Problem 1

The weak form is

$$\int_0^1 (1+x)\frac{du}{dx}\frac{dv}{dx}dx = \int_0^1 v dx.$$

The piecewise linear basis functions are described on pages 17 and 19 of the lecture 4 notes.

Part a

$$\begin{aligned}
\mathbf{K}_{11} &= \int_0^1 (1+x)(\phi_1')^2 dx \\
&= \int_0^{\frac{1}{2}} (1+x)(2)^2 dx + \int_{\frac{1}{2}}^1 (1+x)(-2)^2 dx \\
&= 4\int_0^1 (1+x)dx = 6 \\
\mathbf{F}_{11} &= \int_0^1 \phi_1 dx \\
&= \int_0^{\frac{1}{2}} 2x dx + \int_{\frac{1}{2}}^1 2(1-x)dx \\
&= x^2\Big|_0^{\frac{1}{2}} + 2x - x^2\Big|_{\frac{1}{2}}^1 \\
&= \frac{1}{4} + 1 - (1 - \frac{1}{4}) = \frac{1}{2} \\
\implies \alpha &= \frac{1}{12}
\end{aligned}$$

Part b

The vector **F** is easy to calculate. All of its entries have to be the same since the basis functions are all translations of the first one (the areas they bound are equal). **F** is a vector with all entries equal to:

$$\int_0^{0.25} 4x dx + \int_{0.25}^{0.5} 4(0.5-x)dx = 0.25$$

We again compute the entry k_{ij} of the stiffness matrix as $k_{ij} = \int_0^1 (1+x)\phi_i'\phi_j' dx$. We have to break these integrals down into a sum of two integrals to deal with each piece of the basis function.

$$k_{11} = \int_0^{0.25} (1+x)(4)^2 dx + \int_{0.25}^{0.5} (1+x)(-4)^2 dx = 16\int_0^{0.5} (1+x)dx = 16*(0.28125+0.34375) = 10$$

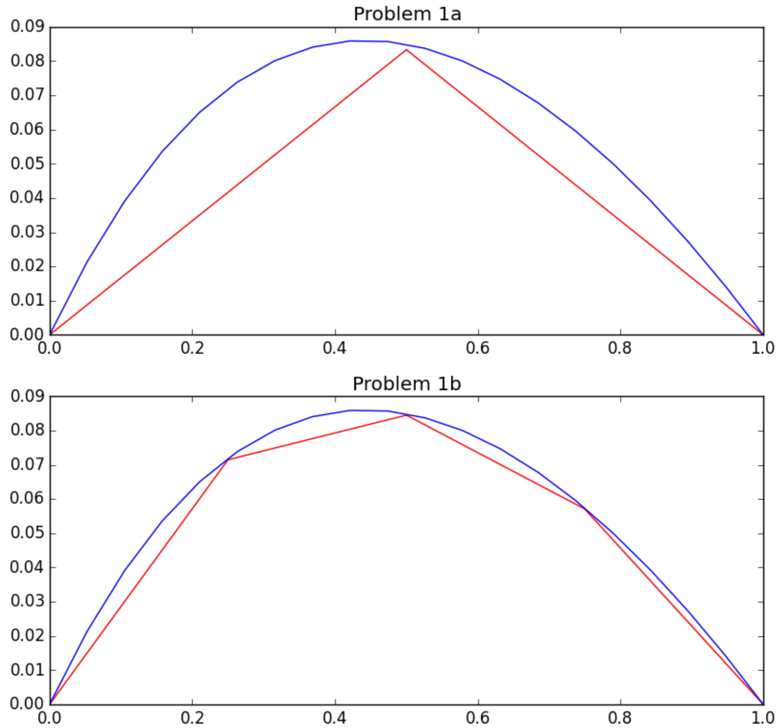
$$k_{12} = \int_0^{0.25} (1+x)(4)(0)dx + \int_{0.25}^{0.5} (1+x)(4)(-4)dx = -16(0.34375) = -5.5$$

