Iterative Program Analysis
Part I

Mooly Sagiv
http://www.cs.tau.ac.il/~msagiv/courses/pa11.html
Tel Aviv University
640-6706

Textbook: Principles of Program Analysis
Chapter 2.1 (modified)

A Simple Example Program

$[x \mapsto 0, y \mapsto 0, z \mapsto 0]$

z = 3

$[x \mapsto 0, y \mapsto 0, z \mapsto 3]$

x = 1

$[x \mapsto 1, y \mapsto 0, z \mapsto 3]$

while (x > 0) (

$[x \mapsto 1, y \mapsto 0, z \mapsto 3]$

if (x = 1) then y = 7

$[x \mapsto 1, y \mapsto 7, z \mapsto 3]$

else y = z + 4

$[x \mapsto 1, y \mapsto 7, z \mapsto 3]$

x = 3

$[x \mapsto 3, y \mapsto 7, z \mapsto 3]$

print y

$[x \mapsto 3, y \mapsto 7, z \mapsto 3]$

)

A Simple Example Program

$[x \mapsto 0, y \mapsto 0, z \mapsto 0]$

z = 3

$[x \mapsto 0, y \mapsto 0, z \mapsto 3]$

x = 1

$[x \mapsto 1, y \mapsto 0, z \mapsto 3]$

while (x > 0) (

$[x \mapsto \top, y \mapsto 0, z \mapsto 3]$

if (x = 1) then y = 7

$[x \mapsto 1, y \mapsto 7, z \mapsto 3]$

else y = z + 4

$[x \mapsto 1, y \mapsto 7, z \mapsto 3]$

x = 3

$[x \mapsto 3, y \mapsto 7, z \mapsto 3]$

print y

$[x \mapsto 3, y \mapsto 7, z \mapsto 3]$

)

A Simple Example Program

$[x \mapsto 0, y \mapsto 0, z \mapsto 0]$

z = 3

$[x \mapsto 0, y \mapsto 0, z \mapsto 3]$

x = 1

$[x \mapsto 1, y \mapsto 0, z \mapsto 3]$

while (x > 0) (

$[x \mapsto \top, y \mapsto 0, z \mapsto 3]$

if (x = 1) then y = 7

$[x \mapsto \top, y \mapsto 7, z \mapsto 3]$

else y = z + 4

$[x \mapsto 1, y \mapsto 7, z \mapsto 3]$

x = 3

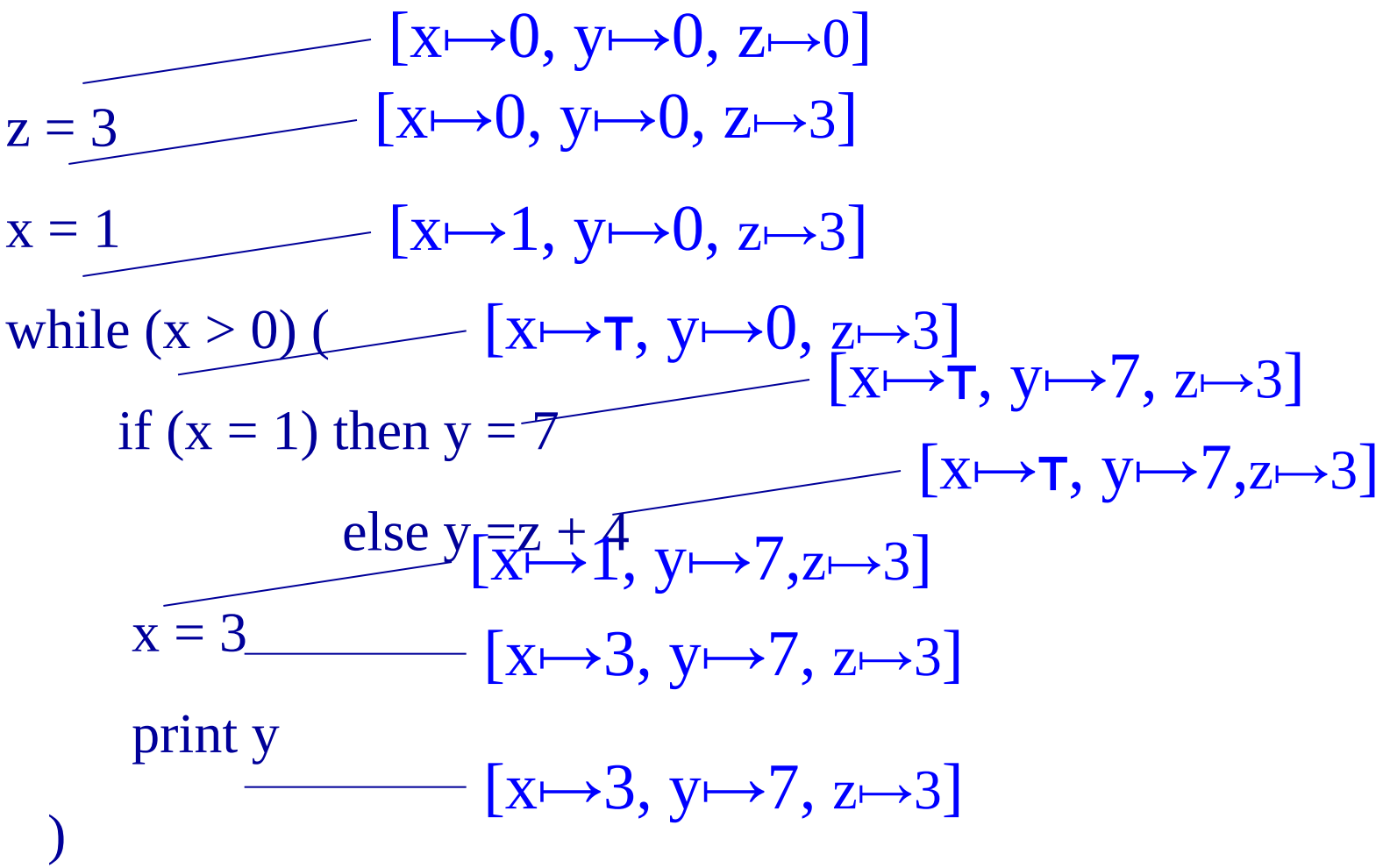
$[x \mapsto 3, y \mapsto 7, z \mapsto 3]$

print y

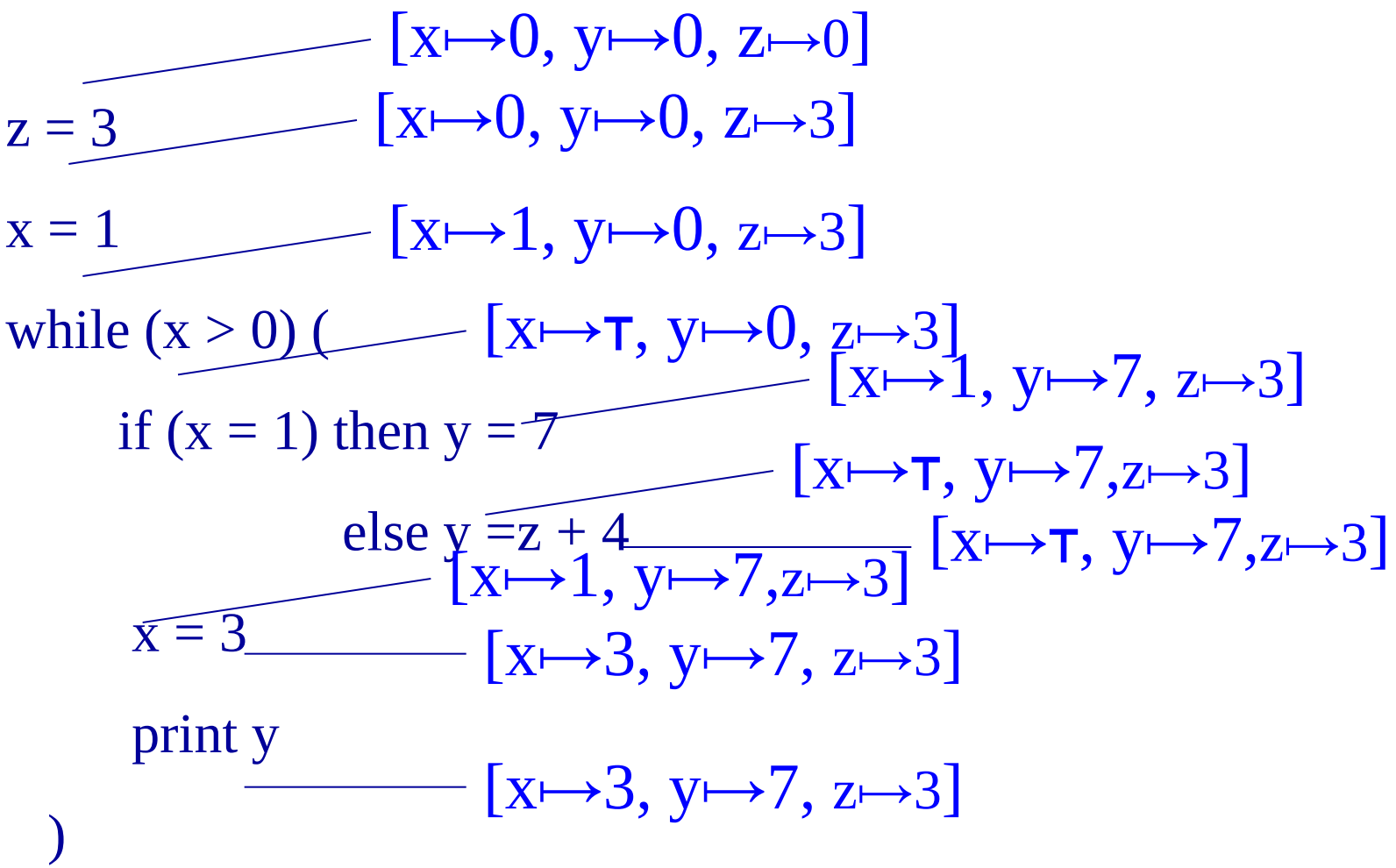
$[x \mapsto 3, y \mapsto 7, z \mapsto 3]$

