Q1,

Yes, it is possible.

Example:

x = [3, 4]   y = [5, 1]

$||x||_1 = 7 > 6 = ||y||_1$;

$||x||_\infty = 4 < 5 = ||y||_\infty$;

Q2,

(a)

$$A = \begin{bmatrix} 3 & 0 \\ 3 & 0 \end{bmatrix}$$

$||A||_\infty = 3 < 6 = ||A||_1$;

(b)

$$B = \begin{bmatrix} 3 & 3 \\ 0 & 0 \end{bmatrix}$$

$||B||_\infty = 6 > 3 = ||B||_1$;

Q4,

(a)

Code:

```
A = zeros(13);
b = zeros(13, 1);
alpha = sqrt(2)/2;

A(1, 2) = 1;
A(1, 6) = -1;
b(1, 1) = 0;

A(2, 3) = 1;
b(2, 1) = 10;

A(3, 1) = alpha;
A(3, 4) = -1;
A(3, 5) = (-1) * alpha;
b(3, 1) = 0;

A(4, 1) = alpha;
A(4, 3) = 1;
A(4, 5) = alpha;
b(4, 1) = 0;

A(5, 4) = 1;
A(5, 8) = -1;
b(5, 1) = 0;

A(6, 7) = 1;
b(6, 1) = 0;

A(7, 5) = alpha;
A(7, 6) = 1;
A(7, 9) = (-1) * alpha;
A(7, 10) = -1;
b(7, 1) = 0;

A(8, 5) = alpha;
A(8, 7) = 1;
A(8, 9) = alpha;
b(8, 1) = 15;

A(9, 10) = 1;
A(9, 13) = -1;
b(9, 1) = 0;

A(10, 11) = 1;
b(10, 1) = 20;

A(11, 8) = 1;
A(11, 9) = alpha;
A(11, 12) = (-1) * alpha;
b(11, 1) = 0;

A(12, 9) = alpha;
A(12, 11) = 1;
```

```
A(12, 12) = alpha;
b(12, 1) = 0;

A(13, 12) = alpha;
A(13, 13) = 1;
b(13, 1) = 0;

f = A\b;
fprintf('          f   \n');
fprintf('    --------------------\n');
fprintf('%26.15d\n', f);
```

The matrix A:

Columns 1 through 11

0	1.0000	0	0	0	-1.0000	0	0	0	0	0
0	0	1.0000	0	0	0	0	0	0	0	0
0.7071	0	0	-1.0000	-0.7071	0	0	0	0	0	0
0.7071	0	1.0000	0	0.7071	0	0	0	0	0	0
0	0	0	1.0000	0	0	0	-1.0000	0	0	0
0	0	0	0	0	0	1.0000	0	0	0	0
0	0	0	0	0.7071	1.0000	0	0	-0.7071	-1.0000	0
0	0	0	0	0.7071	0	1.0000	0	0.7071	0	0
0	0	0	0	0	0	0	0	0	1.0000	0
0	0	0	0	0	0	0	0	0	0	1.0000
0	0	0	0	0	0	0	1.0000	0.7071	0	0
0	0	0	0	0	0	0	0	0.7071	0	1.0000
0	0	0	0	0	0	0	0	0	0	0

Columns 12 through 13

0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	-1.0000
0	0
-0.7071	0
0.7071	0

0.7071   1.0000

Result:

f

-2.828427124746190e+01
000000000000020
000000000000010
-3.000000000000000e+01
1.414213562373095e+01
000000000000020
000000000000000
-3.000000000000000e+01
7.071067811865476e+00
2.500000000000000e+01
000000000000020
-3.535533905932737e+01
2.500000000000000e+01

Q4(b)

Code:

```
r = b - A * f;
delta_b = A * f - b;
relativeResidual = norm(r, 1) / norm(b, 1);
boundOfRelativeError = cond(A, 1) * (norm(delta_b, 1) / norm(b, 1));
fprintf('The upper bound of relative error is %15.15d, and the relative residual is
%15.15d.\n', boundOfRelativeError, relativeResidual);
```

Result:

The upper bound of relative error is 5.244311207218988e-15, and the relative residual is 2.368475785867001e-16.

Q5,

question 1: The maximum n is 12 before the relative error greater than or equals to 1.

Support:

Code:

```
relativeError = 0;
n = 0;
H = 0;
x = 0;
x_hat = 0;
fprintf('n          ||x - x_hat||/||x||\n');
fprintf('-------------------------------\n');
while relativeError < 1
    n = n + 1;
```

```
    H = hilb(n);
    x = ones(n, 1);
    b = H * x;
    x_hat = H \ b;
    relativeError = norm(x - x_hat, Inf) / norm(x, Inf);
    fprintf('%2.3d %26.15d\n', n, relativeError);
end
fprintf('--------------------------------\n');
```

Result:

```
n        ||x - x_hat||/||x||
-----------------------------------------
001          000000000000000
002      7.771561172376096e-16
003      1.110223024625157e-16
004      2.325917236589703e-13
005      4.896305583201865e-12
006      1.156209683550458e-09
007      1.269028304662356e-08
008      3.336011955568097e-07
009      1.912674193871311e-05
010      1.196518824667536e-03
011      4.839864875607969e-02
012      3.262108108653414e-01
013      2.329730937495737e+01
-------------------------------------------
```

We can see when n = 13, the relative error is greater than 1.

