PSEUDO-CODING

PSEUDOCODE

An informal high-level description of the operating principle of a computer program or other algorithm.

- It uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading.
- The programming language is augmented with natural language description details, where convenient, or with compact mathematical notation.
- The purpose of using pseudocode is that it is easier for people to understand than conventional programming language code, and that it is an efficient and environment-independent description of the key principles of an algorithm.

KEEP IT LITE:
KEY THINGS TO INCLUDE IN PSEUDOCODE

Assumptions:

state what you are expecting to be available as is, vs. what will be discovered or computed by your code.

Ex. The user will enter the purchase list and their prices, and the tax percentage will be hard coded into the code.

This way later on if your outcomes are wrong, you can validate against your assumptions and adjust them and accordingly correct your code.

Ex. Say the tax percentage will vary from user to user as they may use this code in different locations.

KEEP IT LITE:
KEY THINGS TO INCLUDE IN PSEUDOCODE

Inputs:

- start with the known inputs and leave some space to add stuff as needed while you are writing your pseudocode.
- Ex. Items purchased and their prices.

KEEP IT LITE:
KEY THINGS TO INCLUDE IN PSEUDOCODE

Variables:

- start with the knows, this are variables other then your input and output variables that you might need.
- Ex. Subtotal before taxes, or a constant variable that stores the tax_percentage.

KEEP IT LITE:
KEY THINGS TO INCLUDE IN PSEUDOCODE

Output:

- same rules as inputs but what are the expected outcomes.
- Ex. Total payment amount including taxes

KEEP IT LITE:
KEY THINGS TO INCLUDE IN PSEUDOCODE

Algorithm:

- This is the semi descriptive, semi-coded set of steps
- Ex.

```
foreach item, price in the inputlist
    subtotal += price
finaltotal = subtotal * (1+tax_percentage)
```

HOW TO BUILD A SANDWICH?

Assumptions:

- The bread, bacon, lettuce and tomato will be made available
- The routine will be implemented in a kitchen

Inputs: bread, bacon, lettuce, tomato

variables: butter knife, mayonnaise, mustard

output: sandwich

Algorithm:

- Take a slice of bread
- Spread it with mayonnaise using a butter knife
- Add a leaf of lettuce
- Add a slice of tomato
- Add a strip of bacon
- Add a strip of bacon
- Take a slice of bread
- Spread it with mustard using a butter knife
- Add this slice of bread mustard side down to the sandwich
- Plate the sandwich