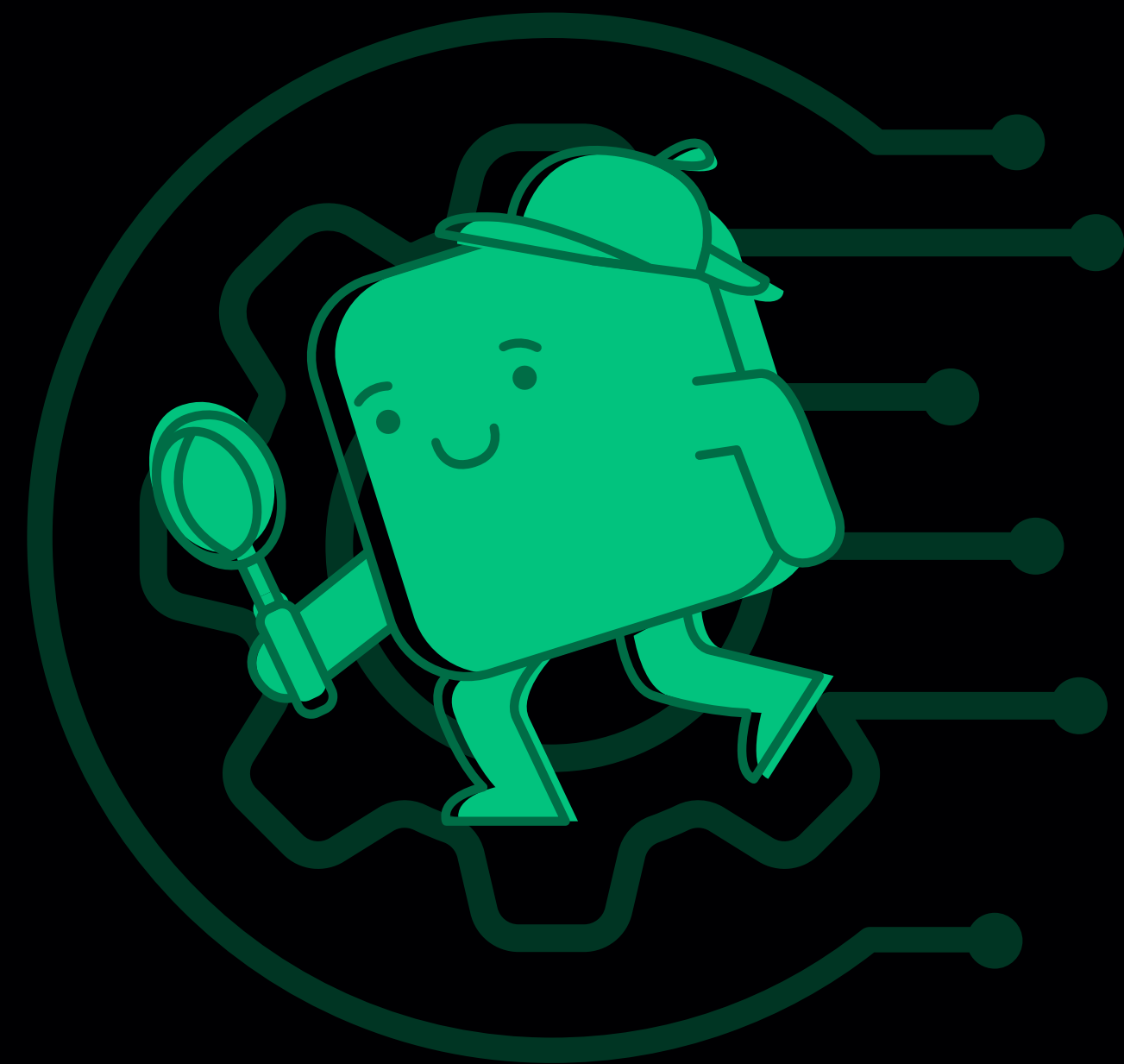
We try to obtain an abstract view, model or structure of a real life problem, and reduce its unnecessary details. With definition of properties of problems, including the data which are affected and the operations which are identified, the model abstracted from problems can be a standard solution to this type of problems.





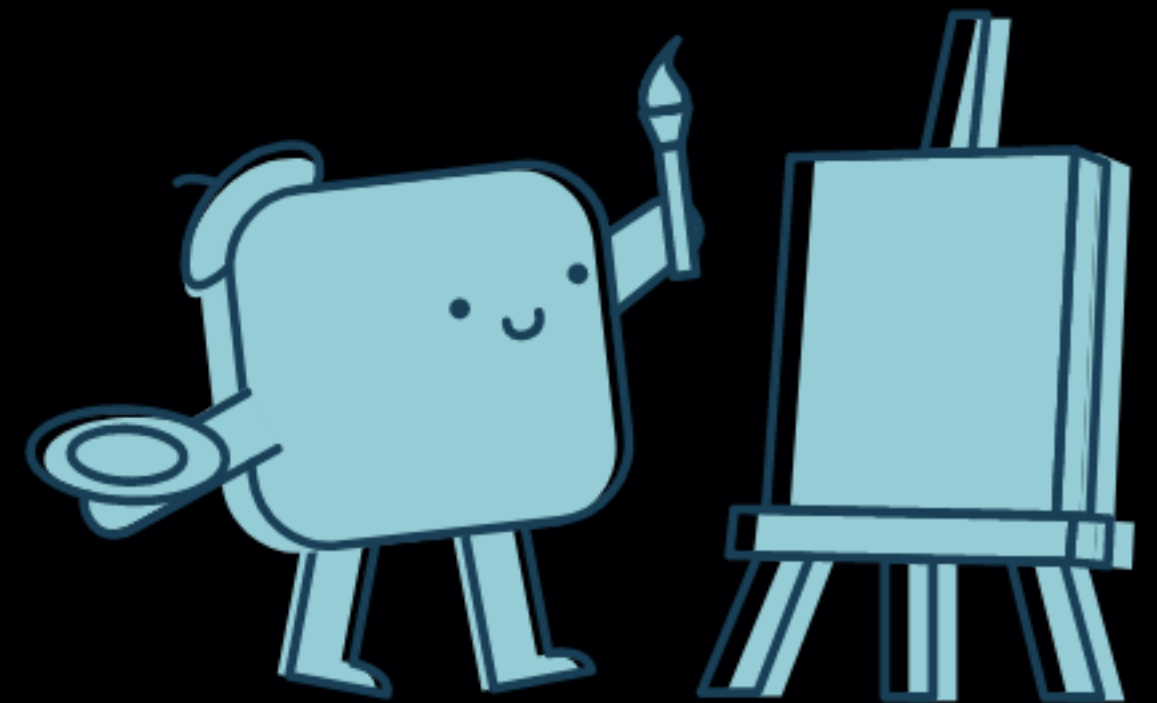
# Abstract class

- An abstract class must be declared with an abstract keyword.
- It can have abstract and non-abstract methods.
- It cannot be instantiated.
- It can have constructors and static methods also.
- It can have final methods which will force the subclass not to change the body of the method.



# Interfaces

- All the fields in interfaces are public, static and final by default.
- All methods are public & abstract by default.
- A class that implements an interface must implement all the methods declared in the interface.
- Interfaces support the functionality of multiple inheritance.





Presented by

Vikas Joshi

