```python
import os, sys
sys.path.append(os.path.join(os.path.dirname(__file__), '../../ch02'))

from program2_1 import Dvector
from program2_2 import Dmatrix
from program2_3 import input_vector, input_matrix

N = 4

def main():
    global N

    a = Dmatrix(1, N, 1, N)
    b = Dvector(1, N)

    # ファイルのオープン
    with open("input.dat", "r") as fin:
        with open("output.dat", "w") as fout:
            input_matrix( a, 'A', fin, fout ) # 行列 A の入出力
            input_vector( b, 'b', fin, fout ) # ベクトル b の入出力
            b = gauss( a, b )                 # ガウス消去法

            # 結果の出力
            fout.write("Ax=b の解は次の通りです\n")
            for i in range(1, N+1):
                fout.write(f"{b[i]}\n")


# 部分ピボット選択付きガウス消去法
def gauss(a: Dmatrix, b: Dvector, N:int=N):
    eps = 2.0 ** -50.0 # eps = 2^{-50}とする

    for k in range(1, N):
        # ピボットの選択
        amax = abs(a[k][k])
        ip = k
        for i in range(k+1, N+1):
            if abs(a[i][k]) > amax:
                amax = abs(a[i][k])
                ip = i

        # 正則性の判定
        if amax < eps:
            print("入力した行列は正則ではない!!")

        # 行交換
        if ip != k:
            for j in range(k, N+1):
                a[k][j], a[ip][j] = a[ip][j], a[k][j]
            b[k], b[ip] = b[ip], b[k]

        # 前進消去
        for i in range(k+1, N+1):
            alpha = - a[i][k] / a[k][k]
            for j in range(k+1, N+1):
                a[i][j] += alpha * a[k][j]
            b[i] += alpha * b[k]

    # 後退代入
    b[N] = b[N] / a[N][N]
    for k in range(N-1, 0, -1):
        tmp = b[k]
        for j in range(k+1, N+1):
            tmp -= a[k][j] * b[j]
        b[k] = tmp / a[k][k]

    return b


if __name__ == "__main__":
    main()
```