[A]	$X_1 \leftarrow X_1 - 1$	(I_1)
	$X_2 \leftarrow X_2 - 1$	(I_2)
	If $X_1 \neq 0$ GOTO B	(I_3)
	$Y \leftarrow Y + 1$	(I_4)
	GOTO E	(I_5)
[B]	If $X_2 \neq 0$ GOTO A	(I_6)
	GOTO E	(I_7)

$\#X_1 = 2$, $\#X_2 = 4$, $\#Y = 1$, $\#A = 1$, $\#B = 2$, $\#E = 5$

Note: $<x, y> = 2^x (2y + 1) - 1$

$\#(I_1)$ = is $<a, <b, c>> = <1, <2, 1> = <1, 11> = 45$

$\#(I_3) = <0, <4, 1> = <0, 47> = 94$

Given # (I_3) = 94 find a, b, c, values

$x = l(z)$ = largest number such that $2^x \mid z + 1$
$y = r(z)$ = solution of $2y + 1 = (z+ 1) / 2^x$

$a = l(94) = 0$, $<b, c> = r(94) = 47$

$b = l(47) = 4$, $c = r(47) = 1$

a = 0 implies instruction is unlabeled.

c = 1 implies variable is $c + 1 = r(r(94)) + 1 = 2 = X_1$

b = 4 implies label is $b - 2 = l(r(94)) - 2 = 4 - 2 = B$

$\#(I_2) = <0, <2, 3> = <0, 27> = 54$

$\#(I_4) = <0, <1, 0> = <0, 1> = 2$

We will consider the program $\mathscr{P}$ consisting of the 4 instructions computed above.

$\#\mathscr{P} = [\#(I_1), \#(I_2), ,\#(I_3), ,\#(I_4)] -1 = [45, 54, 94, 2] -1 = 2^{45} \times 3^{54} \times 5^{94} \times 7^{2} - 1$

We will consider the program $\mathscr{P}$ consisting of the 4 instructions computed above.

$\#\mathscr{P}$ = [#(I_1), #(I_2), ,#(I_3), ,#(I_4)] -1 = [45, 54, 94, 2] -1 = $2^{45} \times 3^{54} \times 5^{94} \times 7^2 - 1$

We show how to use simulate the behavior of this program (of 4 instructions) by the universal program $\mathscr{U}_2$. Thus we consider $\mathscr{U}_2$ that computes $\Phi^{(2)}(X_1, X_2, X_3)$ where X_3 is $\#\mathscr{P}$ above and the input values are $X_1 = 2$ and $X_2 = 1$. Hence $\mathscr{U}_2$ computes $\Phi^{(2)}(2, 1, \#\mathscr{P})$.

$Z \leftarrow X_3 + 1$ Hence Z = $\#\mathscr{P}$ + 1 = [#(I_1), #(I_2), ,#(I_3), ,#(I_4)] = $2^{45} \times 3^{54} \times 5^{94} \times 7^2$.

Note that Z is set to the program number of $\mathscr{P}$ plus 1 which is a single number. In order to get the component instructions #(I_i) we would need to have a macro that could figure out these values [45, 54, 94, 2] = [(Z_1), (Z_2), (Z_3), (Z_4)]. We have previously shown in the section on Godel numbering that the function (Z_i) is primitive recursive and hence could be done by a macro.

$S \leftarrow \prod(i=1 \text{ to } 2)\ (p_{2i})^{Xi}$ Hence S = $3^{X1} \times 7^{X2} = 3^2 \times 7^1$ [0, 2, 0, 1] which is the initialization of input values. Not that the value of all other variables (including Y) is initially 0 and the value of Y is obtained by the exponent of 2 in the product S.

$K \leftarrow 1$. This ends the initialization phase.

We next turn to the code of the universal program which is shown in Figure 3.1 of the text in Chapter 4 and apply it to our specific program $\mathscr{P}$ and inputs: $X_1 = 2$ and $X_2 = 1$.

$U \leftarrow r((Z_K))$ K = 1, hence (Z_1) is simply 45, which is the value of instruction #I_1.

Thus U = $r((Z_K))$ is simply 11 which is <2,1> = <b, c>. Thus, r(U) = c = 1 which means that the variable number is r(U) + 1 = 2 which means it is X_1. Similarly, l(U) = b = 2 implies that the instruction is of the form $V \leftarrow V - 1$. Note that computing both r(U) and l(U) would be done by macros. The instruction to be executed is thus simply $X_1 \leftarrow X_1 - 1$.

$P \leftarrow p_{r(U)+1}$ r(U) + 1 = 2 which is the variable number of X_1. Now, p_2 is the second prime: 3.

Note that *S* stores all the values of variables as a product of primes to exponents of the value of the variables. Thus the exponent of 3 in *S* is the value of the variable X_1. It is currently 2. In order to execute the instruction $X_1 \leftarrow X_1 - 1$ we divide S by P or compute "integer part" of [S/P]. Note that in this case in *S* the exponent of 3 would reduce by 1 and become 1:

$S \leftarrow [S/P]$.

After this K is incremented by 1 and the second instruction is similarly executed.

See the universal program 3.1.

$$
\begin{array}{ll}
 & Z \leftarrow X_{n+1} + 1 \\
 & S \leftarrow \prod_{i=1}^{n} (p_{2i})^{X_i} \\
 & K \leftarrow 1 \\
[C] & \text{IF } K = \text{Lt}(Z) + 1 \lor K = 0 \text{ GOTO } F \\
 & U \leftarrow r((Z)_K) \\
 & P \leftarrow p_{r(U)+1} \\
 & \text{IF } l(U) = 0 \text{ GOTO } N \\
 & \text{IF } l(U) = 1 \text{ GOTO } A \\
 & \text{IF } \sim(P \mid S) \text{ GOTO } N \\
 & \text{IF } l(U) = 2 \text{ GOTO } M \\
 & K \leftarrow \min_{i \le \text{Lt}(Z)} [l((Z)_i) + 2 = l(U)] \\
 & \text{GOTO } C \\
[M] & S \leftarrow \lfloor S/P \rfloor \\
 & \text{GOTO } N \\
[A] & S \leftarrow S \cdot P \\
[N] & K \leftarrow K + 1 \\
 & \text{GOTO } C \\
[F] & Y \leftarrow (S)_1
\end{array}
$$

Figure 3.1. Program $\mathscr{U}_n$, which computes $Y = \Phi^{(n)}(X_1, \ldots, X_n, X_{n+1})$.

instruction counter is increased by 1 and the computation returns to process the next instruction. To conclude the program,

$$[F] \quad Y \leftarrow (S)_1$$

On termination, the value of Y for the program being simulated is stored as the exponent on $p_1(= 2)$ in S. We have now completed our description of $\mathscr{U}_n$ and we put the pieces together in Fig. 3.1.

For each $n > 0$, the sequence

$$\Phi^{(n)}(x_1, \ldots, x_n, 0), \Phi^{(n)}(x_1, \ldots, x_n, 1), \ldots$$

enumerates all partially computable functions of n variables. When we want to emphasize this aspect of the situation we write

$$\Phi_y^{(n)}(x_1, \ldots, x_n) = \Phi^{(n)}(x_1, \ldots, x_n, y).$$

It is often convenient to omit the superscript when $n = 1$, writing

$$\Phi_y(x) = \Phi(x, y) = \Phi^{(1)}(x, y).$$