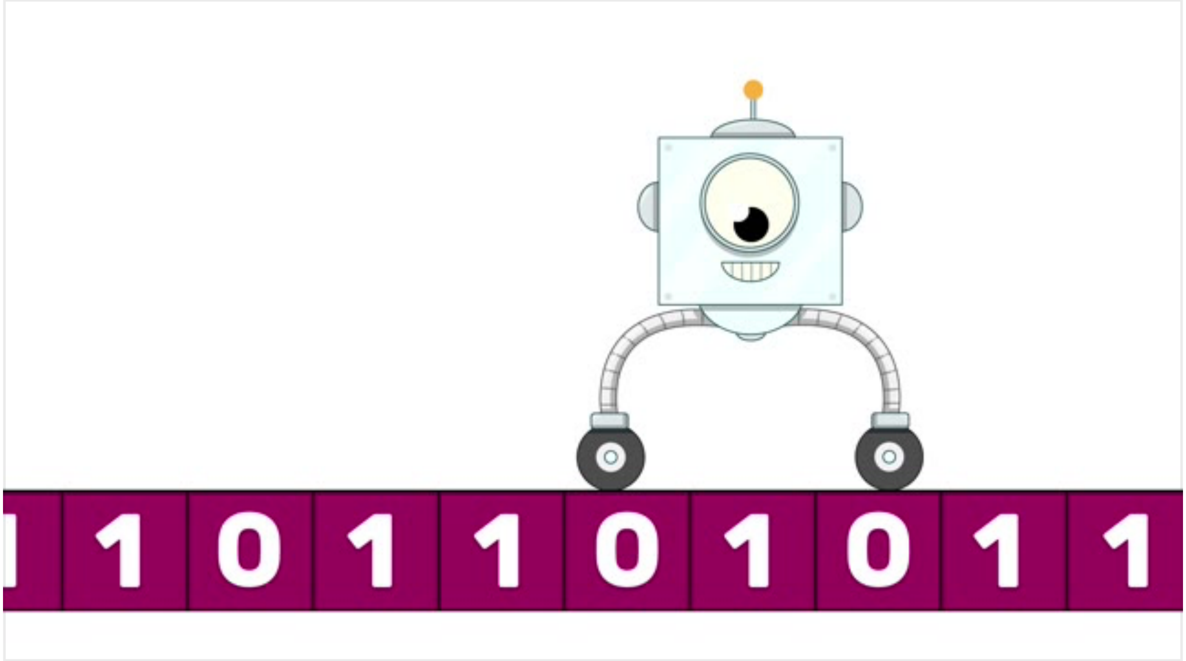
The Anatomy of Computer Program

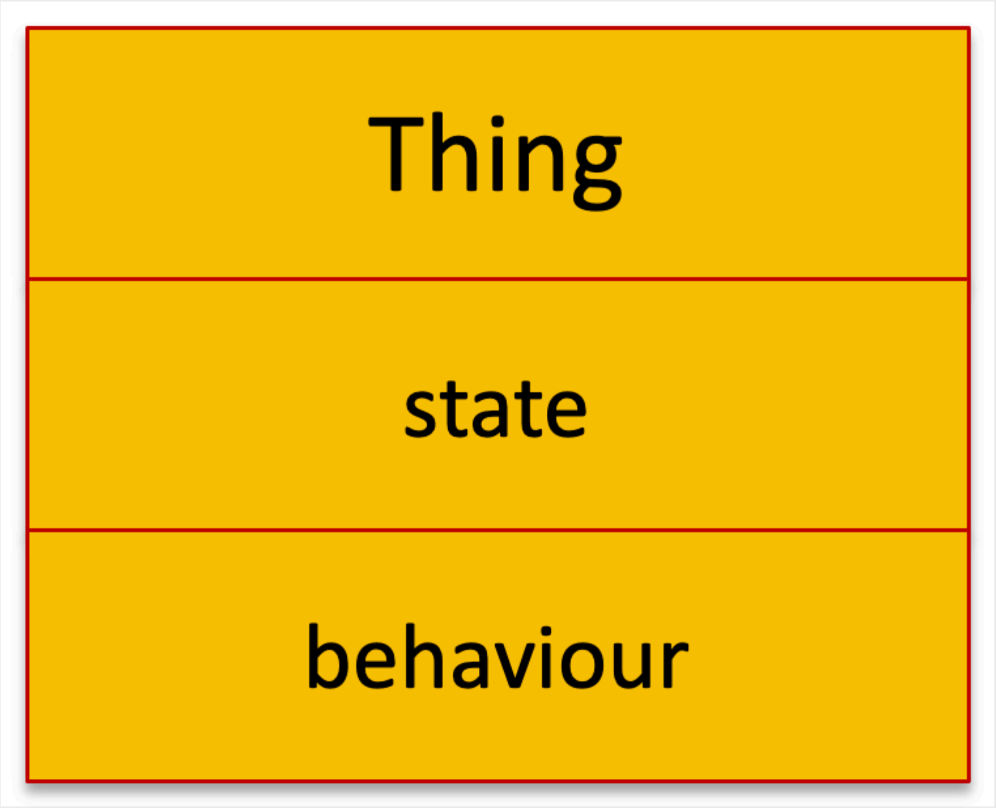


Anatomy is the science that studies the structure of a thing. An understanding of the essence of program structure is fundamental to design and build computer systems. Abbreviated to the fundamental, abstracted to the highest level, the anatomy of a computer program is the same as anatomy of anything in this universe:

