show that $\overline{F}$ is recursive iff $F$ is recursive. We cannot use the preceding equation as an explicit definition; for we cannot fill in ... until we know the value of the argument $y$. However, we have the explicit definitions

$$\overline{F}(y,\vec{x}) \simeq \mu z(Seq(z) \ \& \ lh(z) = y \ \& \ (\forall i < y)((z)_i = F(i,\vec{x}))),$$
$$F(y,\vec{x}) \simeq (\overline{F}(y+1,x))_y.$$

Given a total function $G$, we may define a total function $F$ by induction on $y$ as follows:

$$F(y,\vec{x}) \simeq G(\overline{F}(y,\vec{x}),\ y,\ \vec{x}).$$

We shall show that if $G$ is recursive, then $F$ is recursive. By the above, it is enough to show that $\overline{F}$ is recursive. But $\overline{F}$ has the inductive definition

$$\overline{F}(0,\vec{x}) \simeq <\ >,$$
$$\overline{F}(y+1,\vec{x}) = \overline{F}(y,\vec{x}) * < G(\overline{F}(y,\vec{x}),\ y\ ,\vec{x}) >.$$

An inductive definition of this sort is called a course–of–values inductive definition.

## 8. Indices

We are now going to assign codes to some of the elements in the operation of the basic machine. This will lead to some of the most important theorems of recursion theory.

First, a general remark on coding. Suppose that we want to code the members of $I$. We may be able to identify each member $b$ of $I$ with a finite sequence $a_1,...,a_k$ of objects which have already been coded. We can then assign to $b$ the code $<x_1,...,x_k>$, where $x_i$ is the code of $a_i$.

We begin by assigning codes to the instructions for the basic machine. We assign the code $<0,i>$ to the instruction INCREASE $\mathfrak{r}i$; the code $<1,i,n>$ to the instruction DECREASE $\mathfrak{r}i,n$; and the code $<2,n>$ to the instruction GO TO $n$. If $P$ is a program consisting of $N$ instructions with codes $x_1,...,x_N$, we assign the code $<x_1,...,x_N>$ to $P$.

We define