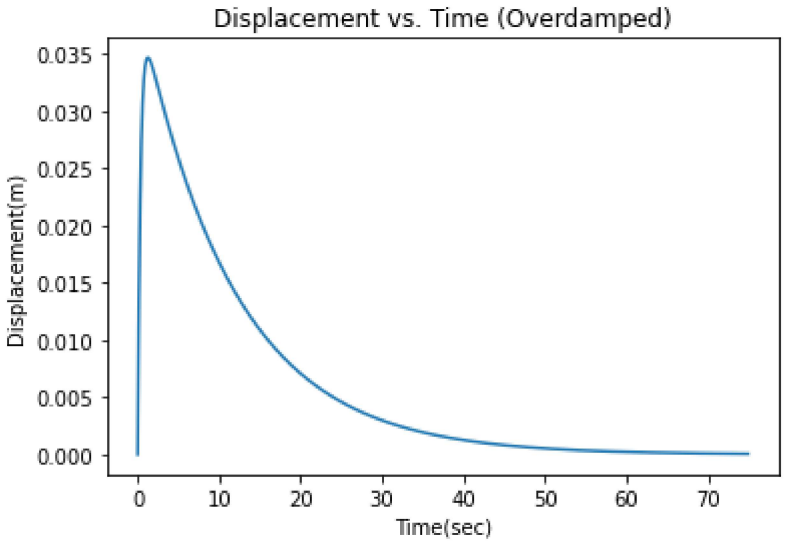
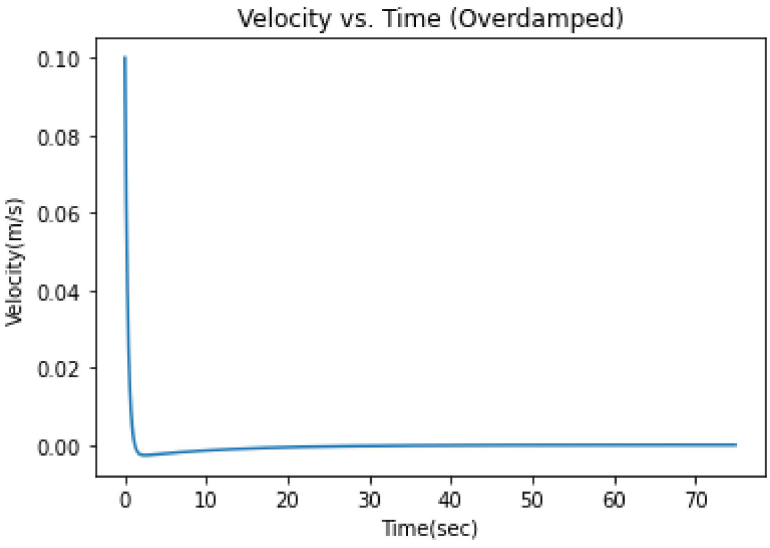
Experiment #1 F/m= 0; overdamped

In [16]:

```
%matplotlib inline
from math import*
import numpy as np
import sys
sys.path.append("/home/nfox4/engr3703/Notes/psm_plot/")
from psm_plot import *

c1 = 1./6
c2 = 2./6
c3 = 2./6
c4 = 1./6
a2 = 1./2
a3 = 1./2
a4 = 1.
b21 = 1./2
b31 = 0.
b32 = 1./2
b41 = 0.
b42 = 0.
b43 = 1.




f_m = lambda t: 0
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 3.0
omega = 0.5
ti = 0.0                          #initial time
tf = 75.0                        #final time
n = 1000                          #segments
dt = (tf-ti)/n                    #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                        #initial position
v_iv = 0.1                        #initial velocity
```

```
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Overdamped)","DvT
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Overdamped)","VvT1.png"
```