$$k_{13} = \int_0^{0.25} (1+x)(4)(0)dx + \int_{0.25}^{0.5} (1+x)(-4)(0)dx = 0$$

$$k_{22} = \int_{0.25}^{0.5} (1+x)(4)^2 dx + \int_{0.5}^{0.75} (1+x)(-4)^2 dx = 12$$

$$k_{23} = \int_{0.25}^{0.5} (1+x)(4)(0)dx + \int_{0.5}^{0.75} (1+x)(-4)(4)dx = -6.5$$

$$k_{33} = \int_{0.5}^{1.0} (1+x)(16)dx = 14$$




```
from numpy import log, linspace, array
from numpy.linalg import solve
import matplotlib.pyplot as plt

# Part A
alpha = 1.0/12.0
el1 = lambda x: 2.0*x
el2 = lambda x: 2.0*(1.0 - x)
u = lambda x: log(1+x)/(log(2))-x

x = linspace(0, 1, 20)
x1 = linspace(0, 0.5, 10)
x2 = linspace(0.5, 1, 10)

plt.subplot(2, 1, 1)
plt.plot(x1, alpha*el1(x1),'r', x2, alpha*el2(x2), 'r', x, u(x), 'b')
plt.title('Problem 1a')

# Part B
K = array([[10, -5.5, 0],[-5.5, 12, -6.5],[0, -6.5, 14]])
f = 0.25
F = array([f, f, f])
alpha =  solve(K,F)

el1 = lambda x: alpha[0]*4*x
el2 = lambda x: alpha[1]*4*(x - 0.25) + alpha[0]*4*(0.5 - x)
el3 = lambda x: alpha[1]*4*(0.75 - x) + alpha[2]*4*(x - 0.5)
el4 = lambda x: alpha[2]*4*(1 - x)
u   = lambda x: log(1+x)/(log(2))-x
```

```
x  = linspace(0,1,20)
x1 = linspace(0, 0.25, 10)
x2 = linspace(0.25, 0.5, 10)
x3 = linspace(0.5, 0.75, 10)
x4 = linspace(0.75, 1, 10)

plt.subplot(2, 1, 2)
plt.plot(x1, el1(x1), 'r', x2, el2(x2), 'r', x3, el3(x3), 'r',\
        x4, el4(x4), 'r', x, u(x), 'b')
plt.title('Problem 1b')
plt.show()
```

Problem 2

For the cubic master element we need four nodes over an interval, and for convenience we use $[-1, 1]$. This gives us the nodes: $\xi_1 = -1,\ \xi_2 = -\frac{1}{3},\ \xi_3 = \frac{1}{3},\ \xi_4 = 1$. Then, the Lagrange shape functions are given by:

$$f(\xi, \xi_i) = \frac{(\xi - \xi_1)...(\xi - \xi_{i-1})(\xi - \xi_{i+1})...(\xi - \xi_n)}{(\xi_i - \xi_1)...(\xi_i - \xi_{i-1})(\xi_i - \xi_{i+1})...(\xi_i - \xi_n)}$$

