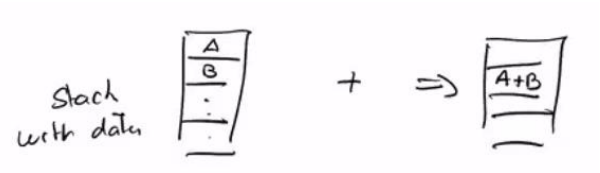
23- Genetic Programming: the stack-based, instruction-driven representation.

The idea is to get closer to a procedural programming language -> set of instructions executed sequentially

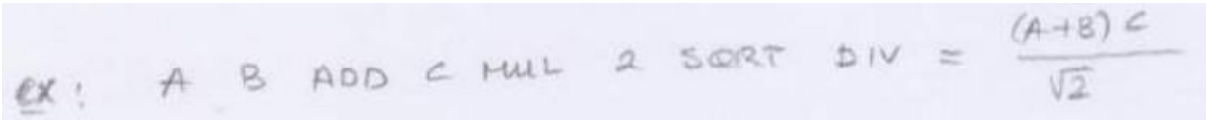
To make sure that a program stays valid after the genetic transformation we use a **stack based** approach

The + operator consumes both top values A and B, performs the operation, and adds the result to the top of the stack



If we code as such, we will be able to have individuals that all have the same length

This will be called **instruction driven representation**



However, it is possible that at some point there aren't enough values in the stack to do an operation (1), or that we end with more than 1 value on the stack (2), if this happens, we:

(1) Skip this operation and leave the stack as is, do the next instruction
(2) Return the value on top of the stack

MUTATION AND CROSSOVER:

We cross the strings of instructions with each other for crossover, and mutation is done with a mutation probability