سوال ۱: تابع Jacobi_eig را به صورت زیر پیاده‌سازی می‌کنیم:

```
function [jV, jD] = Jacobi_eig(A)
    tol = 0.0001;
    delta = tol*norm(A, 'fro');
    n = size(A, 1);
    jV = eye(n);
    jD = A;
    while (off(jD) > delta)
        for p=1:n-1
            for q=p+1:n
                [c, s] = symSchur2(jD, p, q);
                J = eye(n);
                J([p, q], [p, q]) = [c, s; -s, c];
                jV = jV*J;
                jD = J'*jD*J;
            end
        end
    end
    jD = diag(jD);
    for i=1:n-1
        for j=1:n-i
            if jD(j) > jD(j+1)
                temp = jD(j);
                jD(j) = jD(j+1);
                jD(j+1) = temp;
                temp = jV(:, j);
                jV(:, j) = jV(:, j+1);
                jV(:, j+1) = temp;
            end
        end
    end
    jD = diag(jD);
end
```

که در آن توابع off و symSchur2 به ترتیب به صورت زیر تعریف شده اند:

```
function s = off(A)
    s = norm(A, 'fro')^2;
    for i=1:min(size(A))
        s = s-A(i, i)^2;
    end
    s = sqrt(s);
end
function [c, s] = symSchur2(A, p, q)
    a = A([p, q], [p, q]);
    if (a(1, 2) == 0)
        c = 1;
        s = 0;
    else
        tau = (a(2, 2)-a(1, 1))/(2*a(1, 2));
        if (tau >= 0)
            t_min = 1/(tau+sqrt(1+tau^2));
        else
            t_min = 1/(tau-sqrt(1+tau^2));
        end
        c = 1/(sqrt(1+t_min^2));
        s = t_min*c;
    end
end
```

مثال ۱:

```
A =

    0.4456    0.4612    0.4142    0.6700
    0.4612    0.6797    0.5767    0.1932
    0.4142    0.5767    0.9597    0.5458
    0.6700    0.1932    0.5458    0.2551

jV =

    0.6071   -0.3869    0.5152    0.4652
   -0.2419   -0.6032   -0.5950    0.4729
    0.2100    0.6931   -0.2912    0.6250
   -0.7272    0.0778    0.5438    0.4115

jD =

   -0.3975         0         0         0
         0    0.2879         0         0
         0         0    0.3861         0
         0         0         0    2.0637

V =

   -0.6071    0.3869   -0.5152    0.4652
    0.2419    0.6032    0.5950    0.4729
   -0.2100   -0.6931    0.2912    0.6250
    0.7272   -0.0778   -0.5438    0.4115

D =

   -0.3975         0         0         0
         0    0.2879         0         0
         0         0    0.3861         0
         0         0         0    2.0637
```

مثال ۲:

A =

0.5060	0.7699	0.7535	0.8582	0.7406	0.2249	0.2999	0.1678
0.7699	0.2543	0.6438	0.4986	0.5296	0.4393	0.1402	0.3469
0.7535	0.6438	0.3517	0.6056	0.5770	0.3577	0.5731	0.1962
0.8582	0.4986	0.6056	0.5678	0.2726	0.3280	0.7221	0.8705
0.7406	0.5296	0.5770	0.2726	0.0119	0.3000	0.1573	0.3995
0.2249	0.4393	0.3577	0.3280	0.3000	0.6541	0.7575	0.7615
0.2999	0.1402	0.5731	0.7221	0.1573	0.7575	0.5383	0.9067
0.1678	0.3469	0.1962	0.8705	0.3995	0.7615	0.9067	0.8687