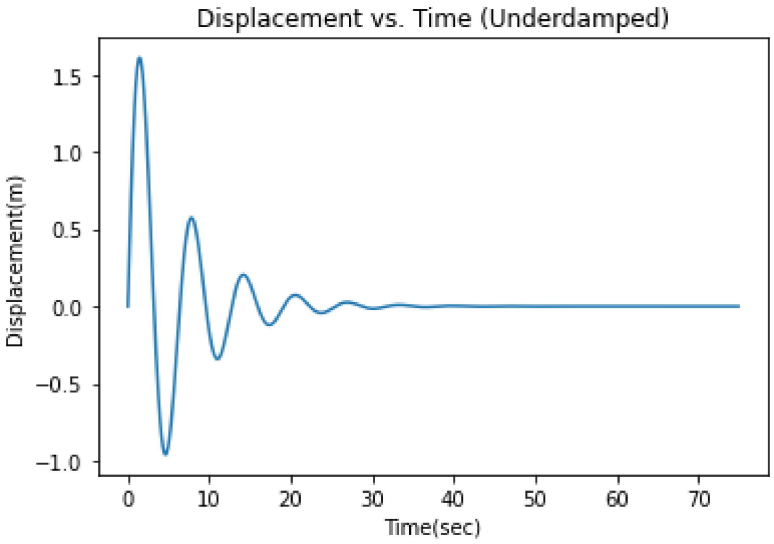


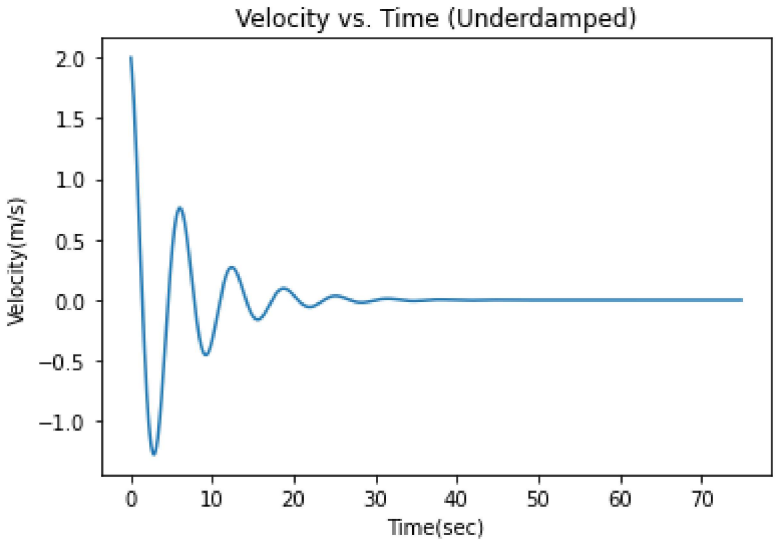
F/m=0 ; Underdamped

In [41]:

```
f_m = lambda t: 0
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
```

```
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 0.2
omega =1.0                          #initial time
tf = 75.0                           #final time
n = 1000                            #segments
dt = (tf-ti)/n                      #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                          #initial position
v_iv = 2.0                          #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Underdamped)","Dv
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Underdamped)","VvT2.png
```



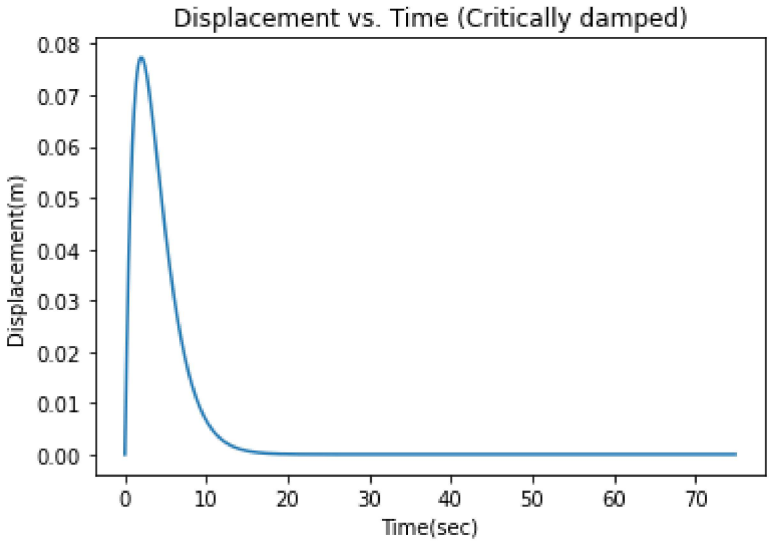


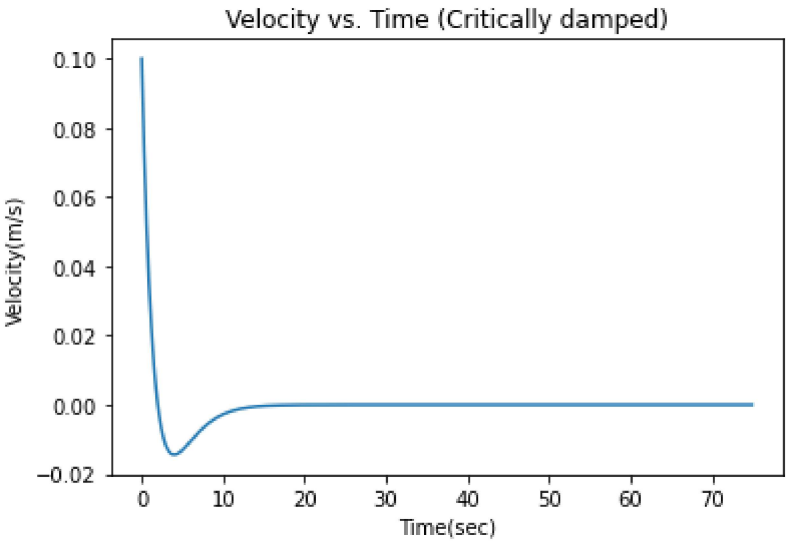
F/m=0 ; Critically damped

In [18]:

```
f_m = lambda t: 0
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 1.0
omega =0.5                      #initial time
tf = 75.0                       #final time
n = 1000                        #segments
dt = (tf-ti)/n                  #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                      #initial position
v_iv = 0.1                      #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
```

```
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)



LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Critically damped
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Critically damped)","Vv
```



