```
1 numbers=left  def fun(A,b, m=50): n = A.shape[1] x = np.zeros(n) N = 1/A.diagonal() for k in range(m): x = x - N
  * (A @ x - b) return x
```

Algorithm 1: Unknown Python code

1. Please describe what each line of the code does (please do not write into the pseudocode).
2. Which algorithm is implemented and what is its purpose? Which role does `N` play here?

Solution:

1.

$(1P)$ 1 function declaration with input $A, b, m := 50$

$(1P)$ 2 set $n :=$ number of columns of A

$(1P)$ 3 set $x = (0, \ldots, 0)^T \in \mathbb{R}^n$

$(1P)$ 4 set $N = (\frac{1}{a_{11}}, \ldots, \frac{1}{a_{nn}})^T$ (inverting diagonal elements)

5 /

$(1P)$ 6 for-loop from $k = 0$ to $k = m - 1$

$(1P)$ 7 update x by $x - D^{-1}(Ax - b), \ D = \begin{pmatrix} a_{11} & & 0 \\ & \ddots & \\ 0 & & a_{nn} \end{pmatrix}$

$(1P)$ 8 output value of x

2.

$(1P)$ Jacobi iteration

$(1P)$ solve $Ax = b$ iteratively

$(1P)$ N is preconditioner (with the "hope" $\rho(I - NA) < 1$)