jV =

-0.4977	-0.2421	0.5302	0.1704	0.0244	-0.1155	-0.4867	0.3643
0.1683	-0.4545	-0.5906	0.2516	0.3047	0.2554	-0.3147	0.3098
-0.2751	0.4368	-0.4174	-0.3419	-0.4715	0.1040	-0.3006	0.3448
0.4043	0.3691	-0.0160	0.4253	-0.0029	-0.5863	-0.0133	0.4186
0.5433	0.0957	0.3486	-0.5646	0.2757	0.2071	-0.2609	0.2615
-0.0060	0.2794	0.2246	0.4075	-0.0509	0.6915	0.3372	0.3330
0.1936	-0.5650	0.0764	-0.1755	-0.5664	-0.0937	0.3750	0.3693
-0.3900	0.0359	-0.1336	-0.3069	0.5337	-0.1922	0.5012	0.4020

jD =

-0.8258	0	0	0	0	0	0	0
0	-0.5940	0	0	0	0	0	0
0	0	-0.3877	0	0	0	0	0
0	0	0	-0.2541	0	0	0	0
0	0	0	0	0.0678	0	0	0
0	0	0	0	0	0.3300	0	0
0	0	0	0	0	0	1.3299	0
0	0	0	0	0	0	0	4.0867

V =

-0.4978	0.2421	-0.5302	0.1705	-0.0244	0.1155	-0.4867	0.3643
0.1683	0.4545	0.5906	0.2516	-0.3047	-0.2554	-0.3147	0.3098
-0.2751	-0.4368	0.4174	-0.3419	0.4715	-0.1040	-0.3006	0.3448
0.4043	-0.3691	0.0160	0.4253	0.0029	0.5863	-0.0133	0.4186
0.5433	-0.0957	-0.3486	-0.5646	-0.2757	-0.2071	-0.2609	0.2615
-0.0060	-0.2794	-0.2246	0.4075	0.0509	-0.6915	0.3372	0.3330
0.1936	0.5650	-0.0764	-0.1755	0.5664	0.0937	0.3750	0.3693
-0.3900	-0.0359	0.1336	-0.3069	-0.5337	0.1922	0.5012	0.4020

D =

-0.8258	0	0	0	0	0	0	0
0	-0.5940	0	0	0	0	0	0
0	0	-0.3877	0	0	0	0	0
0	0	0	-0.2541	0	0	0	0
0	0	0	0	0.0678	0	0	0
0	0	0	0	0	0.3300	0	0
0	0	0	0	0	0	1.3299	0
0	0	0	0	0	0	0	4.0867

سوال ۲: تابع Jacobi_svd_2sided را به صورت زیر پیاده‌سازی می‌کنیم:

```
function [jU2, jS2, jV2] = Jacobi_svd_2sided(A)
    tol = 0.0001;
    delta = tol*norm(A, 'fro');
    [m, n] = size(A);
    jU2 = eye(m);
    jS2 = A;
    jV2 = eye(n);
    while (off(jS2) > delta)
        for p=1:min(m, n)-1
            for q=p+1:min(m, n)
                [c1, s1, c2, s2] = asymSchur2(jS2, p, q);
                J1 = eye(m);
                J1([p, q], [p, q]) = [c1, s1; -s1, c1];
                J2 = eye(n);
                J2([p, q], [p, q]) = [c2, s2; -s2, c2];
                jU2 = jU2*J1;
                jS2 = J1'*jS2*J2;
                jV2 = jV2*J2;
            end
        end
        if m < n
            for p=1:m
                for q=m+1:n
                    if jS2(p, p) == 0
                        c2 = 0;
                        s2 = 1;
                    else
                        t = -jS2(p, q)/jS2(p, p);
                        c2 = 1/sqrt(1+t^2);
                        s2 = t*c2;
                    end
                    [~, ~, c2, s2] = asymSchur2(jS2, p, q);
                    J2 = eye(n);
                    J2([p, q], [p, q]) = [c2, s2; -s2, c2];
                    jS2 = jS2*J2;
                    jV2 = jV2*J2;
                end
            end
        elseif m > n
            for p=n+1:m
                for q=1:n
                    if jS2(q, q) == 0
                        c1 = 0;
                        s1 = 1;
                    else
                        t = -jS2(p, q)/jS2(q, q);
                        c1 = 1/sqrt(1+t^2);
                        s1 = t*c1;
                    end
                    J1 = eye(m);
                    J1([q, p], [q, p]) = [c1, s1; -s1, c1];
                    jU2 = jU2*J1;
                    jS2 = J1'*jS2;
                end
            end
        end
    end
    for i=1:min(m, n)
        if jS2(i, i) < 0
```

```
            jS2(i, i) = -jS2(i, i);
            jU2(:, i) = -jU2(:, i);
        end
    end
    jS2 = diag(jS2);
    for i=1:min(m, n)-1
        for j=1:min(m, n)-i
            if jS2(j) < jS2(j+1)
                temp = jU2(:, j);
                jU2(:, j) = jU2(:, j+1);
                jU2(:, j+1) = temp;
                temp = jS2(j);
                jS2(j) = jS2(j+1);
                jS2(j+1) = temp;
                temp = jV2(:, j);
                jV2(:, j) = jV2(:, j+1);
                jV2(:, j+1) = temp;
            end
        end
    end
    temp = jS2;
    jS2 = zeros(m, n);
    for i=1:min(m, n)
        jS2(i, i) = temp(i);
    end
end
```