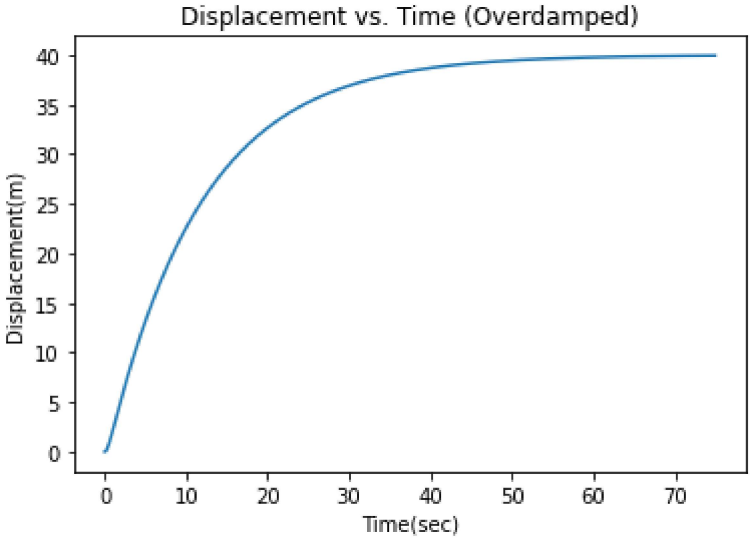


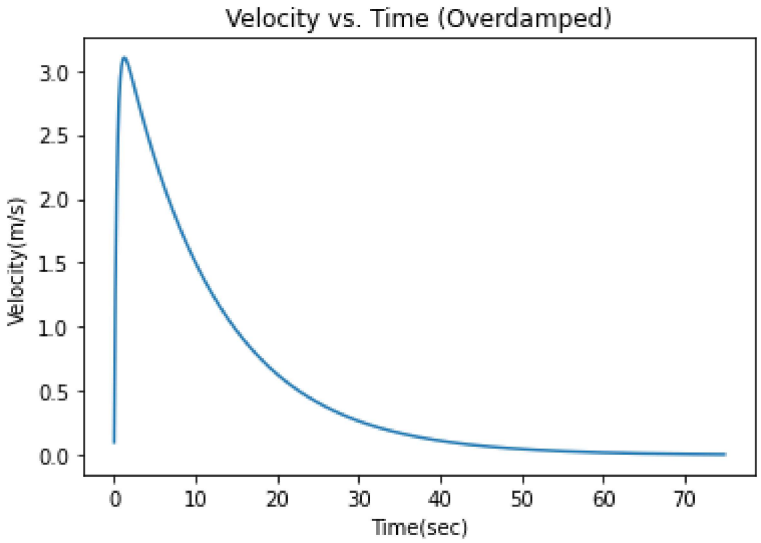
F/m= 10 ; Overdamped

In [19]:

```
f_m = lambda t: 10
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1
```

```
def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 3.0
omega = 0.5
ti = 0.0                          #initial time
tf = 75.0                        #final time
n = 1000                          #segments
dt = (tf-ti)/n                    #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                        #initial position
v_iv = 0.1                        #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)

LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Overdamped)","DvT
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Overdamped)","VvT4.png"
```