)

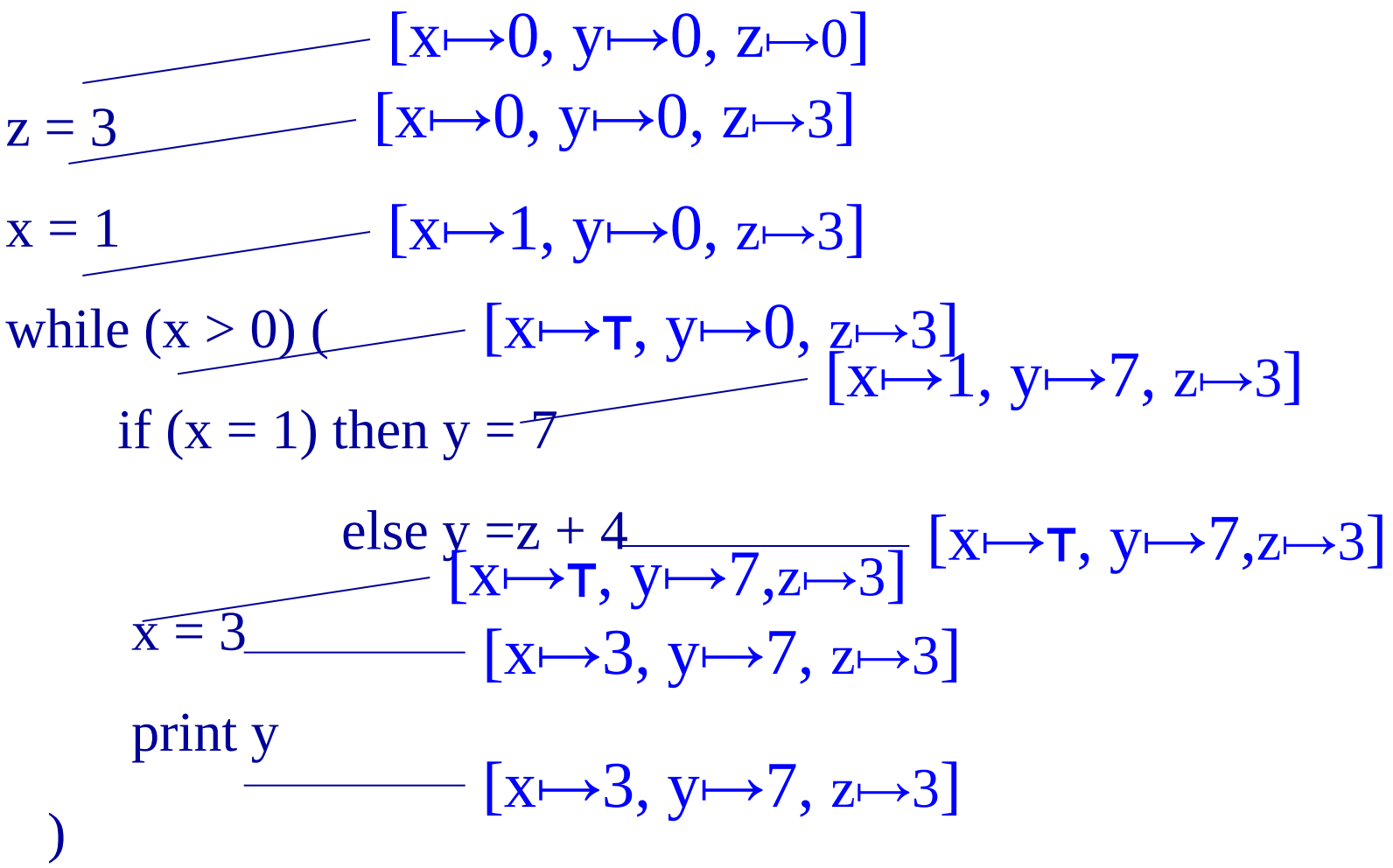
A Simple Example Program



A Simple Example Program



A Simple Example Program

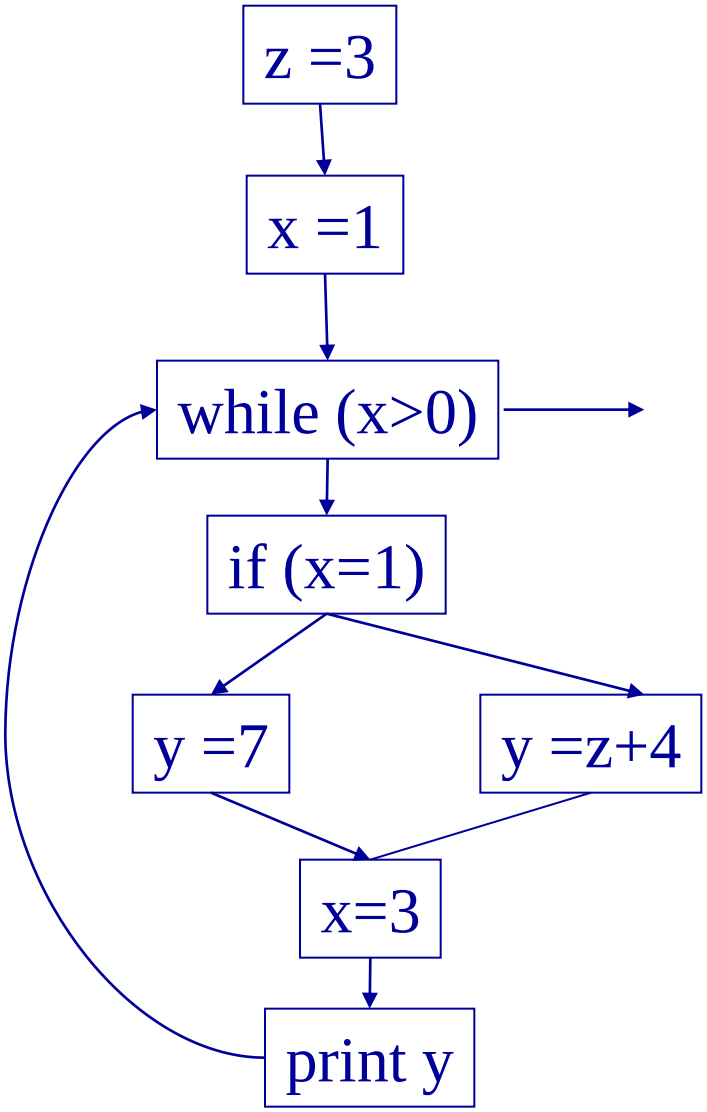


Computing Constants

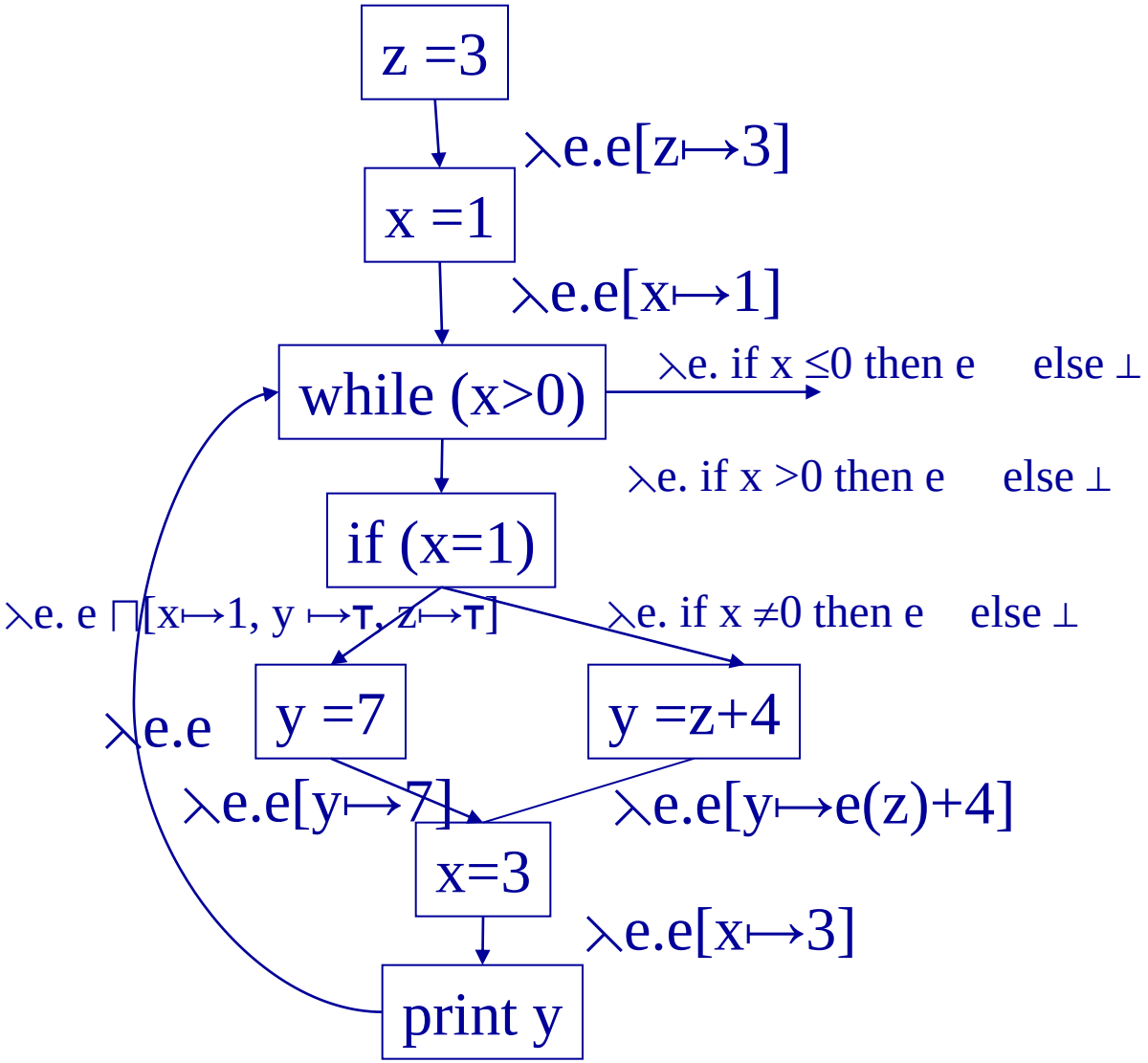
- Construct a control flow graph (CFG)
- Associate transfer functions with control flow graph edges