که در آن تابع asymSchur2 به صورت زیر تعریف شده است:

```
function [c1, s1, c2, s2] = asymSchur2(A, p, q)
    a = A([p, q], [p, q]);
    if (a(1, 2) == a(2, 1))
        c = 1;
        s = 0;
    else
        t = (a(2, 1)-a(1, 2))/(a(1, 1)+a(2, 2));
        c = 1/(sqrt(1+t^2));
        s = t*c;
    end
    temp = [c, s; -s, c]*a;
    [c2, s2] = symSchur2(temp, 1, 2);
    c1 = c * c2 + s * s2;
    s1 = c * s2 - s * c2;
end
```

مثال ۱:

```
B =

    0.0844    0.4314
    0.3998    0.9106
    0.2599    0.1818
    0.8001    0.2638

jU2 =

    0.2987   -0.3694    0.7697    0.4265
    0.7404   -0.4817   -0.4542   -0.1162
    0.2371    0.1460    0.4487   -0.8492
    0.5535    0.7811   -0.0000    0.2889

jS2 =

    1.2908         0
         0    0.5713
         0         0
         0         0

jV2 =

    0.6396    0.7687
    0.7687   -0.6396

U =

   -0.2987   -0.3694   -0.2745   -0.8360
   -0.7404   -0.4817    0.0289    0.4679
   -0.2371    0.1460    0.9183   -0.2814
   -0.5535    0.7811   -0.2837   -0.0542

S =

    1.2908         0
         0    0.5713
         0         0
         0         0

V =

   -0.6396    0.7687
   -0.7687   -0.6396
```

مثال ۲:

B =

0.7447	0.3685	0.9294	0.4468	0.8176	0.8116	0.8759	0.2077
0.1890	0.6256	0.7757	0.3063	0.7948	0.5328	0.5502	0.3012
0.6868	0.7802	0.4868	0.5085	0.6443	0.3507	0.6225	0.4709
0.1835	0.0811	0.4359	0.5108	0.3786	0.9390	0.5870	0.2305

jU2 =

0.6145	0.1752	-0.1577	-0.7529
0.4754	-0.1861	0.8433	0.1680
0.4921	-0.6267	-0.4869	0.3578
0.3929	0.7361	-0.1639	0.5262

jS2 =

3.1632	0	0	0	0	0	0	0
0	0.7682	0	0	0	0	0	0
0	0	0.4528	0	0	0	0	0
0	0	0	0.3996	0	0	0	0

jV2 =

0.3027	-0.2604	-0.7124	-0.4672	0.0151	0.1737	-0.2870	0.0439
0.2970	-0.6264	0.1684	0.3741	-0.2661	0.3564	-0.0037	-0.3915
0.4270	0.0445	0.4398	-0.4153	-0.5647	-0.2579	-0.1617	0.1957
0.2754	0.1023	-0.3167	0.4149	-0.0628	-0.7174	-0.1753	-0.3033
0.4255	-0.1690	0.3656	-0.1310	0.7785	-0.1267	-0.1306	-0.0179
0.4089	0.6697	-0.0073	0.2454	0.0002	0.4956	-0.2694	-0.0763
0.4226	0.1212	-0.1623	-0.0888	0.0032	0.0008	0.8781	-0.0376
0.1875	-0.1889	-0.1011	0.4604	0.0008	0.0005	-0.0003	0.8408