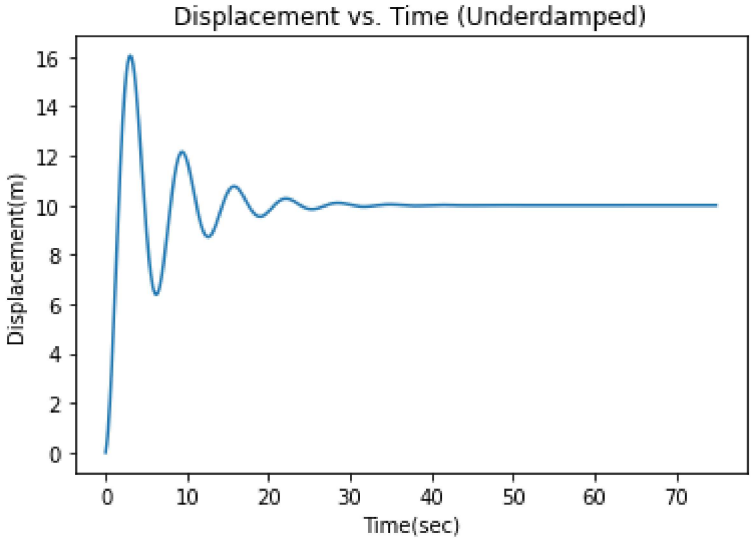


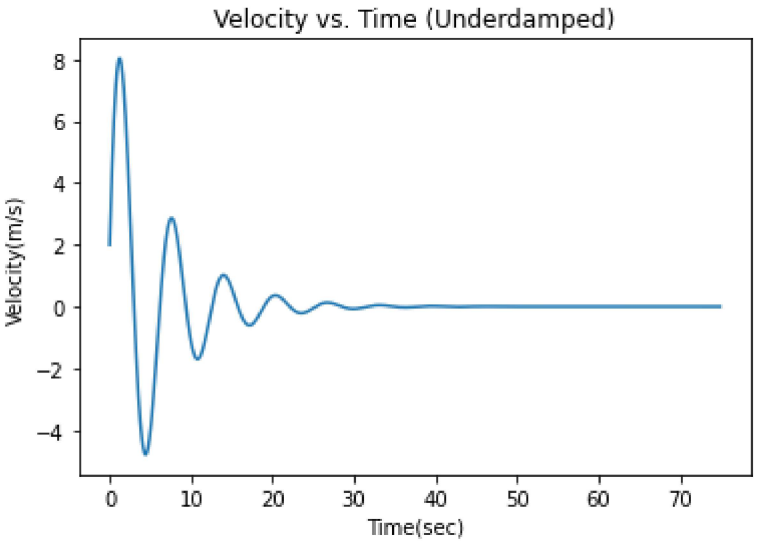
F/m= 10 ; Underdamped

In [20]:

```
f_m = lambda t: 10
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 0.2
omega =1.0                      #initial time
tf = 75.0                       #final time
n = 1000                        #segments
dt = (tf-ti)/n                  #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                      #initial position
v_iv = 2.0                      #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
```

```
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Underdamped)","Dv
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Underdamped)","VvT5.png
```



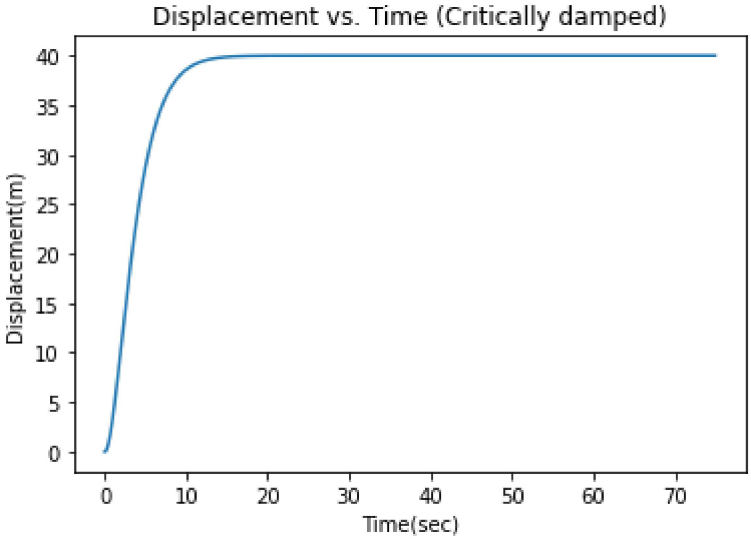


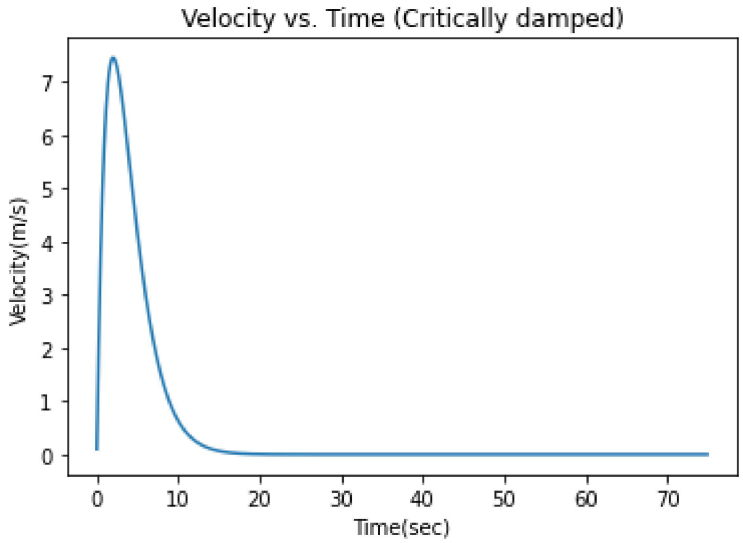
F/m= 10 ; Critically Damped

In [29]:

```
f_m = lambda t: 10
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
```

```
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 1.0
omega =0.5                          #initial time
tf = 75.0                           #final time
n = 1000                            #segments
dt = (tf-ti)/n                      #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                          #initial position
v_iv = 0.1                          #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)



LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Critically damped
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Critically damped)","Vv
```



