# Test 1

Controllable [A,B] with SISO, refers from textbook page 257.

```
A1=[0 0 0;0 1 0;0 0 2]
```

```
A1 = 3×3
     0     0     0
     0     1     0
     0     0     2
```

```
B1=[1;1;1]
```

```
B1 = 3×1
     1
     1
     1
```

```
P1=[-1,-1,-1]
```

```
P1 = 1×3
    -1    -1    -1
```

```
K1 = myPlace(A1,B1,P1)
```

```
K1 = 1×3
    0.5000   -8.0000   13.5000
```

Controllable [A,B], there always exist 1 solution for K, so we can compare the result with the answer provided by the book:

K1 = [0.5, -8, 13.5]

# Test 2

Uncontrollable [A,B] with SISO, refers from textbook page 267.

```
A2=[0 1 0; 0 0 1; -6 -11 -6];
B2=[0;1;-3];
P2=[-2+2i,-2-2i];
%Compute K
K2 = myPlace(A2,B2,P2)
```

```
K2 = 1×3
    3.8571   -2.2143   -1.0714
```

```
%examine whether the poles are on the right points
%if the output contains [-2+2i,-2-2i],
%then it is successful.
uncontrollable_SISO_poles = eig(A2-B2*K2);
disp('Test 2 modifiled poles lise on:');
```

```
Test 2 modifiled poles lise on:
```

```
disp(uncontrollable_SISO_poles);
```

```
   -2.0000 + 2.0000i
   -2.0000 - 2.0000i
   -3.0000 + 0.0000i
```

This test is also satisfied.

# Test 3

Controllable [A,B] with MIMO

This is quite difficult.

## Test 3-1

Controllable [A,B], with 2 inputs, with [A, b1] is uncontrollable,

```
A31=[0 1 0; 0 0 1; -6 -11 -6];
B31=[0 2;1 1;-3 -1];
disp("rank of [A, b1]:")
```

```
rank of [A, b1]:
```

```
disp(rank(ctrb(A31,B31(:,1:1)))) % [A, b1] is uncontrollable
```

```
     2
```

```
disp("rank of [A, B]:")
```

```
rank of [A, B]:
```

```
disp(rank(ctrb(A31,B31))) %[A,B] is controllable
```

```
     3
```

```
P31 = [-2 -1+i, -1-i];
K31 = myPlace(A31,B31,P31)
```

```
K31 = 2×3 complex
  -0.0000 - 1.0000i   0.0000 + 0.5000i   0.0000 + 0.5000i
  -0.6667 + 0.3333i  -1.0000 + 0.5000i  -0.3333 + 0.1667i
```

```
% Check poles
disp('Test 3-1 modifiled poles lies on:')
```

```
Test 3-1 modifiled poles lies on:
```

```
disp(eig(A31-B31*K31))
```

```
   -1.0000 + 1.0000i
   -1.0000 - 1.0000i
   -2.0000 + 0.0000i
```

Test result is pretty good.

## Test 3-2

2 inputs, with the first column making [A,b1] controllable.

```
A32 = [0 1 0;
    0 0 1;
    1 -2 -3]
```

```
A32 = 3×3
     0     1     0
     0     0     1
     1    -2    -3
```

```
B32 = [0 0; 3 1; 2 0]
```

```
B32 = 3×2
     0     0
     3     1
     2     0
```

```
%check rank of b1
disp("rank of [A, b1]:")
```

```
rank of [A, b1]:
```

```
disp(rank(ctrb(A32,B32(:,1:1))))
```

```
     3
```

```
P32=[-2 -1+i, -1-i]
```

```
P32 = 1×3 complex
  -2.0000 + 0.0000i  -1.0000 + 1.0000i  -1.0000 - 1.0000i
```

```
K32 = myPlace(A32,B32,P32)
```

```
K32 = 2×3
    0.4347    0.2848    0.0728
         0         0         0
```

```
%check poles
disp('Test 3-2 modifiled poles lies on:')
```

```
Test 3-2 modifiled poles lies on:
```

```
disp(eig(A32-B32*K32))
```

```
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
  -2.0000 + 0.0000i
```

Performs well.

## Test 3-3

3 inputs.

```
A33=[0 1 0; 0 0 1; -6 -11 -6];
```

```
B33=[0 2 1;1 1 0;-3 -1 1];
P33 = [-2 -1+i, -1-i];
K33 = myPlace(A33,B33,P33);
% Check poles
disp('Test 3-3 modifiled poles lies on:')
```

```
Test 3-3 modifiled poles lies on:
```

```
disp(eig(A33-B33*K33))
```

```
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
  -2.0000 - 0.0000i
```

Test well.

# Test 4

Uncontrollable multiple inputs.

```
A4=[0 1 0; 0 0 1; -6 -11 -6];
B4=[0 0;1 1;-3 -3];
%check the controllablility-uncontrollable.
disp('rank of controllibility matrix:')
```

```
rank of controllibility matrix:
```

```
disp(rank(ctrb(A4,B4)))
```

```
     2
```

```
P4=[-1,-2]% rank of the controllible part is 2, so we put 2 target poles.
```

```
P4 = 1×2
    -1    -2
```

```
K4 = myPlace(A4,B4,P4);
disp('Test 4 modified poles are on:')
```

```
Test 4 modified poles are on:
```

```
disp(eig(A4-B4*K4))
```

```
   -1.0000
   -2.0000
   -3.0000
```

Test results are pretty good.