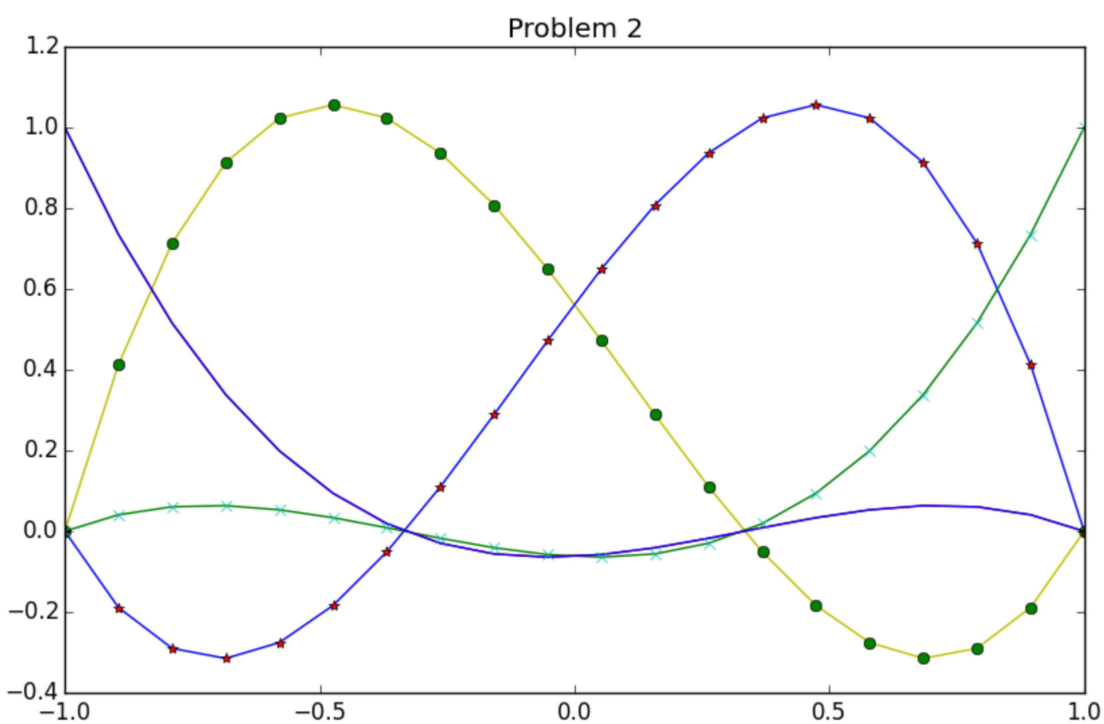
$$\psi_i = f(\xi, \xi_i), \quad n = 4$$

$$\psi_1 = -\frac{9}{16}x^3 + \frac{9}{16}x^2 + \frac{x}{16} - \frac{1}{16}$$

$$\psi_2 = \frac{27}{16}x^3 - \frac{9}{16}x^2 - \frac{27}{16}x + \frac{9}{16}$$

$$\psi_3 = -\frac{27}{16}x^3 - \frac{9}{16}x^2 + \frac{27}{16}x + \frac{9}{16}$$

$$\psi_4 = \frac{9}{16}x^3 + \frac{9}{16}x^2 - \frac{x}{16} - \frac{1}{16}$$



Problem 3

Below I output the results of the my code for $k = 3$, and then print out the decimal values of the fractions which appear as the coefficients for you to compare with Problem 2. The numbers are the same.

```
psi = lagrange_poly(3);

psi(1)
ans =

    fun: [-0.5625 0.5625 0.0625 -0.0625]
    der: [-1.6875 1.1250 0.0625]

psi(2)
ans =

    fun: [1.6875 -0.5625 -1.6875 0.5625]
    der: [5.0625 -1.1250 -1.6875]

psi(3)
ans =

    fun: [-1.6875 -0.5625 1.6875 0.5625]
    der: [-5.0625 -1.1250 1.6875]

psi(4)
ans =

    fun: [0.5625 0.5625 -0.0625 -0.0625]
    der: [1.6875 1.1250 -0.0625]


[9/16, 27/16, 1/16]
ans =

    0.5625    1.6875    0.0625
```

```
function psi = lagrange_poly(k)
%{
    k is the degree of the basis functions.
    First, set up the xi and y coordinates for the polyfit.
    Then, polyfit the coordinates for each set of y coordinates
    and this is a psi function. Use polyder to get the
    derivtives.
%}

xi = linspace(-1, 1, k+1);
for func=1:k+1
    y = zeros(1,k+1);
    y(func) = 1;
    F = polyfit(xi, y, k);
    psi(func).fun = F;
    psi(func).der = polyder(F);

end
```