U =

-0.6145	0.1752	-0.1577	-0.7529
-0.4754	-0.1861	0.8433	0.1680
-0.4921	-0.6267	-0.4869	0.3578
-0.3929	0.7361	-0.1639	0.5262

S =

3.1632	0	0	0	0	0	0	0
0	0.7682	0	0	0	0	0	0
0	0	0.4528	0	0	0	0	0
0	0	0	0.3996	0	0	0	0

V =

-0.3027	-0.2604	-0.7124	-0.4672	-0.0116	0.1974	-0.2723	0.0386
-0.2970	-0.6264	0.1684	0.3741	-0.3458	0.2923	0.0192	-0.3817
-0.4270	0.0445	0.4398	-0.4153	-0.4796	-0.3658	-0.2238	0.1894
-0.2754	0.1023	-0.3167	0.4149	0.1076	-0.6890	-0.2247	-0.3232
-0.4255	-0.1690	0.3656	-0.1310	0.7916	0.0648	-0.0884	-0.0306
-0.4089	0.6697	-0.0073	0.2454	-0.1026	0.5065	-0.2269	-0.0740
-0.4226	0.1212	-0.1623	-0.0888	-0.0373	-0.0780	0.8746	-0.0106
-0.1875	-0.1889	-0.1011	0.4604	0.0123	-0.0129	-0.0269	0.8402

سوال ۳: تابع Jacobi_svd_1sided را به صورت زیر پیاده‌سازی می‌کنیم:

```
function [jU1, jS1, jV1] = Jacobi_svd_1sided(A)
    [m, n] = size(A);
    tol = 0.0001;
    if m <= n
        delta = tol * norm(A*A', 'fro');
        D = A';
        jV1 = eye(m);
        while (off(D'*D) > delta)
            for p=1:m-1
                for q=p+1:m
                    [c, s] = orthogonalization(D(:, p), D(:, q));
                    J = eye(m);
                    J([p, q], [p, q]) = [c, s; -s, c];
                    D = D*J;
                    jV1 = jV1*J;
                end
            end
        end
        jU1 = zeros(n, m);
        jS1 = zeros(n, m);
        for i = 1:m
            jU1(:, i) = D(:, i)/norm(D(:, i));
            jS1(i, i) = norm(D(:, i));
        end
        temp = jU1;
        jU1 = jV1;
        jS1 = jS1';
        jV1 = temp;
    else
        delta = tol * norm(A'*A, 'fro');
        D = A;
        jV1 = eye(n);
        while (off(D'*D) > delta)
            for p=1:n-1
                for q=p+1:n
                    [c, s] = orthogonalization(D(:, p), D(:, q));
                    J = eye(n);
                    J([p, q], [p, q]) = [c, s; -s, c];
                    D = D*J;
                    jV1 = jV1*J;
                end
            end
        end
        jU1 = zeros(m, n);
        jS1 = zeros(m, n);
        for i = 1:n
            jU1(:, i) = D(:, i)/norm(D(:, i));
            jS1(i, i) = norm(D(:, i));
        end
    end
        jS1 = diag(jS1);
    for i=1:min(m, n)-1
        for j=1:min(m, n)-i
            if jS1(j) < jS1(j+1)
                temp = jU1(:, j);
                jU1(:, j) = jU1(:, j+1);
                jU1(:, j+1) = temp;
                temp = jS1(j);
                jS1(j) = jS1(j+1);
```

```
                jS1(j+1) = temp;
                temp = jV1(:, j);
                jV1(:, j) = jV1(:, j+1);
                jV1(:, j+1) = temp;
            end
        end
    end
    temp = jS1;
    jS1 = zeros(m, n);
    for i=1:min(m, n)
        jS1(i, i) = temp(i);
    end
end
```