- Iterate until a solution is found
- The solution is unique
  - But order of evaluation may affect the number of iterations

Constructing CFG

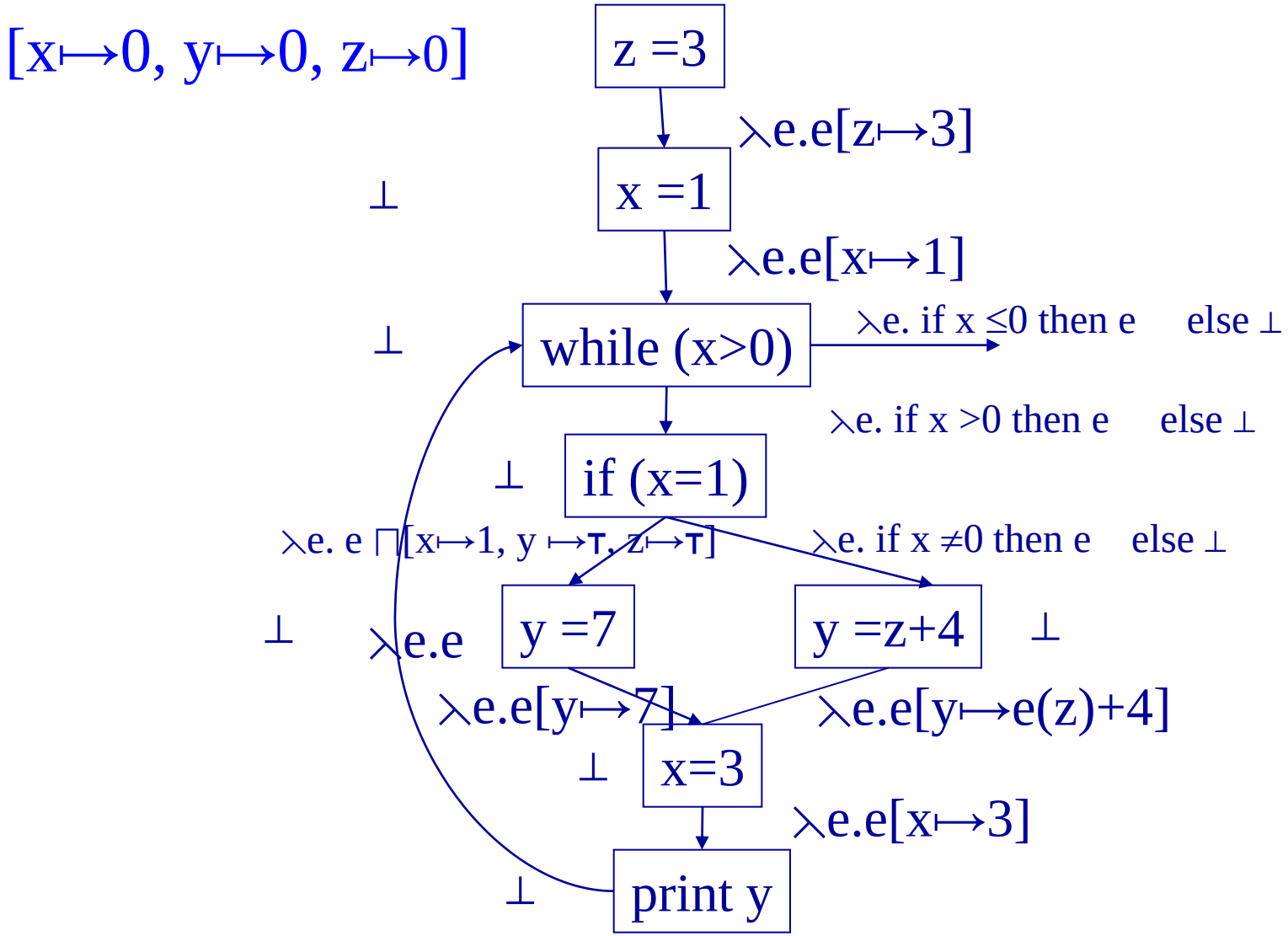
```
z = 3
x = 1
while (x > 0) (
        if (x = 1) then y = 7
                   else y =z + 4
        x = 3
        print y
   )
```