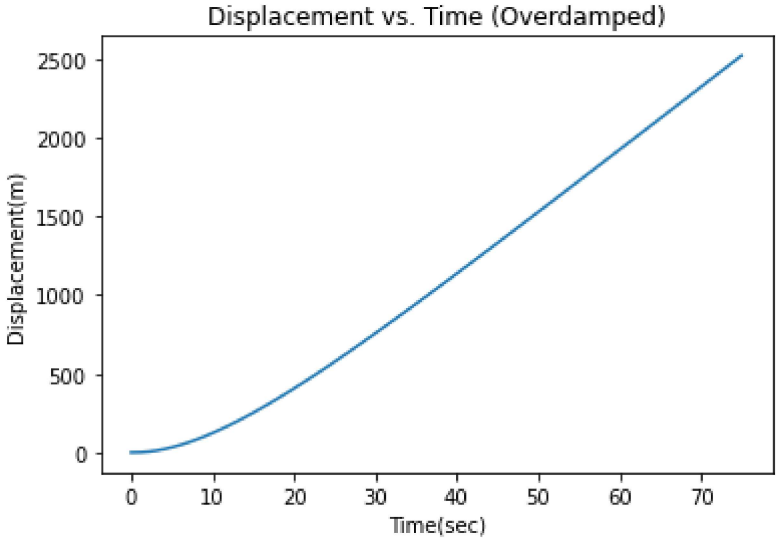


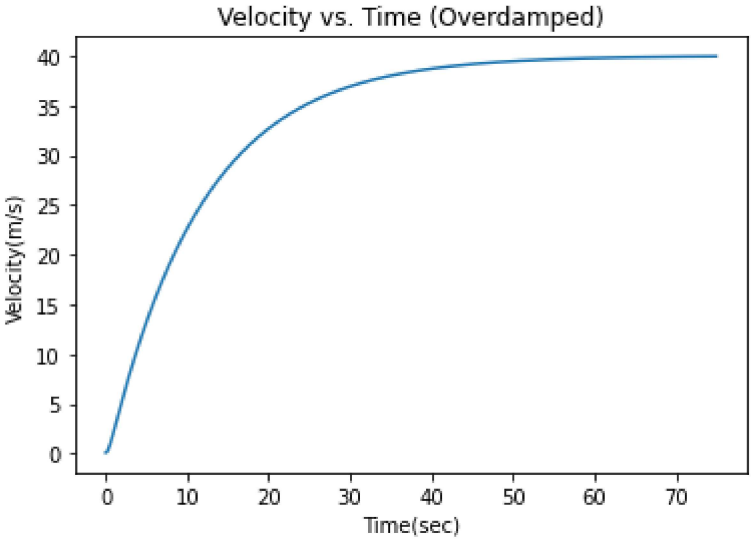
F/m= 10t ; Overdamped

In [23]:

```
f_m = lambda t: 10*t
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 3.0
omega = 0.5
ti = 0.0                          #initial time
tf = 75.0                        #final time
n = 1000                          #segments
dt = (tf-ti)/n                    #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                        #initial position
v_iv = 0.1                        #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv
```

```
for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)

LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Overdamped)","DvT
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Overdamped)","VvT7.png"
```



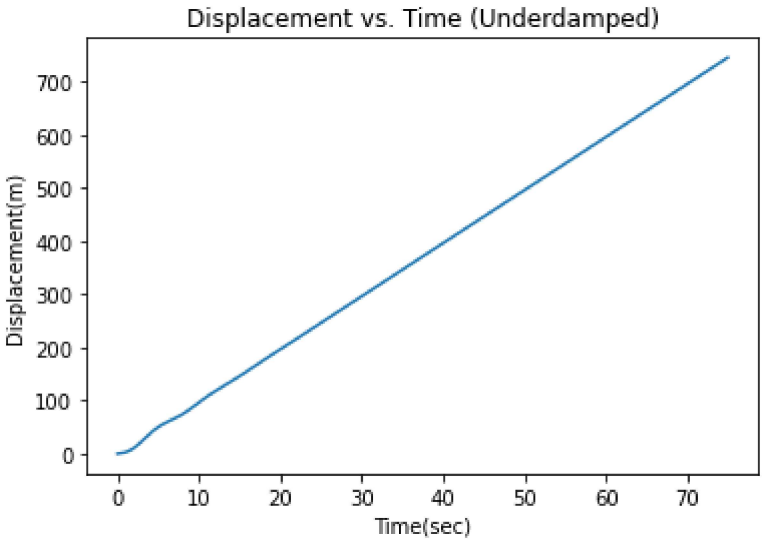


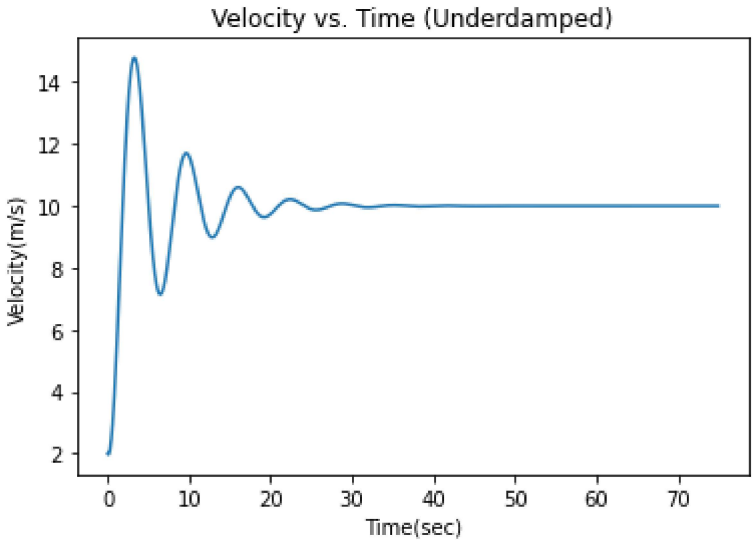
F/m= 10t ; Underdamped

In [25]:

```
f_m = lambda t: 10*t
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
```

```
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 0.2
omega =1.0                          #initial time
tf = 75.0                           #final time
n = 1000                            #segments
dt = (tf-ti)/n                      #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                          #initial position
v_iv = 2.0                          #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Underdamped)","Dv
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Underdamped)","VvT8.png
```



