```
      begin A[J, I]:=A[I, J]; B[J, I]:=−B[I,
        J] end
    end READ and FORM HERMIX MATRIX;
      if AN=0 then go to LABEL 3;
      READ and FORM HERMITE MATRIX;
      HERMITE MATRIX PRINT;
LABEL 2: A plus Bi type JACOBI;
      EIGEN VALUE PRINT;
      UNITARY MATRIX PRINT;
      CRLF;
      go to LABEL 1;
LABEL 3:
end
```

6403. 行列の固有値と固有ベクトル

清水公子（東大物性研究所）

n次の行列 $A=(a_{ij})$ の固有値と固有ベクトルを power method で絶対値最大のものから m 個求めるプログラムである．

$$X=(1,00\cdots\cdots0)^T \qquad (1)$$

を与え，

$$Y\leftarrow AX,\ X\leftarrow Y \qquad (2)$$

を適当にくり返し，第一根 λ_1 を X, Y の比として求め，対応するベクトル U_1 も求める．次に λ_1 に対応する A^T の固有ベクトル V_1 を求め

$$U_1{}^TV_1=1$$

となるように規格化する．

$$A_2=A-\lambda_1U_1V_1{}^T$$

を新しい A と考えて上のプロセスをくり返し，第二根と，対応する固有ベクトルを求める．以下同様にして第 m 根まで求める．

（2）のくり返しの収束判定には

$$\sum|x_i-y_i|<\text{eps}$$

を使う．（x_i, y_i はベクトル X, Y の i 番目のエレメントである．）

固有ベクトルは長さが1に等しくなるように規格化されている．

データは次の順序に入れる．

n, m, eps, a_{11}, a_{12}, ……a_{1n}, a_{21}, ……a_{nm}

〔注〕 このプログラムは ISSP-ALGOL で通したものであるが，入出力の命令は投稿規定に従って変更した．

```
POWER METHOD:
  begin integer n, m; real eps;
    READ(n); READ(m); READ(eps);
  begin integer i, j, k, h;
    real w, SQ, delta, lambda, sigma;
    array x, y, u[1 : n], A[1 : n, 1 : n];
    switch sw 1:=normal, trans;
    switch sw 2:=result, prep;
    procedure INNERPRODUCT(p, a, b, r);
      integer p; real a, b, r;
      begin real s; s:=0;
      fof p:=1 step 1 until n do s:=s+a×b;
        r:=s
      end INNERPRODUCT;
start: for i:=1 step 1 until n do
      for j:=1 step 1 until n do READ(A[i, j]);
      h:=1;
      for i:=1 step 1 until m do
  begin
  st 1: x[1]:=1.0;
       for j:=2 step 1 until n do x[j]:=0;
  st 2: go to sw 1[h];
  normal: for j:=1 step 1 until n do
      INNERPRODUCT(k, A[j, k], x[k], y[j]);
      go to test;
  trans: for j:=1 step 1 until n do
      INNERPRODUCT(k, A[k, j], x[k], y[j]);
  test: if y[1]≠0 then
      begin lambda:=y[1]; w:=1.0/lambda;
        y[1]:=1.0; delta:=0;
        for j:=2 step 1 until n do
          begin y[j]:=y[j]×w;
                delta:=delta+abs(x[j]−y[j])
          end;
        if delta<eps then go to sw 2[h]
      end;
      for j:=1 step 1 until n do x[j]:=y[j];
      go to st 2;
  prep: INNERPRODUCT(j, u[j], y[j], sigma);
      lambda:=lambda/sigma;
      for j:=1 step 1 until n do
      for k:=1 step 1 until n do
        A[j, k]:=A[j, k]−lambda×u[j]×y[k];
      h:=1;
      go to st 1;
  result: sigma:=0;
      for j:=1 step 1 until n do
        sigma:=sigma+y[j]↑2;
      SQ:=sqrt(sigma); SQ:=1.0/SQ;
      for j:=1 step 1 until n do u[j]:
         =y[j]×SQ;
      CRLF; PRINTSTRING('i= '); PRINT(i);
        SPACE(3);
      PRINTSTRING('eigen value='');
      PRINT(lambda); CRLF;
      PRINTSTRING('eigen vector '); CRLF;
      for j:=1 step 1 until n do
        begin PRINT(u[j]); if (j÷6)×6=j then
          CRLF
        end;
      h:=2
    end i loop
  end
end
```