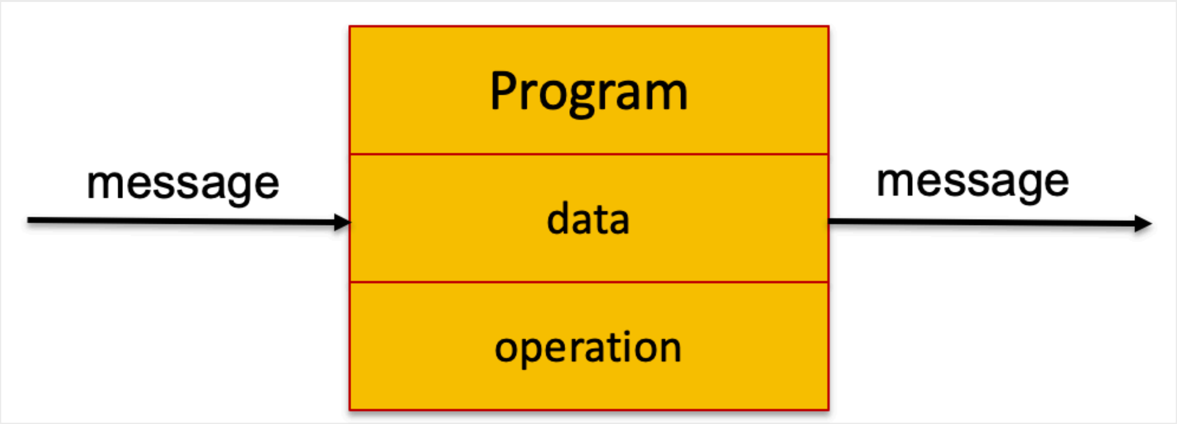
Thing = state + behaviour.
Program = data + operation.
Class = variables + methods.

Thing = state + behaviour

Any “thing” in the universe has state and behaviour. An atom has mass, and its electrons orbit its nucleus. A human being has a brain to think, eyes to see, ears to hear, and muscles to move. The universe has galaxies, which are moving away from us at increasing speed. State and behaviour are correlated: state affects behaviour, and behaviour changes state.

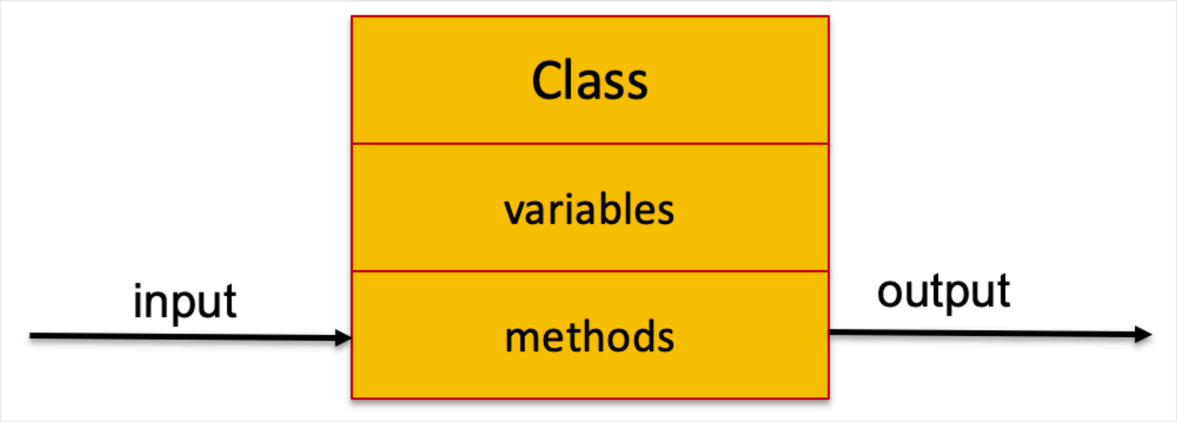


Program = data + operation
Programs also have state and behaviour. A program is conceived, observed, and understood by behaviour, which constantly changing its state. A program's behaviour is defined as the set of activities it can perform, described by algorithms, and the sum of these activities is called its protocol. A program's state is referred to as its data, and behaviour referred to as operation. Programs do not exist in isolation, they reach out and touch others, by sending messages to other programs, and operate as requested by the messages it receives. Programs interact with each other in a virtual universe in order to fulfil their purposes.