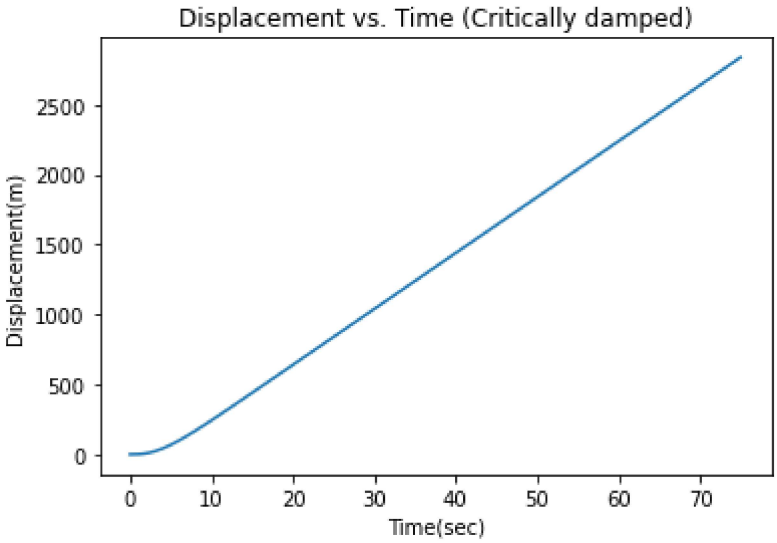


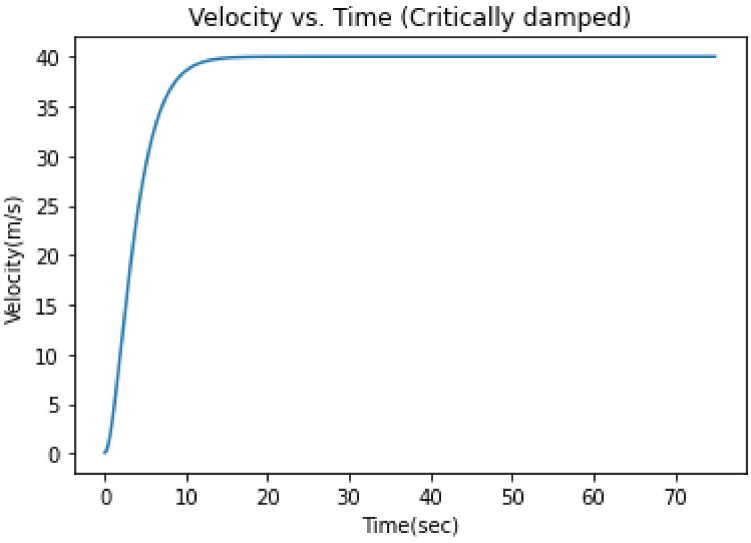
F/m= 10t ; Critically damped

In [28]:

```
f_m = lambda t: 10*t
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 1.0
omega =0.5                          #initial time
tf =20.0                            #final time(different because it goes up so fast)
n = 1000                            #segments
dt = (tf-ti)/n                      #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                          #initial position
v_iv = 0.1                          #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
```

```
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)



LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Critically damped
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Critically damped)","Vv
```



