THEORY:

If A is a square matrix of size n and $\lambda_1, \lambda_2, \ldots \lambda_n$ are the eigen values of a matrix A satisfying the relation $|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|$, then λ_1 is called the dominant (absolutely largest) eigen value and the eigen vector corresponding to this eigen value λ_1 is known as dominant eigen vector.

Working procedure:

Start with a column vector $X^{(0)}$, which is as near the solution as possible and evaluate $AX^{(1)}$, which is written as $\lambda^{(1)} X^{(1)}$ after normalization. This gives the first approximation $\lambda^{(0)}$ to the eigen vector. Similarly, we evaluate $AX^{(1)} = \lambda^{(2)} X^{(2)}$ which gives the second approximation. We repeat this process till $|X^{(r)} - X^{(r-1)}|$ becomes negligible, that is, if $|X^{(r)} - X^{(r-1)}| < \varepsilon, \varepsilon > 0$. Then $\lambda^{(r)}$ will be the largest eigen-value and $X^{(r)}$, the corresponding eigen vector of A.

ALGORITHM:

1. Start
2. Input order of matrix A, n.
3. Input matrix A and column vector X
4. For i = 1 to n, loop.
   $Z_i = 0$
   For j = 1 to n, loop.
   $Z_i = Z_i + A_{ij} \times X_j$
   End loop
   End loop
5. $Z_{max} = |Z_1|$
6. For i = 2 to n, loop
   If $|Z_i| > Z_{max}$, then,
   $Z_{max} = |Z_i|$
   End if
   End loop

7. For $i = 1$ ton, loop.

$$z_i = \frac{z_i}{z_{max}}$$

End loop

8. For $i = 1$ to n, loop,

$E_i = ||z_i| - |x_i||$

End loop

9. $\epsilon_{max} = \epsilon_1$

10. For $i = 2$ ton, loop,

If $\epsilon_i > \epsilon_{max}$, then,

$\epsilon_{max} = \epsilon_i$

End if

End loop

11. For $i = 1$ ton, loop,

$x_i = z_i$

End loop

12. If $\epsilon_{max} > 0.001$, then

Goto 4

End if

13. For $i = 1$ ton, loop,

Display z_i

End loop

14. Stop

SOURCE CODE

```
#include<stdio.h>
#include<math.h>
int main(){
    int i,j,n;
    float A[40][40],x[40],z[40],e[40],zmax,emax;
    printf("Enter order of matrix:");
    scanf("%d",&n);
    printf("Enter matrix elements:\n");
    for(i=0;i<n;i++){
       for(j=0;j<n;j++){
          scanf("%f",&A[i][j]);
       }
    }
    printf("Enter the column vector:\n");
    for(i=0;i<n;i++){
       scanf("%f",&x[i]);
    }
    do{
       for(i=0;i<n;i++){
          z[i]=0;
          for(j=0;j<n;j++){
             z[i]=z[i]+A[i][j]*x[j];
          }
       }
       zmax=fabs(z[0]);
       for(i=1;i<n;i++){
          if(fabs(z[i])>zmax){
             zmax=fabs(z[i]);
          }
       }
```

```
for(i=0;i<n;i++){
        if z[i]=z[i]/zmax;
}
for(i=0;i<n;i++){
        e[i]=fabs(fabs(z[i])-fabs(x[i]));
}
emax=e[0];
for(i=1;i<n;i++){
        if(e[i]>emax){
            emax=e[i];
        }
}
for(i=0;i<n;i++){
    x[i]=z[i];
}
} while(emax>0.001);
printf(" The eigen value is %.4f\n",zmax);
printf(" The eigen vector is:\n");
for(i=0;i<n;i++){
    printf("%.4f\n",z[i]);
}
return 0;
}
```

①

Enter the order of matrix: 2

Enter matrix elements:

1 5

4 2

Enter the column vector:

2

3

The eigen value is 5.9985

The eigen vector is:

0.9996

1.0000

②

Enter the order of matrix: 3

Enter matrix elements:

1 5 4

4 8 7

2 1 0

Enter the column vector:

2

3

4

The eigen value is 11.4692

The eigen vector is:

0.5474

1.0000

0.1826

③

Enter the order of matrix : 4

Enter matrix elements:

2 4 5 7
0 1 6 9
5 1 1 2
2 5 7 8

Enter the column vector:

2
6
7
2

The eigen value is 16.5639.

The eigenvector is:

0.8403
0.7505
0.4467
1.0000

DISCUSSION:

We were assigned to write a program to find eigenvalue and eigenvector of a square matrix using power method. For that purpose, a program was written in C using Visual Studio Code, which was then compiled using GNU C Compiler (GCC). Then different test cases were given and the output obtained was expected which showed that the program worked.