Below we have the control-flow graph of a simple program, and the equations that are produced to solve reaching-definition analysis for this program. The questions in this exercise refer to these equations.

```
01 int foo() {
02   int x = 1;
03   while (x == 0) {
04     x = 2;
05   }
06   return x;
07 }
```

Figure 1: example of unreachable code.

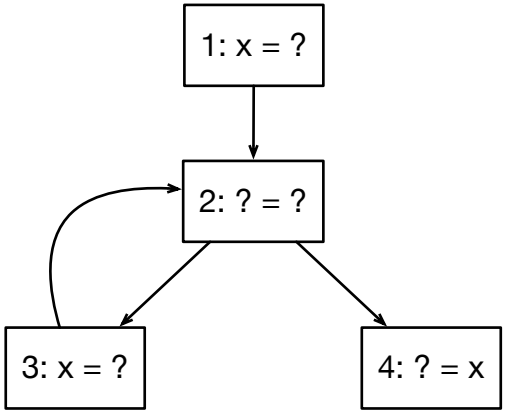

Figure 2: the CFG of the program in Fig-1

IN[1] = {}
OUT[1] = (IN[1] \ {1, 3, 4}) U {1}
IN[2] = OUT[1] U OUT[3]
OUT[2] = IN[2]
IN[3] = OUT[2]
OUT[3] = (IN[3] \ {1, 3, 4}) U {3}
IN[4] = OUT[2]
OUT[4] = (IN[4] \ {1, 3, 4}) U {4}

Figure 3: equations for the reaching-definition analysis extracted from the CFG in Fig-2.

UFMG
DCC
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
Compilers Laboratory

1. What is the general format of the equations? Write a general formula for the IN and OUT sets, assuming an instruction "v = E" at program point p. Let defs(v) be the sites where variable v is defined, and assume that E is any expression (does the format of E bear any influence on the equations?).

2. What would be a solution to the equations in Figure 3? (Before answering, take a look into Q4 below).

3. Is the solution that you wrote for Q2 unique?

4. When solving the equations in Fig-3, to answer Q2, how many times did you have to evaluate each one of these eight equations? Was this number the same for all of them?

5. If you initialize every one of these four IN and OUT sets with the empty set, and keeps evaluating the equations, is it possible that, at some point, at least one of these sets will become smaller?

6. What is the meaning of the solution that you got for Q2? Can the definition of x at Line 04 in Figure 1 ever reach the return point in Line 06?