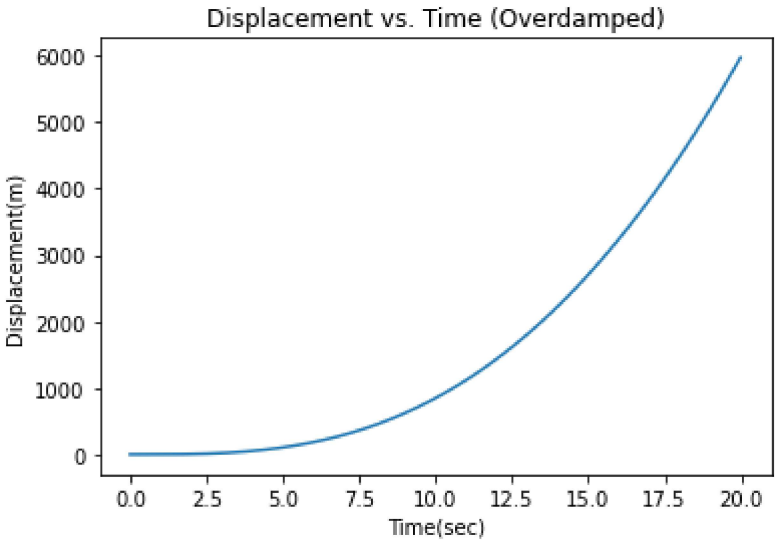


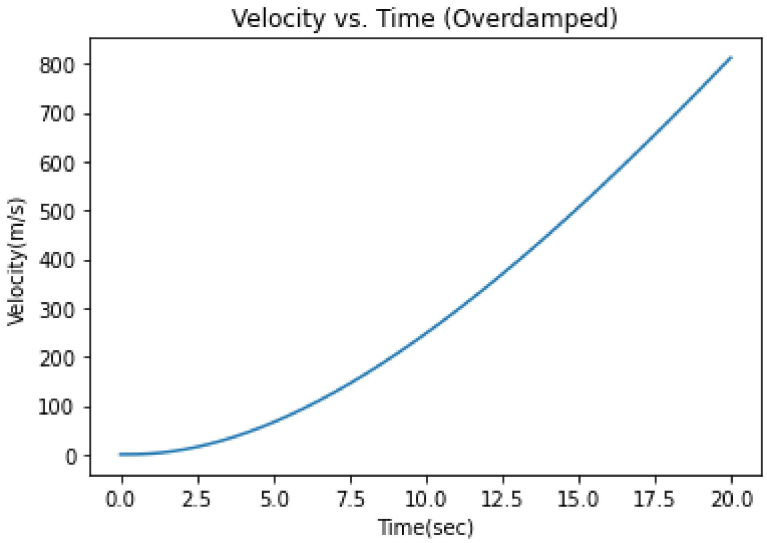
F/m= 10t**2 ; Overdamped

In [34]:

```
f_m = lambda t: 10*t**2
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1
```

```
def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 3.0
omega = 0.5
ti = 0.0                          #initial time
tf = 20.0                         #final time(different because it goes up so fast)
n = 1000                          #segments
dt = (tf-ti)/n                    #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                        #initial position
v_iv = 0.1                        #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Overdamped)","DvT
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Overdamped)","VvT10.png
```



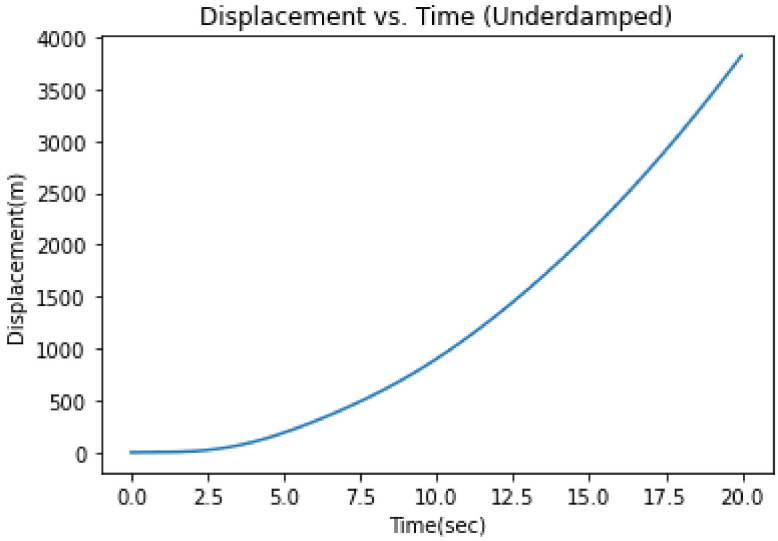


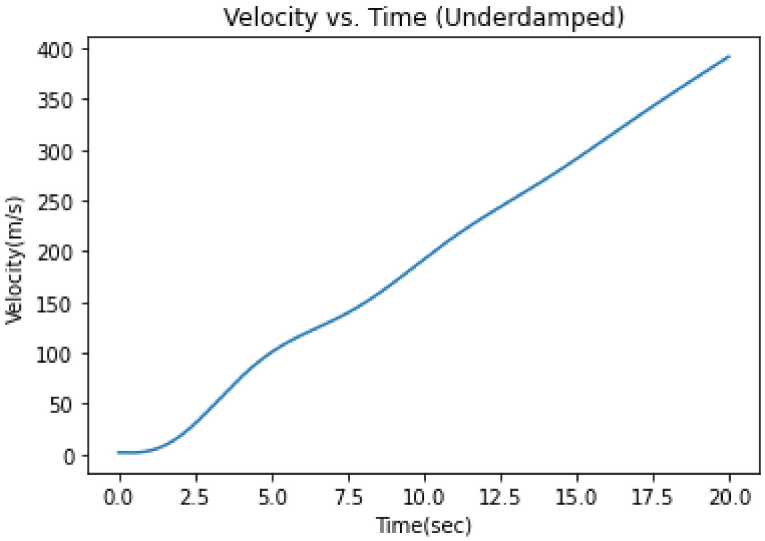
F/m= 10t**2 ; Underdamped

In [33]:

```
f_m = lambda t: 10*t**2
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 0.2
omega =1.0                      #initial time
tf = 20.0                       #final time(different because it goes up so fast)
n = 1000                        #segments
dt = (tf-ti)/n                  #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                      #initial position
v_iv = 2.0                      #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
```

```
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Underdamped)","Dv
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Underdamped)","VvT11.pn
```



