What is Pseudocode?

Pseudocode literally means ‘fake code’. It is an **informal** and **contrived** way of writing programs in which you represent the sequence of actions and instructions (aka algorithms) in a form that humans can easily understand. You see, computers and human beings are quite different, and therein lies the problem.

The language of a computer is very rigid: you are not allowed to make any mistakes or deviate from the rules. Even with the invention of high-level, human-readable languages like JavaScript and Python, it’s still pretty hard for an average human developer to reason and program in those coding languages.

With pseudocode, however, it’s the exact opposite. You make the rules. It doesn’t matter what language you use to write your pseudocode. All that matters is comprehension.

In pseudocode, you don't have to think about semi-colons, curly braces, the syntax for arrow functions, how to define promises, DOM methods and other core language principles. You just have to be able to explain what you're thinking and doing.

Benefits of Writing Pseudocode

When you're writing code in a programming language, you’ll have to battle with strict syntax and rigid coding patterns. But you write pseudocode in a language or form with which you're very familiar.

Since pseudocode is an informal method of program design, you don't have to obey any set-out rules. **You make the rules yourself.**
Pseudocode acts as the bridge between your brain and computer's code executor. It allows you to plan instructions which follow a logical pattern, without including all of the technical details.

Pseudocode is a great way of getting started with software programming as a beginner. You won't have to overwhelm your brain with coding syntax.

In fact, many companies organize programming tests for their interviewees in pseudocode. This is because the importance of problem solving supersedes the ability to 'hack' computer code.

You can get quality code from many platforms online, but you have to learn problem solving and practice it a lot.

Planning computer algorithms with pseudocode makes you meticulous. It helps you explain exactly what each line in a software program should do. This is possible because you are in full control of everything, which is one of the great features of pseudocode.

How to Solve Programming Problems with Pseudocode

Solving programming problems can be hard. Not only do you have the logical part to reckon with, but also the technical (code forming) part as well. I recently uncovered a brilliant and effective formula for solving tricky coding problems.

Here are the steps you can follow to solving programming problems with pseudocode:

Step 1: Understand what the function does

First, you need to understand that all a function does is (optionally) accept data as input, work on the data little by little, and finally return an output. The body of the function is what actually solves the problem and it does so line by line.

Step 2: Make sure you understand the question

Next, you need to read and understand the question properly. This is arguably the most important step in the process.

If you fail to properly understand the question, you won't be able to work through the problem and figure out the possible steps to take. Once you identify the main problem to be solved you'll be ready to tackle it.

Step 3: Break the problem down.

Now you need to break down the problem into smaller parts and sub-problems. With each smaller problem you solve, you'll get closer to solving the main problem.

It helps to represent these problem solving steps in the clearest and most easily understandable way you can – which is psedocode!