# 

# **Programming Languages and Solidity Overview**

# **Table of Contents**

## Overview of programming languages

## What is Pseudocode?

## Pseudocode Implementation

## How to write computer code logic?

## What is Solidity programming?

# **Overview of programming languages**

Nearly everyone has used a computer at some point in their lives. You may have used a computer to play video games, write a paper, buy online, listen to music, or interact with pals on social media. Computers are used to forecast the weather, build aircraft, generate movies, administer companies, conduct financial transactions, and manage factories.

A contemporary computer is defined as "A machine that stores and manipulates information under the direction of a changing program" . This definition has two main components. The first is that computers are information-processing machines. This implies we may input data into a computer, have it change it into new, useful forms, and then have it output or show the data for us to comprehend.

Computers aren't the only devices that can manipulate data. When you use a basic calculator to add up a column of numbers, you input data (numbers), and the calculator processes the data to generate a running total, which is then shown. A gas pump is another basic example. The pump contains 2 inputs while you fill your tank: the current price of petrol per gallon and signals from a sensor that measures the rate of gas pouring into your vehicle. This information is converted by the pump into information about how much petrol you used and how much money you owe.

Although newer versions of these devices may feature integrated computers, we would not consider the calculator or the gas pump to be full-fledged computers. They vary from computers in that they are designed to do a single, focused job. The second half of our definition enters the scene at this point: A changeable software directs the operation of computers. So, what does this imply?

A computer program is a thorough collection of instructions that tells a computer exactly what to perform in a step-by-step manner. When we modify the software, the computer follows a new set of instructions and therefore completes a different job.

This adaptability is what enables your PC to be a word processor one minute, a financial planner the next, and an arcade game the next. The machine remains the same, but the software that controls it changes.

Every computer is nothing more than a device for running (carrying out) instructions.

There are several types of computers. You may be acquainted with Macintoshes, PCs, laptops, tablets, and smartphones, but there are literally thousands of additional actual and hypothetical computers. One of computer science's most astounding findings is that all of these diverse computers have the same capability; given the right programming, any computer can practically accomplish everything that any other computer can do. In this sense, the computer that you may have on your desk is a universal machine. It can do whatever you desire as long as you provide it with a detailed description of the work to be completed. That's a formidable machine!

### Programming 'language': Brain scans reveal coding uses same regions as speech - [Article Link](https://medicalxpress.com/news/2020-06-language-brain-scans-reveal-coding.html)

This proves that Programming is nothing but a language that you learn to speak to communicate with the computer.

# **What is Pseudocode?**

1. The phrase "pseudo code" is often used in the world of algorithmic programming. It's a way for programmers to express how an algorithm really works. Simply put, it's a cooked-up algorithm representation. Pseudo codes are often used to express algorithms so that they may be understood by programmers of all skill levels. Even if you're not a programmer, you can understand pseudocode since it's a fake representation of code.
2. To put it another way, an algorithm is an orderly, methodical approach to solving a problem. An algorithm is used to solve a problem by a programmer. Natural-sounding verbal annotations are used to describe algorithms.
3. Code in the form of comments and informational text written in ordinary English is known as pseudocode. A computer can't compile or comprehend it since it lacks the syntax of any programming language.

Benefits of pseudocode

* Enhances the clarity of any approach. It's one of the finest ways to begin implementing an algorithm.
* Bridges the gap between the algorithm or flowchart and the code. In addition, pseudo code may be used as a rudimentary description of a single developer's software. The attitude to documentation is vital in the industry. When it comes to this, a pseudo-code is essential.
* Pseudo code's primary objective is to make the coding process simpler for programmers by outlining precisely what each line of a code should perform.

# **Pseudocode Implementation**

### 

### 

### Example 1

Wake up in the morning

if snowing

if roads closed

go back to sleep

if roads not closed

get dressed, clean driveway

if not snowing

get dressed

breakfast

drive to work

### 

### 

### Example 2

start

input string

One neuron list

if person A in one neuron

print "Person A is a student of One neuron"

else

print "Please Purchase One neuron to get ott platform for education"

Stop

# 

# 

# **How to write computer code logic?**

### **Practice writing a lot of code**

Start reading lots of open source code and try to make contributions to open source code.