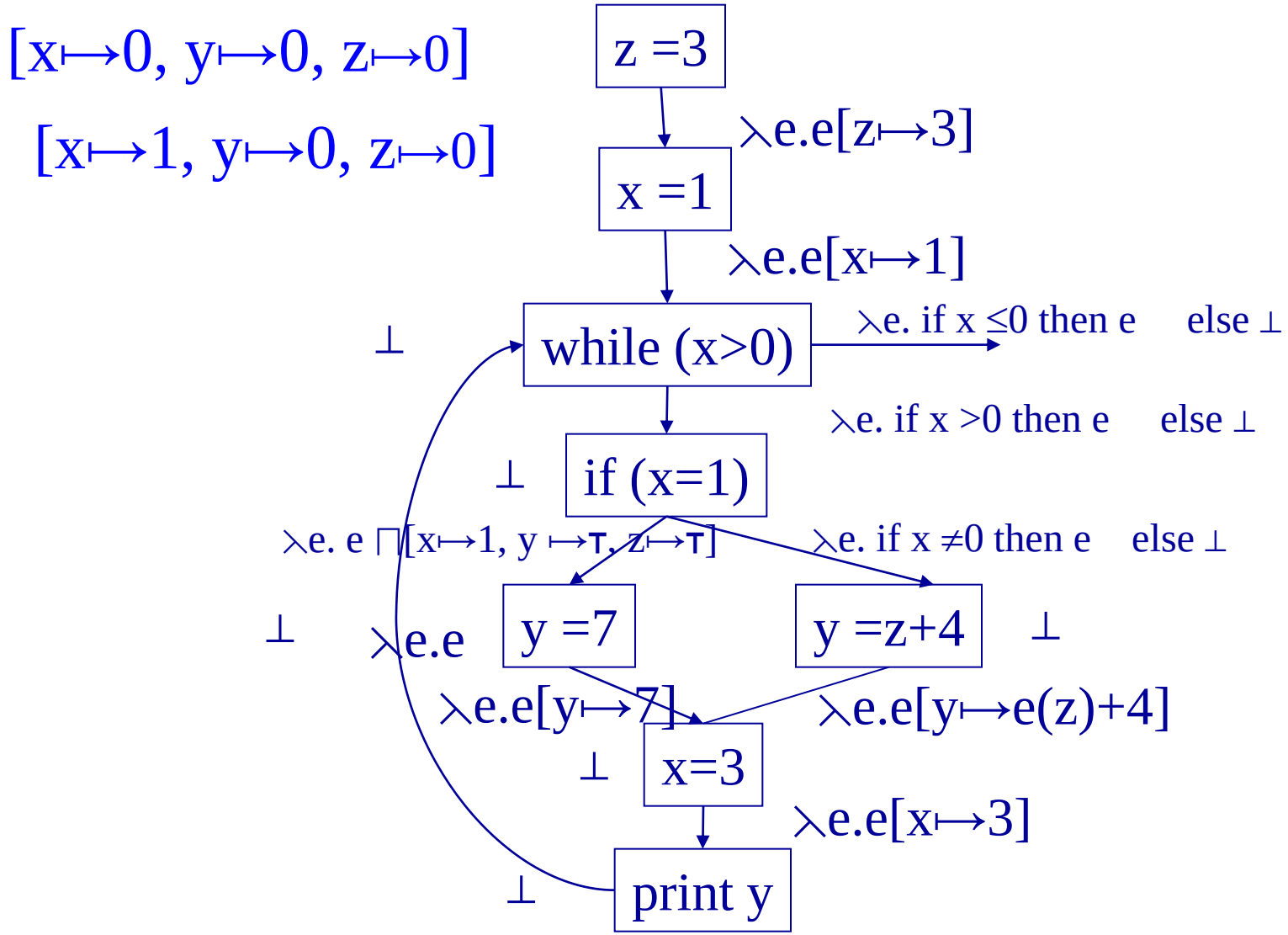
Associating Transfer Functions



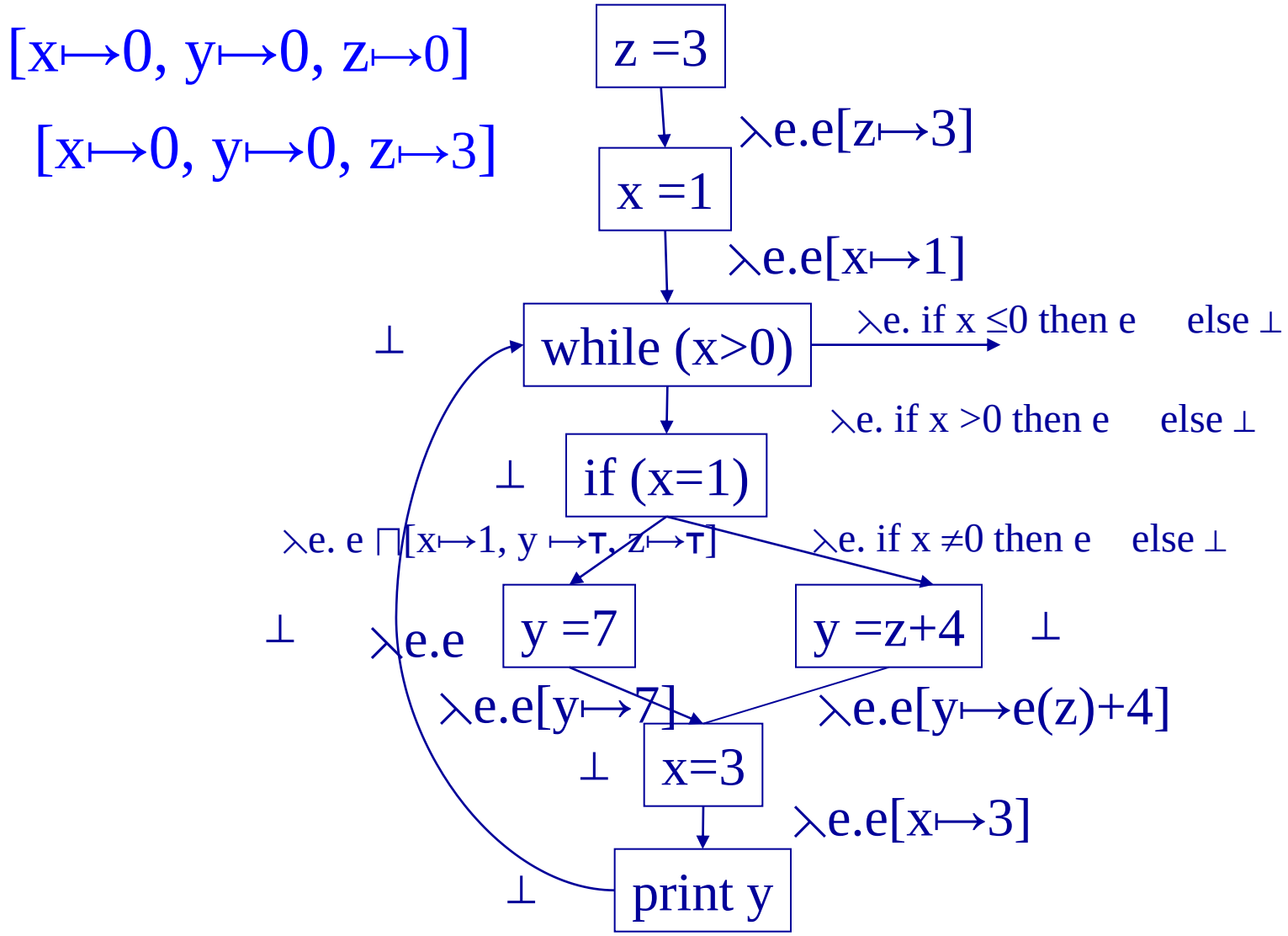
Iterative Computation



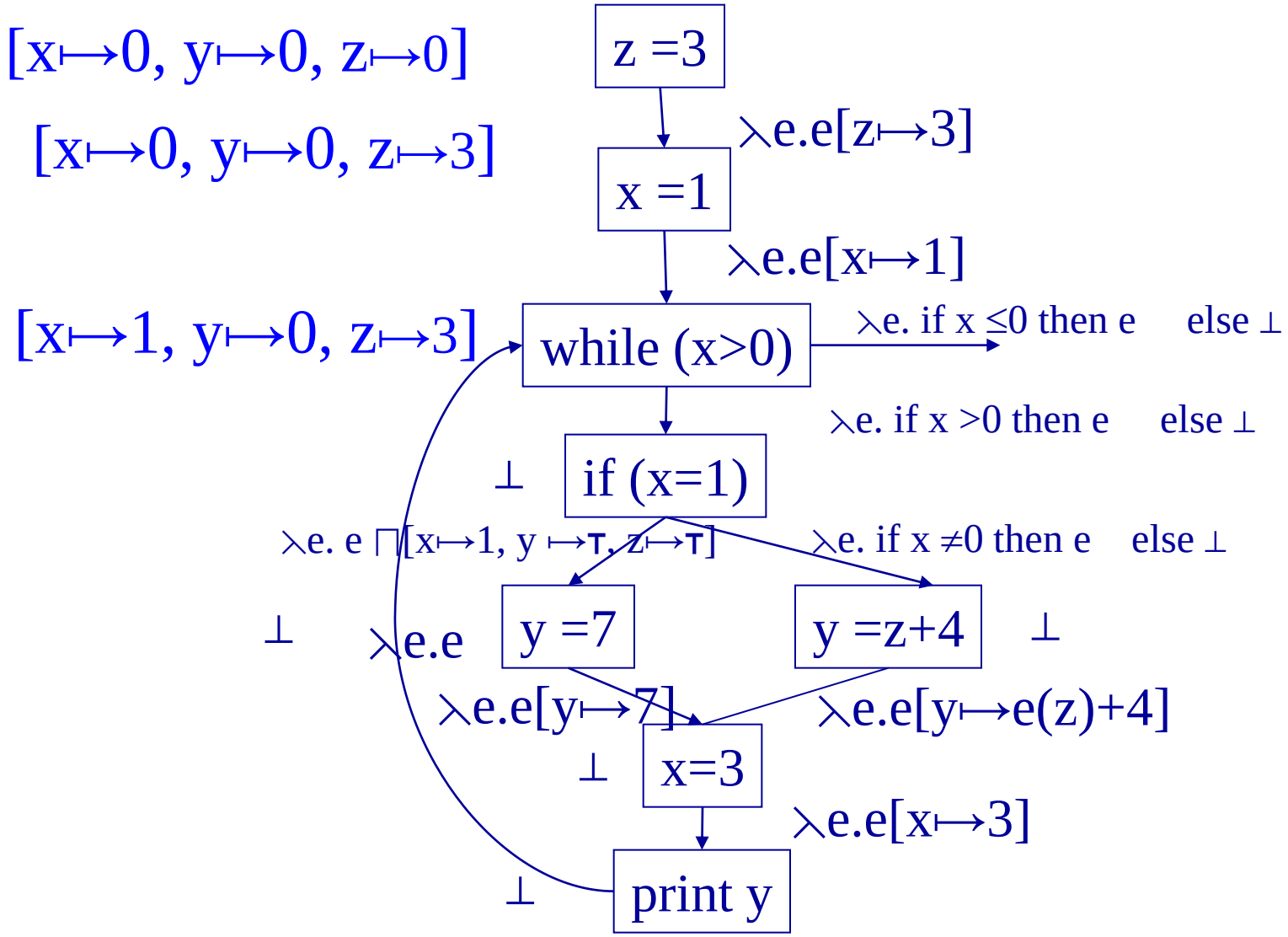
Iterative Computation



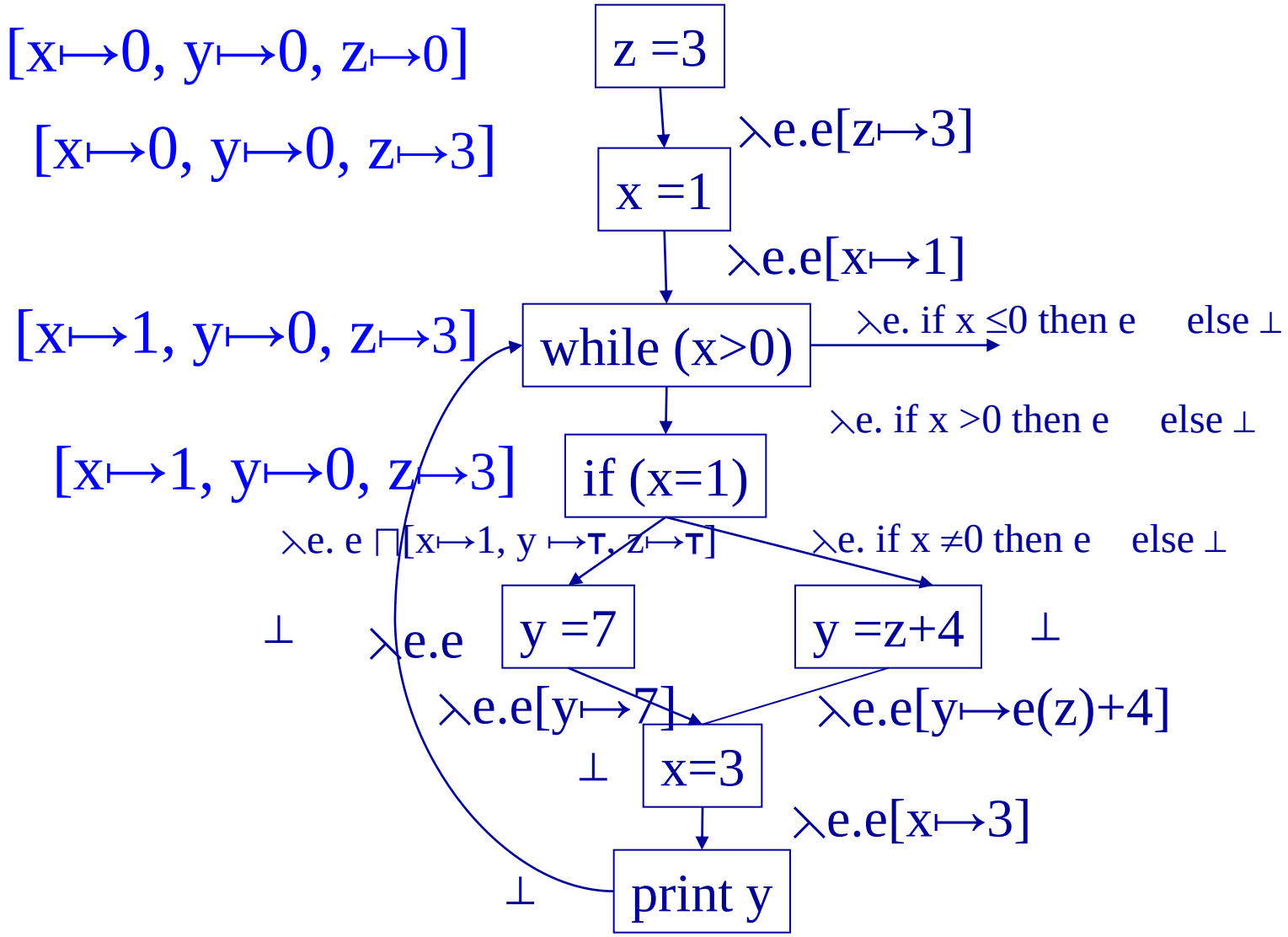
Iterative Computation



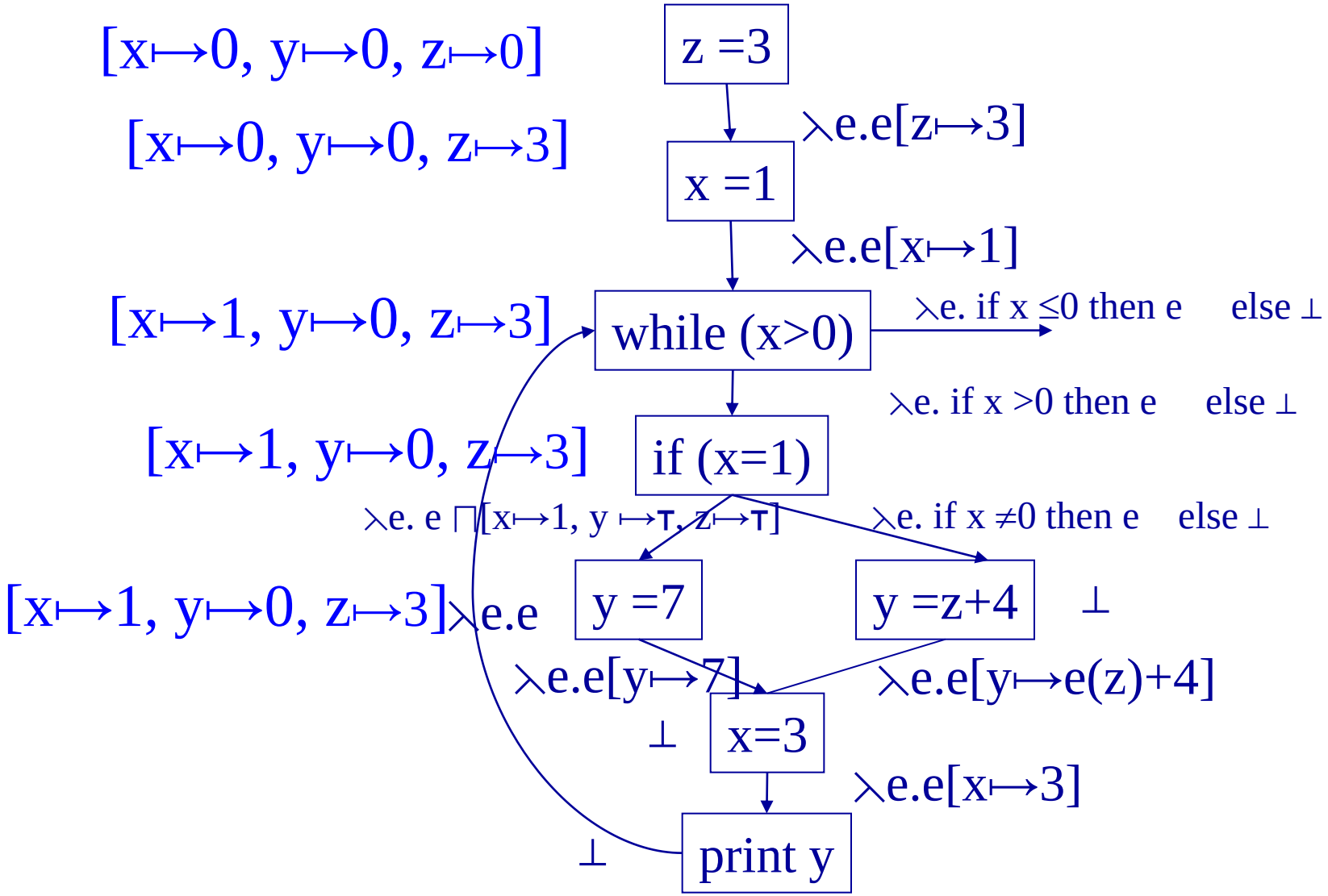
Iterative Computation



