```
0001  //POWER & INVERSE POWER METHOD : Finding largest &
 smallest Eigenvalues
0002
0003  clc; clear; fd=%io(2); //File Descriptor(=6)
0004
0005  A=[2 -1 0; -1 2 -1; 0 -1 2]; //Input Data
0006  X=[1; 1; 1]; //Initial Eigenvector
0007  iter=1; //Set iteration number to 1
0008  maxerr=1e-2;    //Set tolerance error (max)
0009  err=1e+4;  //Set initial error (as large as possible)
0010  lam1=1e10;   //Set initial Eigen value (as large as
 possible)
0011
0012  choice=int(input("Enter the choice: (1 - Largest & 2 -
 Smallest ) "));
0013  if choice==1 then   //Finding largest Eigenvalue
0014      B=A;
0015  elseif choice==2 then    //Finding smallest Eigenvalue
0016      B=inv(A);
0017  else
0018      disp("Invalid Choice (Choose between 1 & 2)");
0019  end
0020
0021  while(err>maxerr)
0022      xold=X; //Preserve old Eigenvector
0023      Y=B*X;  //Compute new matrix
0024      eigval=max(abs(Y)); //Compute EigenValue (largest)
0025      eigvec=Y./eigval;   //Compute new Eigenvector
0026      X=eigvec;   //Store Eigenvector values
0027      err=abs(sum(xold-X));   //Compute error
0028      lam1=eigval;    //Update EigenValue
0029      iter=iter+1;    //Update iteration counter
0030  end
0031
0032  mfprintf(fd,"Method converge in %d iteration \n
\n",iter-1);
0033  if choice==1 then
0034      mfprintf(fd,"Greatest EigenValue = %5.5f\n",lam1);
0035  elseif choice==2 then
0036      mfprintf(fd,"Smallest EigenValue = %5.5f\n",1/lam1);
0037  end
0038  disp("The corresponding eigenvector is:",X);
0039
0040  //OUTPUT No 1:
```

```
0041  //Enter the choice: (1 - Largest & 2 - Smallest ) 1
0042  //
0043  //Method converge in 6 iteration
0044  //
0045  //Greatest EigenValue = 3.41667
0046  //
0047  //  "The corresponding eigenvector is:"
0048  //
0049  //   0.7073171
0050  //  -1.
0051  //   0.7073171
0052  //Output No 2 :
0053  //Enter the choice: (1 - Largest & 2 - Smallest ) 2
0054  //
0055  //Method converge in 4 iteration
0056  //
0057  //Smallest EigenValue = 0.58537
0058  //
0059  //  "The corresponding eigenvector is:"
0060  //
0061  //   0.7073171
0062  //   1.
0063  //   0.7073171
```