Question 1.

a.

```
function Euler(m,c,g,t0,v0,tn,n)
% print headings and initial conditions
fprintf('values of t approximations v(t)\n')
fprintf('%8.3f',t0),fprintf('%19.4f\n',v0)
% compute step size h
h=(tn-t0)/n;
% set t,v to the initial values
t=t0;
v=v0;
% compute v(t) over n time steps using Euler's method
for i=1:n
    v=v+(g-c/m*v)*h;
    t=t+h;
    fprintf('%8.3f',t),fprintf('%19.4f\n',v)
end
```

b.

```
>> Euler (82.6,12.5,9.81,0,0,12,20)
values of t approximations v(t)
   0.000          0.0000
   0.600          5.8860
   1.200         11.2376
   1.800         16.1032
   2.400         20.5270
   3.000         24.5492
   3.600         28.2062
   4.200         31.5311
   4.800         34.5541
   5.400         37.3026
   6.000         39.8016
   6.600         42.0736
   7.200         44.1394
   7.800         46.0176
   8.400         47.7252
   9.000         49.2778
   9.600         50.6894
  10.200         51.9729
  10.800         53.1398
  11.400         54.2008
  12.000         55.1654
```

c.

```
>>Euler (82.6,12.5,8.83,0,0,12,20)
values of t approximations v(t)
   0.000        0.0000
   0.600        5.2980
   1.200       10.1149
   1.800       14.4945
   2.400       18.4764
   3.000       22.0968
   3.600       25.3884
   4.200       28.3812
   4.800       31.1022
   5.400       33.5761
   6.000       35.8255
   6.600       37.8705
   7.200       39.7299
   7.800       41.4205
   8.400       42.9576
   9.000       44.3551
   9.600       45.6257
  10.200        46.7809
  10.800        47.8312
  11.400        48.7862
  12.000        49.6545
```

d.

```
>> m=82.6,c=12.5,t=12,g=9.81

m = 82.6000
c = 12.5000
t = 12
g = 9.8100
>> v = ((g*m)/c)*(1-(exp(-(c*t)/m)))
v = 54.2789
>> x = v - 55.1654
x = -0.8865
>> y = abs(x)
y = 0.8865
>> e = y/v
e = 0.0163 (1.63%)
```

Question 2.

a.

```
function Euler2(m,k,g,t0,v0,tn,n)
% print headings and initial conditions
fprintf('values of t approximations v(t)\n')
fprintf('%8.3f',t0),fprintf('%19.4f\n',v0)
% compute step size h
h=(tn-t0)/n;
% set t,v to the initial values
t=t0;
v=v0;
% compute v(t) over n time steps using Euler's method
for i=1:n
    v=v+(g-k/m*(v^2))*h;
    t=t+h;
    fprintf('%8.3f',t),fprintf('%19.4f\n',v)
end
```

b.

```
>> diary filename
>> Euler2(82.6,0.234,9.81,0,0,12,20)
values of t approximations v(t)
   0.000         0.0000
   0.600         5.8860
   1.200        11.7131
   1.800        17.3659
   2.400        22.7393
   3.000        27.7464
   3.600        32.3238
   4.200        36.4339
   4.800        40.0636
   5.400        43.2213
   6.000        45.9320
   6.600        48.2319
   7.200        50.1638
   7.800        51.7725
   8.400        53.1025
   9.000        54.1954
   9.600        55.0889
  10.200        55.8165
  10.800        56.4070
  11.400        56.8848
  12.000        57.2706
```

c.

Exact solution using calculator = 56.73110

Error = 57.2706 – 56.73110
= 0.5395
Relative Error = 0.5395/56.73110
= 0.0095 (0.95%)

Question 3.

Method 1:

```
import math
def main():
   i = 1
   j = 1

   for j in range(8):
      value = 0
      temp = 0
      print ("Using",j,"Terms", end=" ")
      for i in range(j):
         temp = ((-1)**i)*(2.75**i)/math.factorial(i)
         value = value + temp
      print ("Approximation=",value, end=" ")
      error = 0.06392786 - value
      abserror = abs(error)
      relative_error = (abserror/0.06392786)
      print ("Relative Error=",relative_error)

main()
```

Output in Terminal:
Using 1 Terms, Approximation= 1.0 , Relative Error= 14.642632179459785
Using 2 Terms, Approximation= -1.75 , Relative Error= 28.37460631405462
Using 3 Terms, Approximation= 2.03125 , Relative Error= 30.774096614527686
Using 4 Terms, Approximation= -1.4348958333333335 , Relative Error= 23.445547736672765
Using 5 Terms, Approximation= 0.948079427083333 , Relative Error= 13.830457754777543
Using 6 Terms, Approximation= -0.36255696614583366 , Relative Error= 6.671345265520129
Using 7 Terms, Approximation= 0.23815138075086772 , Relative Error= 2.725314452116303

Method 2:

```
import math

def main():
  i = 0

  for j in range(7):
    x = 0
    temp = 0
    value = 0
    x = j + 1
    print ("Using",x,"Terms", end=" ")
    for i in range(x):
      temp = (2.75**i)/math.factorial(i)
      value = value + temp
    value = 1/value
    print ("Approximation=",value, end=" ")
    error = 0.06392786 - value
    abserror = abs(error)
    relative_error = (abserror/0.06392786)
    print ("Relative Error=",relative_error)

main()
```

Output in Terminal:

```
Using 1 Terms Approximation= 1.0 Relative Error= 14.642632179459785
Using 2 Terms Approximation= 0.26666666666666666 Relative Error= 3.1713685811892756
Using 3 Terms Approximation= 0.13278008298755187 Relative Error= 1.0770299989324195
Using 4 Terms Approximation= 0.09093061804404451 Relative Error= 0.4223942119139371
Using 5 Terms Approximation= 0.07473634273619677 Relative Error= 0.16907311986036705
Using 6 Terms Approximation= 0.06806885102238994 Relative Error= 0.06477599942169085
Using 7 Terms Approximation= 0.06539488510404382 Relative Error= 0.022948134100591072
```

Method 2 is more accurate as the relative error is smaller than the relative error in Method 1. However, this could be due to a small number of terms. If more terms were used for Method 1, the relative error could be smaller.