1.(15pts.)

```matlab
function root = Bisection(a,b)
tol=1/8;

while ((b-a)>tol)
    m=a+(b-a)/2;
    f_a=3*a^3-5*a^2-4*a+4;
    f_m=3*m^3-5*m^2-4*m+4;
    if (sign(f_a)==sign(f_m))
        a=m;
    else b=m;
    end
end
root=m;
fprintf('the root is found in the interval [ %g, %g],

Output:
>> root=Bisection(0,1)
the root is found in the interval [ 0.625, 0.75]
root =
0.62500000000000

Achieving an error tolerance of $10^{-6}$ requires \[ \log_2 \left( \frac{b-a}{10^{-6}} \right) \] = 20 iterations.

2. (20pts.)
Firstly, we simply plot $\exp(x)$ Vs $100x^2$:
```matlab
x=-10:.01:10;
y1=exp(x);
y2=100*(x.^2);
plot(x,y1,'r-',x,y2,'b-');

We can learn from the plot that probably there are 3 intersection points.
To verify this, we got $f(-1) < 0$, $f(0) > 0$, $f(1) < 0$, $f(10)>0$. 3 sign changes means there are at least 3 solutions. Plus the 3rd order derivative of $\exp(x)-100x^2$ is positive whatever x is, so there are at most 3 roots.
We implement the `Bisection.m` again for the following three intervals (-0.2, 0), (0, 0.2), (9, 10).

```matlab
function root = Bisection(a,b)
    tol=1e-3;
    while ((b-a)>tol)
        m=a+(b-a)/2;
        f_a=exp(a)-100*(a^2);
        f_m=exp(m)-100*(m^2);
        if (sign(f_a)==sign(f_m))
            a=m;
        else
            b=m;
        end
    end
    root=m;
    fprintf('the root is found in the interval [ %g, %g]
    ans =
    >> Bisection(-0.2,0)
    the root is found in the interval [-0.0960938, -0.0953125]
    ans =
    -0.0961
    >> Bisection(0,0.2)
    the root is found in the interval [0.104688, 0.105469]
    ans =
    0.1055
    >> Bisection(9,10)
    the root is found in the interval [9.99902, 10]
    ans =
    9.9990

3. (15pts.)
    x=0.7;
    for i=1:4
        x=x-((3*x^3-5*x^2-4*x+4)/(9*x^2-10*x-4));
    end

    1st iteration: x = 0.6665
    2nd iteration: x = 0.6667
    3rd iteration: x = 0.6667
    4th iteration: x = 0.6667

4. (15pt.)
    1/x – c = 0;
    By Newton’s method,
Thus \( x_{n+1} = \frac{\frac{1}{c} - x_n}{\frac{1}{c} - \frac{1}{c^2} x_n} = x_n (2 - cx_n). \)

If \( x_n < \frac{1}{c} \), then \( x_{n+1} = x_n (2 - cx_n) > x_n (2 - c(1/c)) = x_n. \)

Therefore, for \( n \geq 1 \), \( x \) is an increasing sequence which converges to \( \frac{1}{c} \).

5. (15pt.)

\[
x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n) = x_n - \frac{x_n - x_{n-1}}{\left( \frac{1}{x_n} - c \right) - \left( \frac{1}{x_{n-1}} - c \right)} \left( \frac{1}{x_n} - c \right) = x_n + x_{n-1} - cx_n x_{n-1}.
\]

\[
x(0) = 0.1;
\]
\[
x(1) = 0.2;
\]
\[
x(2) = 0.16;
\]
\[
x(3) = 0.136;
\]
\[
x(4) = 0.1437;
\]
\[
x(5) = 0.1429; \text{ fourth iteration}
\]

6. (20pt.)

Script:
\[
x(1) = 0.8; \\
y(1) = 0.8;
\]
\[
\text{for } i=1:2 \\
\quad f1=x(i)^4+x(i)*y(i)^2+y(i)^4-1; \\
\quad f2=x(i)^2+x(i)*y(i)-y(i)^2/4-1; \\
\quad f1x=4*x(i)^3+y(i)^2; \\
\quad f1y=2*x(i)*y(i)+4*y(i)^3; \\
\quad f2x=2*x(i)+y(i); \\
\quad f2y=x(i)-y(i)/2; \\
\quad h=(-f1*f2y+f2*f1y)/(f1x*f2y-f1y*f2x); \\
\quad k=(f1*f2x-f2*f1x)/(f1x*f2y-f1y*f2x); \\
\quad x(i+1)=x(i)+h; \\
\quad y(i+1)=y(i)+k;
\]
\end{verbatim}

Finally we get:
\[
\begin{pmatrix}
  x_1 \\
  y_1
\end{pmatrix} = \begin{pmatrix}
  0.7614 \\
  0.7317
\end{pmatrix}, \quad \begin{pmatrix}
  x_2 \\
  y_2
\end{pmatrix} = \begin{pmatrix}
  0.7622 \\
  0.7197
\end{pmatrix}.
\]