Q5 question2:

n has a linear relationship with the log of the condition number of hilb(n), more info following.

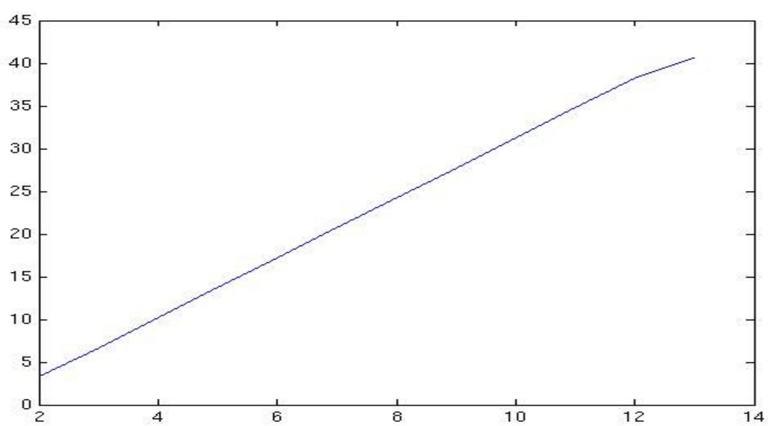
Code:

```
fprintf('n     m=log(cond(hilb(n),Inf))    m/n \n');
fprintf('---------------------------------------------\n');
x = [];
y = [];
for n = 2:13
    con = log(cond(hilb(n), Inf));
    div = con/n;
    fprintf('%2.2d %15.3d %26.3d\n', n, con, div);
    x = [x, n];
    y = [y, con];
end
fprintf('---------------------------------------------\n');
plot(x, y);
```

Result:

n	m=log(cond(hilb(n),Inf))	m/n
02	3.296e+00	1.648e+00
03	6.617e+00	2.206e+00
04	1.025e+01	2.563e+00
05	1.376e+01	2.752e+00
06	1.719e+01	2.864e+00
07	2.071e+01	2.958e+00
08	2.425e+01	3.031e+00
09	2.773e+01	3.081e+00
10	3.120e+01	3.120e+00
11	3.475e+01	3.159e+00
12	3.818e+01	3.181e+00
13	4.060e+01	3.123e+00

Plot figure:



As we can see from the figure and the printed result, n has a linear relationship with the log of the condition number of hilb(n):
=> k * n = log(cond(hilb(n))) (k is around 3) ← got this from m/n in the printed result.

Q5 question3:

As condition number of hilb(n) turns larger, the number of correct digits turns smaller but very slowly. (shown in the figure following)

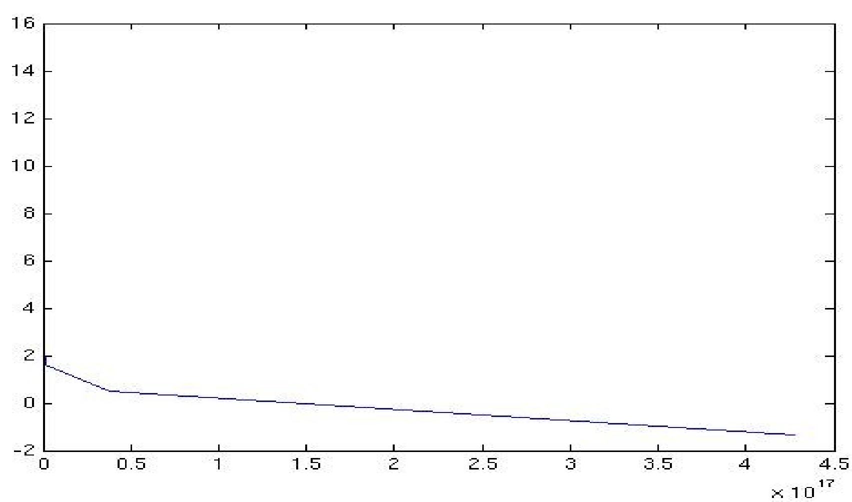
Code:
```
x_axis = [];
y_axis = [];
for n = 2:13
    H = hilb(n);
```

```
    x = ones(n);
    b = H * x;
    x_hat = H \ b;
    con = cond(H, Inf);
    con_log10 = log(con);
    logr10 = -log10(norm(x - x_hat, Inf)/norm(x, Inf));
    multiply = con_log10 * logr10;
    x_axis = [x_axis, con];
    y_axis = [y_axis, logr10];
    fprintf('%2.2d %15.3d %26.3d %26.3d %26.3d\n', n, con, con_log10, logr10,
multiply);
end
plot(x_axis, y_axis);
```

Printed result:

n	condition	correct digits
02	2.700e+01	1.511e+01
03	7.480e+02	1.595e+01
04	2.837e+04	1.263e+01
05	9.437e+05	1.131e+01
06	2.907e+07	8.937e+00
07	9.852e+08	7.897e+00
08	3.387e+10	6.477e+00
09	1.100e+12	4.767e+00
10	3.535e+13	2.991e+00
11	1.230e+15	1.587e+00
12	3.798e+16	4.865e-01
13	4.276e+17	-1.367e+00

If I put the condition number of hilb(n) as x axis, and put -log10(|| x - x_hat || / || x ||) as y axis, I will get a figure below,



So, we can see -log10(|| x - x_hat || / || x ||) is turning smaller very slowly.