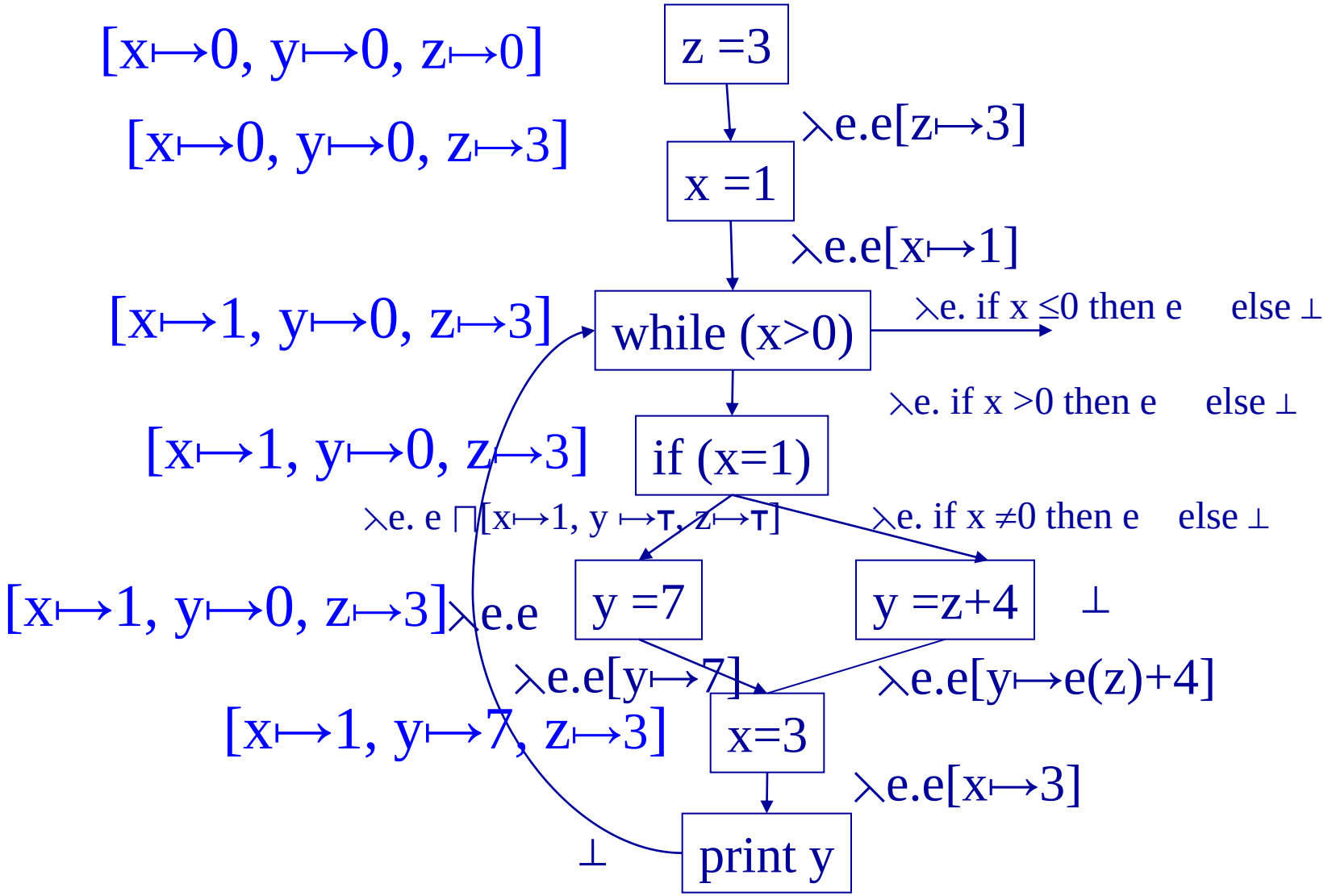
Iterative Computation



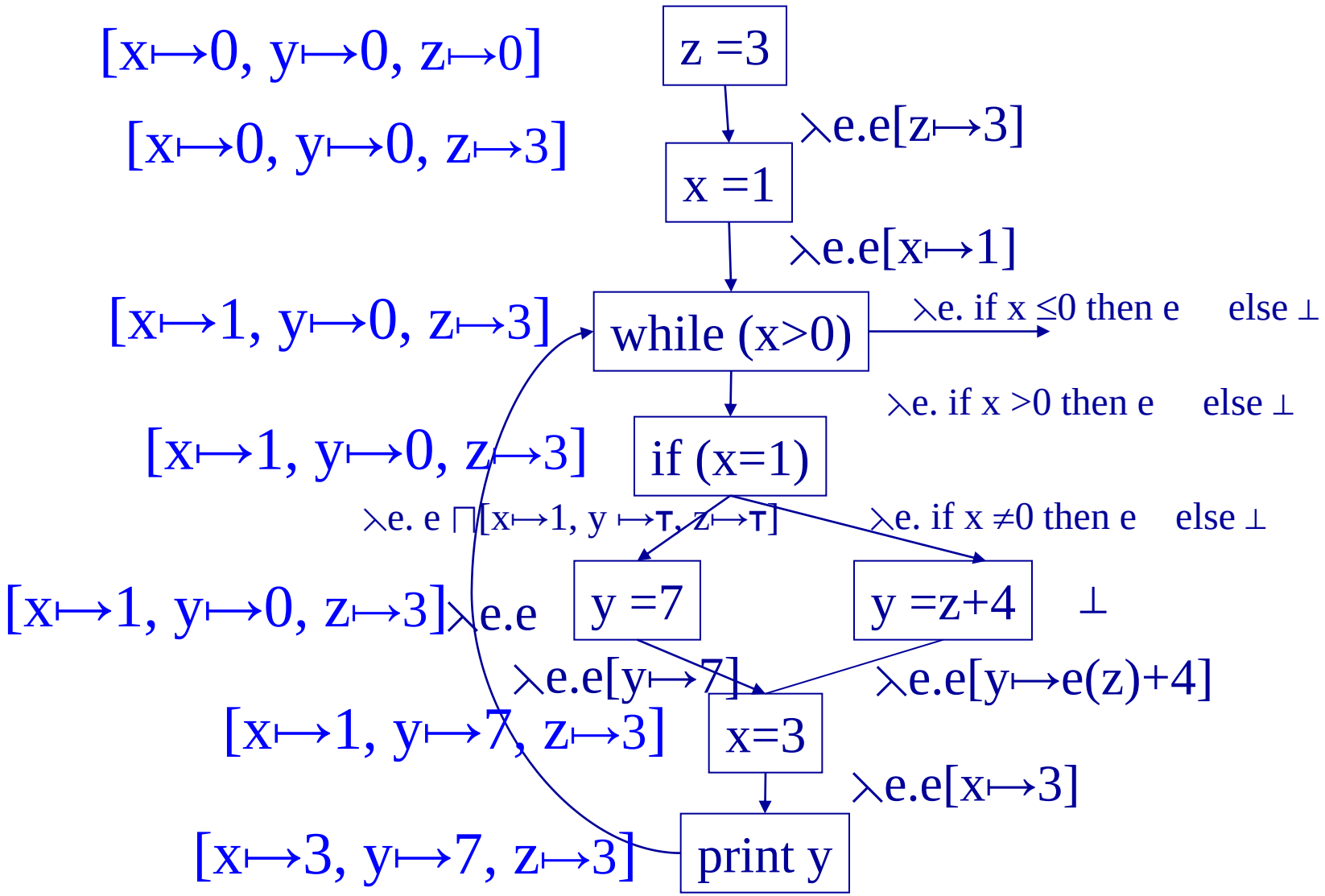
Iterative Computation



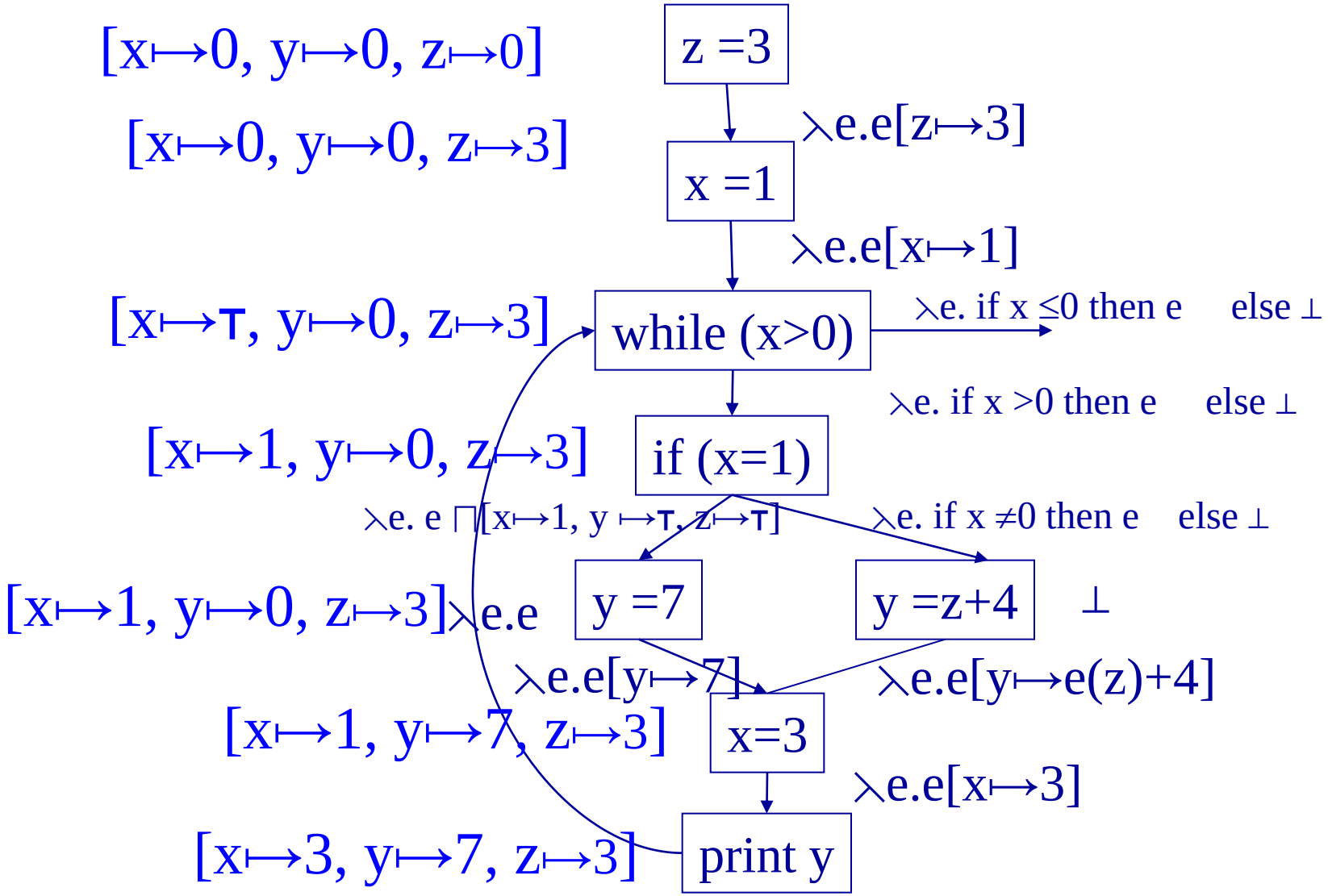
Iterative Computation



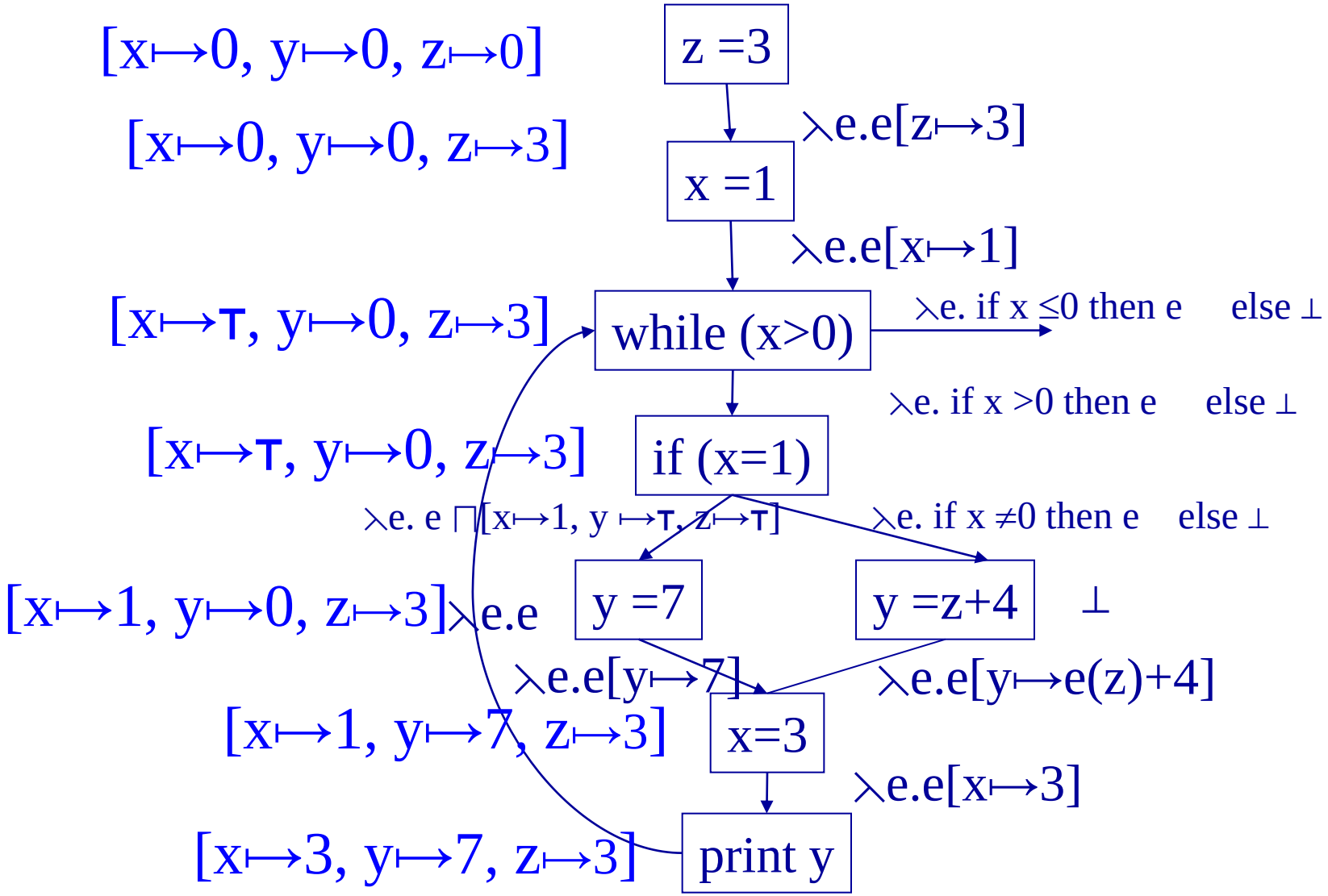
Iterative Computation



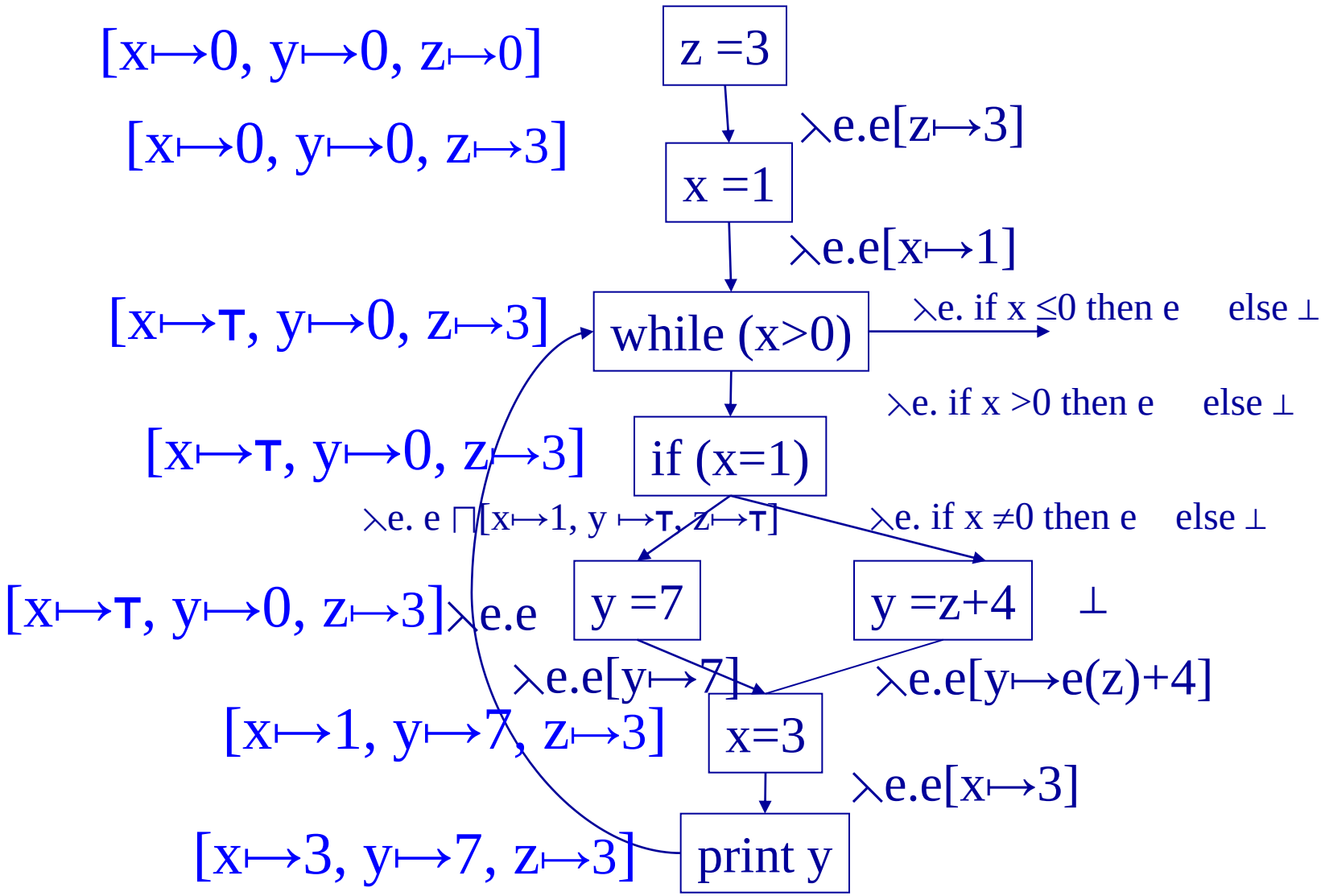
Iterative Computation



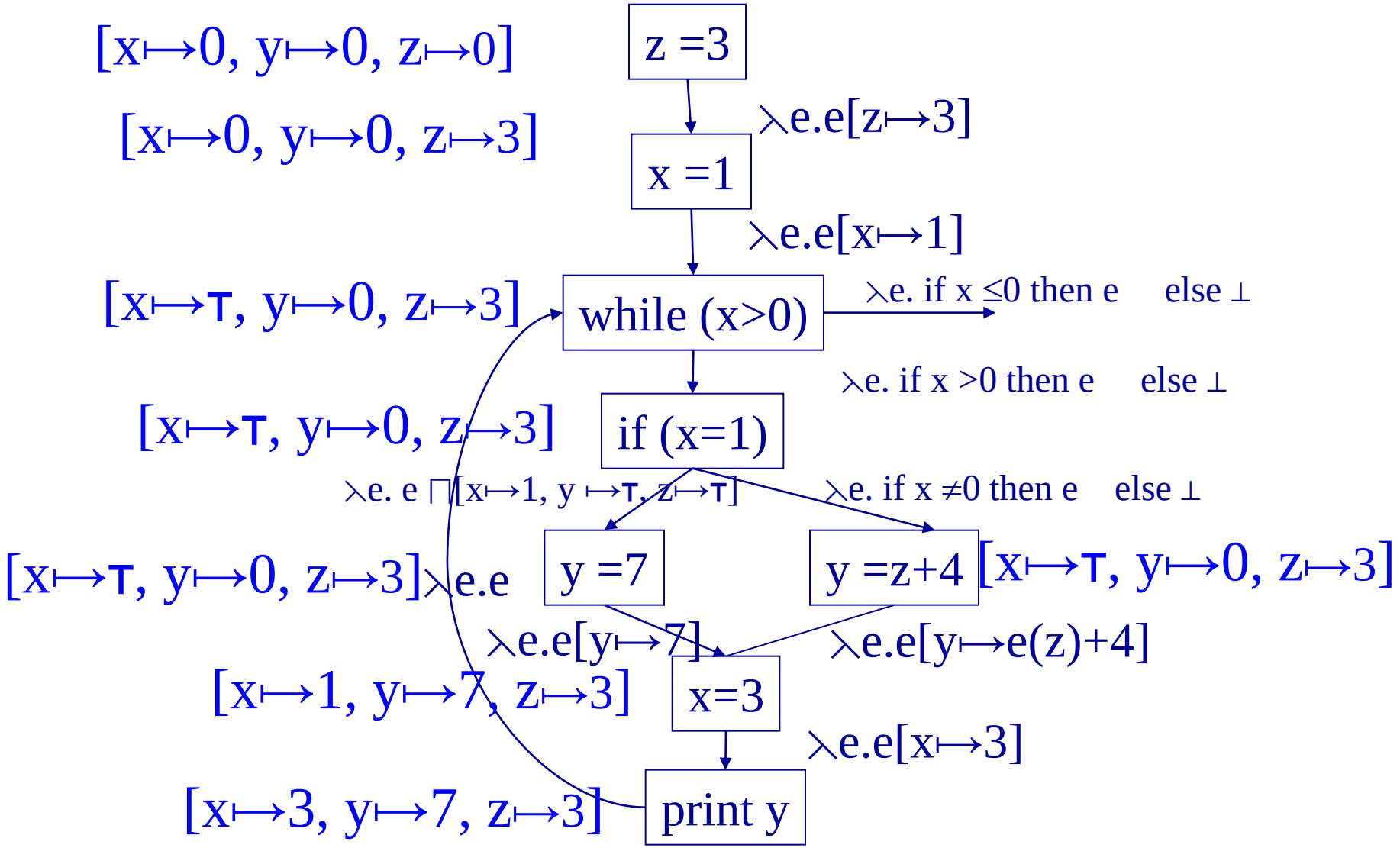