### **Check solutions by other people**

It's a good programming practice to read code written by others, it will help you in understanding different approaches for the same problem.

### **Use a pen and paper to work out solutions**

It is always a good practice to write down the logic of the code on pen and paper, It will help you build strong logic for your programs.

### **Keep learning new things**

Whenever you find something new just code it out, that's the best way to learn programming.

### **Be consistent**

Daily Assign some amount of time apart from your regular for just programming. This will keep you up-to-date.

### **Face problems head-on**

If you have problems early on while learning it's good because it will lay a solid foundation for programming.

### **Don’t lose motivation**

## **What is Solidity?**

In order to create applications on Ethereum and the EVM chain, i.e. smart contracts on the Ethereum blockchain, Solidity is one of the programming languages utilized. One of the most popular decentralized smart contract platforms in crypto is this one. Decentralized exchanges (DEXs), digital art markets (NFTs), financial Decentralized Main (DeFi), Metaverse, etc. may all be created on Ethereum.

Ethereum application developers use the Solidity programming language, which is aimed for high-level programmers. The Solidity programming language makes it easier for developers (developer) to build programs since it uses a mix of letters and numbers.

The language was created by Ethereum contributors under the leadership of Christian Reitwiessner, Vitalik Buterin, Alex Beregszaszi, and others after it was first suggested by Gavin Wood in 2014.

## How Solidity Works on Ethereum

Smart contracts on Ethereum are created using the Solidity programming language, which enables a wide range of possibilities, including the following:

Fungible, Non-Fungible tokens should be minted (NFT).

Compound, Aave, and other fungible tokens may be lent out in decentralized lending markets

Token exchange marketplaces like Opensea, SuperRare, etc. have been established.

The EVM is a critical part of executing Solidity Code once developers have written it in Solidity. EVM is a blockchain-based "virtual computer" that aids in the conversion of developers' Solidity code into Ethereum-based apps.

Solidity enables developers to create "machine level" code that may be run on the EVM at a higher level. Compilers are then used to break down the code that developers write into instructions for processors to execute.

## Advantages & Disadvantages of Solidity language

A popular language for creating blockchain applications, Solidity nevertheless has its pros and cons like any other programming language.

Advantages

* Using Solidity, you can create smart contracts that are safe, clear, and dependable. They have the potential to improve operational efficiency, reduce costs, and reduce reliance on third parties.
* Programming Perceptions found in other languages, such as variables, string manipulation, classes, functions, arithmetic operations, and so on, are used extensively in Solidity. Because C++, Python, and JavaScript are major influences on Solidity, learning and using Solidity will be a breeze if you are already familiar with one of these languages.
* There is a lot of open source documentation on how apps function and numerous use cases available for the Ethereum and EVM chains, making it easy to construct more complicated applications using other people's products written in Solidity.
* On the other hand, if you're constructing your own Decentralized Exchange and want to add novel features like Liquidity Farming, Lending, etc., you may totally fork the source code of Uniswap.

### Disadvantages

* Immutability has many benefits, but it may also be a drawback in some situations. For instance, once a contract has been signed, it cannot be upgraded or enhanced in any way.
* Because Ethereum is based on Blockchain technology, it is unable to gather and update information into the system on its own. In the case of volatile data, such as the price of currency pairings (Forex), this might be a major issue.
* Solidity's major flaw is its infancy (when compared to more mature languages like C or Java). Solidity might be a tough endeavour for newcomers to learn since there are relatively few libraries and references available..

## Why should you learn Solidity Programming?

* When it comes to earning potential, the Blockchain Development industry is one of the most enticing in the world. Since there is a greater demand for Blockchain Developers than there is supply, it is understandable that earnings are high, regardless of whether you work for a business or develop your own programme.
* Because Solidity is a programming language used to construct applications on the Ethereum and EVM chains, it may be regarded as a blockchain programming language by blockchain developers and seems to have the most demand in the Blockchain Developer community. The most used method of communication nowadays.
* If you're hoping to get a career in the web3 industry, you should consider studying Solidity. Because Ethereum's popularity has been on the rise for some time, I believe Solidity will be a valuable skill to have in the future. In other words, there is a lot of content available for you to learn from.
* Considering that the Ethereum and EVM chain ecosystem is now the biggest and most mature in the cryptocurrency world, there is significant room for expansion going forward.