```python
import os, sys
sys.path.append(os.path.join(os.path.dirname(__file__), '../ch02'))
sys.path.append(os.path.join(os.path.dirname(__file__), '../ch05'))

from math import sqrt

from program2_1 import Dvector
from program2_2 import Dmatrix
from program2_3 import input_vector, input_matrix
from program2_4 import inner_product
from program5_4 import matrix_vector_product

N = 4

def main():
    a = Dmatrix(1, N, 1, N) # 行列領域の確保
    x = Dvector(1, N)       # ベクトル領域の確保

    # ファイルのオープン
    with open("input_eigen.dat", "r") as fin:
        with open("result_eigen.dat", "w") as fout:
            input_matrix( a, 'A', fin, fout ) # 行列 A の入出力
            input_vector( x, 'x', fin, fout ) # ベクトル x の入出力
            power_method( a, x, fout )        # べき乗法


# べき乗法
def power_method(a: Dmatrix, x: Dvector, fout):
    k = 0 # 反復回数
    eps = 10.0 ** -8.0 # eps=10^{-8}とする

    v = Dvector(1, N) # ベクトル領域の確保

    while True:
        v       = matrix_vector_product(a, x)
        lambda_ = inner_product(v, x)
        v2      = inner_product(v,v)
        v2s     = sqrt(v2)
        for i in range(1, N+1):
            x[i] = v[i] / v2s
        k += 1
        if abs(v2 - lambda_ * lambda_) < eps:
            break
    fout.write(f"反復回数は {k}\n")
    fout.write("絶対値最大固有値 lambda は{:.6f}\n".format(lambda_))
    fout.write("これに対応する固有ベクトルは次の通りです\n")

    for i in range(1, N+1):
        fout.write("v[{}]={:.6f}\n".format(i, x[i]))


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