Iterative Computation



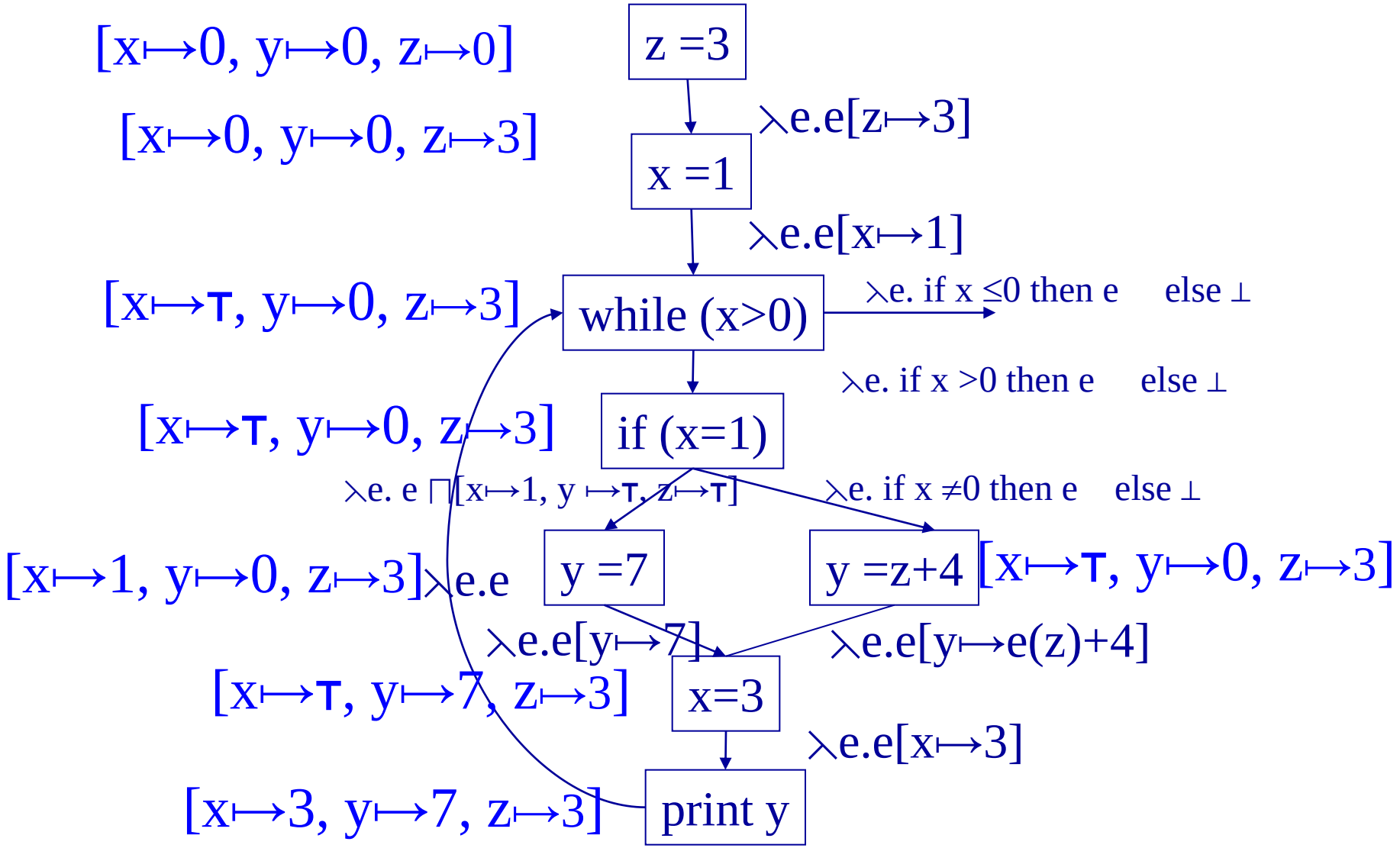
Iterative Computation



Iterative Computation



Iterative Computation



Low Level View

- Explicitly represent the program counter
- Create an abstract transition system which represents the analysis
- Execute transitions in arbitrary order

Low Level View (Example)

State : PC → (Var → Val)

Transformer: State → State

1: z = 3

2: x = 1

while 3: (x > 0) (

4: if (x = 1) then 5: y = 7

else 6: y =z + 4

7: x = 3

8: print y

)

$$\lambda S.\ \lambda pc.\ \lambda v:$$

$$\begin{cases} 0 & pc = 1 \\ S\ 1\ [z \mapsto 3]\ v & pc = 2 \\ (S\ 2\ [x \mapsto 1] \sqcup S\ 8)\ v & pc = 3 \\ S\ 3\ \ v & pc = 4 \\ (S\ 4 \sqcap [x \mapsto 1,\ y \mapsto \top,\ z \mapsto \top])\ v & pc = 5 \\ S\ 4\ v & pc = 6 \\ (S\ 5\ [y \mapsto 7] \sqcup (S\ 6\ [y \mapsto (S\ 6\ z) + 4] & pc = 7 \\ S\ 7\ [x \mapsto 3]\ v & pc = 8 \end{cases}$$

Summary

- Chaotic iterations is a powerful technique
- Easy to implement
- Rather precise
- But expensive
  - More efficient methods exist for structured programs