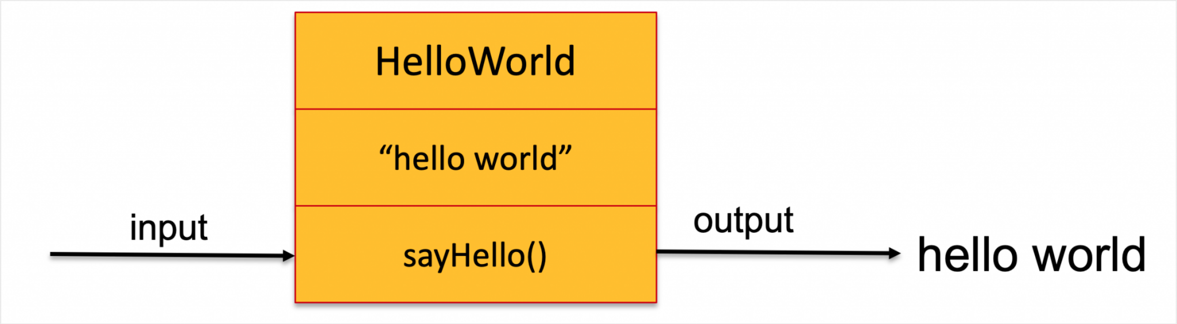


Class = variable + method
In Object-Oriented Programming (OOP) languages such as Java, the minimal program unit is called a Class, which is an encapsulation of state and behaviour. A class is an abstract data type, and its instances are called objects. In a class,

its state is represented by variables, and its behaviour is represented by methods. All instances of a class have the same behaviour as defined by class methods, while each instance of a class (object) has its own state with different values stored in their variables.



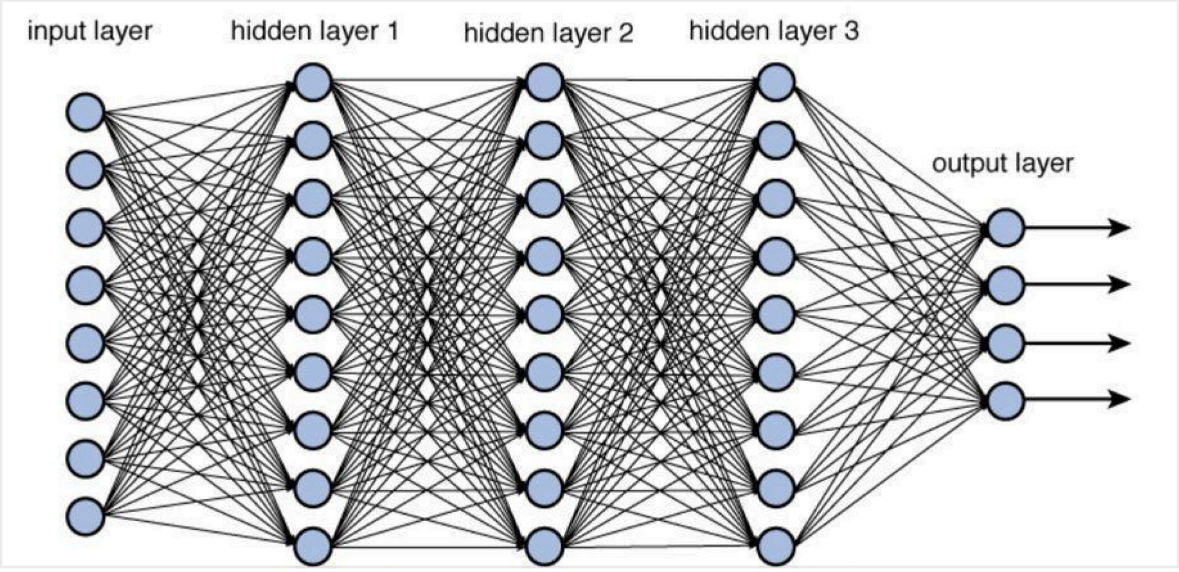
And here is the HelloWorld class we coded in our Hello World post. Its state is the string message “hello world”, and the behaviour is its method sayHello(). There is no input required, and the output is the terminal printout of “hello world”.



Animistic Program

As declared in our Coder Manifesto, code is life. A computer program has its own intelligence, is animistic, thus is democratic by nature rather than autocratic. This is why we see the rising of distributed blockchain now as the new web 3.0.

Following is the architecture of AI neural network, capable of deep learning. As complexed and animistic as it is, the anatomy is the same as any other programs.



In a virtual universe, such as a Java Virtual Machine (JVM), we create various virtual entities in the form of Java classes, and define how they behave and interact with each other. Once classes are compiled and loaded, it's the Big Ban in JVM, and Java objects come to life in the virtual simulation.
To be or not to be? That is the question. Animistic program resoundingly chooses to be. Therefore to program, is to create a universe.