computed by P is the function computed by A_k^P. A k–ary function F is recursive if it is the k–ary function computed by some program for the basic machine. (In accordance with our convention, a relation is recursive iff its representing function is recursive.)

It is clear that every recursive function is computable. It is not at all evident that every computable function is recursive; but, after some study of the recursive functions, we shall argue that this is also the case.

4. Macros

It is tedious to write programs for the basic machine because of the small number of possible instructions. We shall introduce some new instructions and show that they do not enable us to compute any new functions. The idea is familiar to programmers: the use of subroutines, or, as they are often called nowadays, macros.

For each program P for the basic machine, we introduce a new instruction P^*, called the macro of P. When the machine executes this instruction, it begins executing program P (with whatever numbers happen to be in the registers at the time). If this execution never comes to an end, then the execution of P^* is never completed. If the execution of P is completed, the machine changes the number in the counter to 1 more than the number of the instruction P^* and continues executing instructions. The macro machine is obtained from the basic machine by adding all macros of programs for the basic machine as new instructions. We define the notion of a program computing a function for the macro machine as we did for the basic machine.

We say that the program P and P' are equivalent if the following holds. Suppose that we start two machines with P in the program holder of the first machine, P' in the program holder of the second machine, and the same number in $\mathfrak{R}i$ in both machines for all i. Then either both machines will compute forever;