که در آن تابع orthogonalization به صورت زیر تعریف شده است:

```
function [c, s] = orthogonalization(x, y)
    if (norm(x) == norm(y))
        c = 1/sqrt(2);
        s = 1/sqrt(2);
    else
        t = 2*x'*y/(norm(y)^2-norm(x)^2);
        c = sqrt((1+1/sqrt(1+t^2))/2);
        s = sqrt((1-1/sqrt(1+t^2))/2);
    end
end
```

مثال ۱:

```
B =

    0.0844    0.4314
    0.3998    0.9106
    0.2599    0.1818
    0.8001    0.2638

jU1 =

    0.2987   -0.3695
    0.7404   -0.4817
    0.2371    0.1461
    0.5536    0.7811

jS1 =

    1.2908         0
         0    0.5713

jV1 =

    0.6397    0.7686
    0.7686   -0.6397

U =

   -0.2987   -0.3694   -0.2745   -0.8360
   -0.7404   -0.4817    0.0289    0.4679
   -0.2371    0.1460    0.9183   -0.2814
   -0.5535    0.7811   -0.2837   -0.0542

S =

    1.2908         0
         0    0.5713
         0         0
         0         0

V =

   -0.6396    0.7687
   -0.7687   -0.6396
```

مثال ۲:

```
B =

    0.7447    0.3685    0.9294    0.4468    0.8176    0.8116    0.8759    0.2077
    0.1890    0.6256    0.7757    0.3063    0.7948    0.5328    0.5502    0.3012
    0.6868    0.7802    0.4868    0.5085    0.6443    0.3507    0.6225    0.4709
    0.1835    0.0811    0.4359    0.5108    0.3786    0.9390    0.5870    0.2305

jU1 =

    0.6144   -0.1757    0.1577    0.7528
    0.4754    0.1862   -0.8433   -0.1678
    0.4921    0.6270    0.4869   -0.3573
    0.3929   -0.7358    0.1639   -0.5267

jS1 =

    3.1632         0         0         0
         0    0.7682         0         0
         0         0    0.4528         0
         0         0         0    0.3996

jV1 =

    0.3027    0.2603    0.7124    0.4677
    0.2970    0.6265   -0.1684   -0.3731
    0.4270   -0.0447   -0.4398    0.4154
    0.2754   -0.1021    0.3167   -0.4149
    0.4255    0.1689   -0.3656    0.1314
    0.4089   -0.6696    0.0073   -0.2461
    0.4226   -0.1212    0.1623    0.0888
    0.1875    0.1891    0.1011   -0.4600

U =

   -0.6145    0.1752   -0.1577   -0.7529
   -0.4754   -0.1861    0.8433    0.1680
   -0.4921   -0.6267   -0.4869    0.3578
   -0.3929    0.7361   -0.1639    0.5262

S =

    3.1632         0         0         0         0         0         0         0
         0    0.7682         0         0         0         0         0         0
         0         0    0.4528         0         0         0         0         0
         0         0         0    0.3996         0         0         0         0

V =

   -0.3027   -0.2604   -0.7124   -0.4672   -0.0116    0.1974   -0.2723    0.0386
   -0.2970   -0.6264    0.1684    0.3741   -0.3458    0.2923    0.0192   -0.3817
   -0.4270    0.0445    0.4398   -0.4153   -0.4796   -0.3658   -0.2238    0.1894
   -0.2754    0.1023   -0.3167    0.4149    0.1076   -0.6890   -0.2247   -0.3232
   -0.4255   -0.1690    0.3656   -0.1310    0.7916    0.0648   -0.0884   -0.0306
   -0.4089    0.6697   -0.0073    0.2454   -0.1026    0.5065   -0.2269   -0.0740
   -0.4226    0.1212   -0.1623   -0.0888   -0.0373   -0.0780    0.8746   -0.0106
   -0.1875   -0.1889   -0.1011    0.4604    0.0123   -0.0129   -0.0269    0.8402
```