# Test 5

Comparation with `place` in matlab:

Here, the programmer have encapsulated the functions into `myPlace`.

## Test 5-1 Unlimited Geometric Multiplicities

```
A51=[0 0 0;0 1 0;0 0 2];
B51=[1;1;1];
P51=[-1,-1,-1];
K511 = myPlace(A51,B51,P51)
```

```
K511 = 1×3
    0.5000   -8.0000   13.5000
```

```
%K512 = place(A51,B51,P51)%error occurs
```

uncomment the last comment, an error will occur.

## Test 5-2 Uncontrollable Controlling

```
A52=[0 1 0; 0 0 1; -6 -11 -6];
B52=[0 0;1 1;-3 -3];
%check the controllablility-uncontrollable.
disp('rank of controllibility matrix:')
```

```
rank of controllibility matrix:
```

```
disp(rank(ctrb(A52,B52)))
```

```
     2
```

```
% rank of the controllible part is 2, so we put 2 target poles.
P52=[-1,-2,-233]
```

```
P52 = 1×3
    -1    -2  -233
```

```
K521 = myPlace(A52,B52,P52);
disp('Test 5-1 modified poles are on:')
```

```
Test 5-1 modified poles are on:
```

```
disp(eig(A4-B4*K521))
```

```
   -1.0000
   -2.0000
   -3.0000
```

```
% K522 = place(A52,B52,P52)%Error occur.
```

Here the matrix is not controllable. The function will only choose the first r terms(r represents the rank of controlibility matrix) of poles to place.

uncomment the comment last line, an error will occur.

# Test 6

New Feature!

```
A6 = [0 1 1;-6 -8 2;0 0 3];
B6 = [0 1;1 0;0 1];
P6 = [-4 -5 -6];
K61 = myOrderedPlace(A6,B6,P6,[1,2]);
disp('The first K I get is:')
```

```
The first K I get is:
```

```
disp(K61);
```

```
   14.0000    1.0000         0
         0         0    9.0000
```

```
disp('The poles of using such K are:');
```

```
The poles of using such K are:
```

```
disp(eig(A6-B6*K61));
```

```
   -4.0000
   -5.0000
   -6.0000
```

```
K62 = myOrderedPlace(A6,B6,P6,[2,1]);
disp('The second K I get is:')
```

```
The second K I get is:
```

```
disp(K62);
```

```
         0         0         0
   -4.3846   -0.8077   14.3846
```

```
disp('The poles of using such K are:');
```

```
The poles of using such K are:
```

```
disp(eig(A6-B6*K62));
```

```
   -4.0000
   -5.0000
   -6.0000
```

```
```

This time, we got 2 Ks, and each one could successfully place the pole to the target position.

## Test 7

```
A7 = [0 1 1;-6 -8 2;0 0 3];
```

```
B7 = [0 1;1 0;0 1];
P7 = [-4 -5 -6];
K7 = myPlace(A7,B7,P7)
```

```
K7 = 2×3
   14.0000    1.0000         0
         0         0    9.0000
```

```
K7r = myRandomPlace(A7,B7,P7)
```

```
K7r = 2×3
         0         0         0
   -4.3846   -0.8077   14.3846
```

```
disp('The default K I get is:')
```

```
The default K I get is:
```

```
disp(K7);
```

```
   14.0000    1.0000         0
         0         0    9.0000
```

```
disp('The poles of using such K are:');
```

```
The poles of using such K are:
```

```
disp(eig(A7-B7*K7))
```

```
   -4.0000
   -5.0000
   -6.0000
```

```
disp('The random K I get is:')
```

```
The random K I get is:
```

```
disp(K7r);
```

```
         0         0         0
   -4.3846   -0.8077   14.3846
```

```
disp('The poles of using such K are:');
```

```
The poles of using such K are:
```

```
disp(eig(A7-B7*K7r))
```

```
   -4.0000
   -5.0000
   -6.0000
```

# Test 8

## Random Placement Test

In this scope, the programmer define a function named "random place", to place the poles.

```
A8=[
    0,1,0;
    0,0,1;
    0,0,0];
B8=[1,0,0;0,1,0;0,0,1];
```

Let's see the rank of (A,b1) (A,b1,b2) (A,B).

```
r1 = rank(ctrb(A8,B8(:,1:1)))
```

```
r1 = 1
```

```
r2 = rank(ctrb(A8,B8(:,1:2)))
```

```
r2 = 2
```

```
r3 = rank(ctrb(A8,B8(:,1:3)))
```

```
r3 = 3
```

```
P8=[-1,-2,-3];
K8=myAdvancedRandomPlace(A8,B8,P8)
```

```
K8 = 3×3
    1.8024    1.0793         0
    2.0000    1.1976         0
         0         0    3.0000
```

```
% examine the poles.
disp('poles:')
```

```
poles:
```

```
disp(eig(A8-B8*K8))
```

```
   -2.0000
   -1.0000
   -3.0000
```