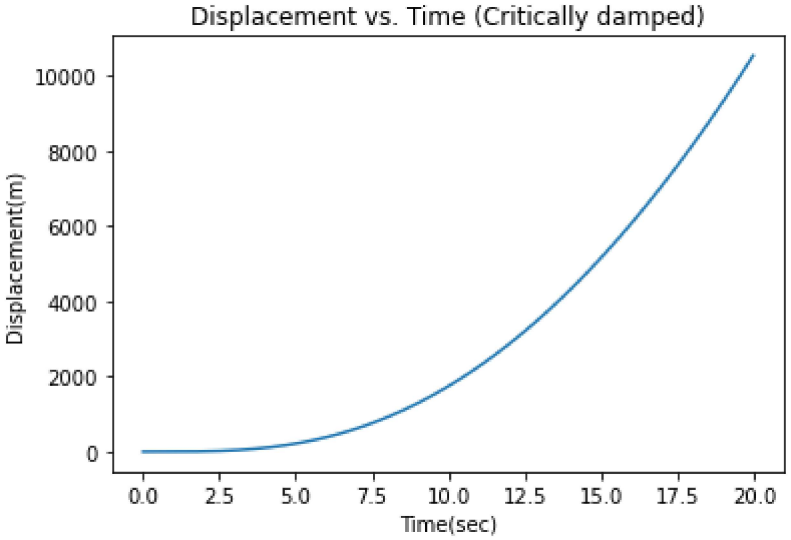


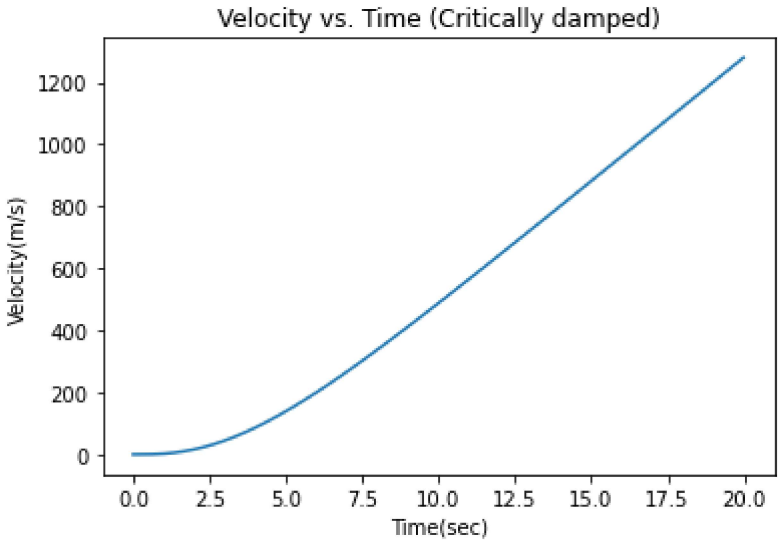
F/m= 10t**2 ; Critically damped

In [35]:

```
f_m = lambda t: 10*t**2
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
```

```
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 1.0
omega =0.5                          #initial time
tf =20.0                            #final time
n = 1000                            #segments
dt = (tf-ti)/n                      #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                          #initial position
v_iv = 0.1                          #initial velocity(different because it goes up so fast
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)



LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Critically damped
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Critically damped)","Vv
```



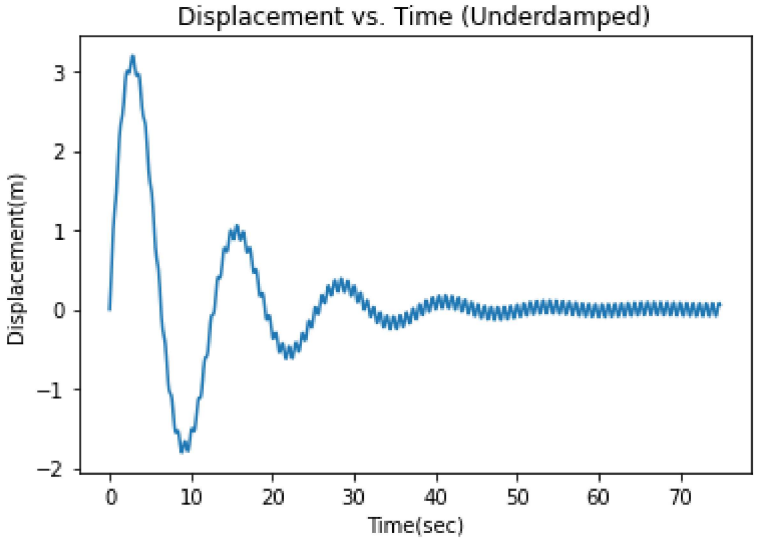


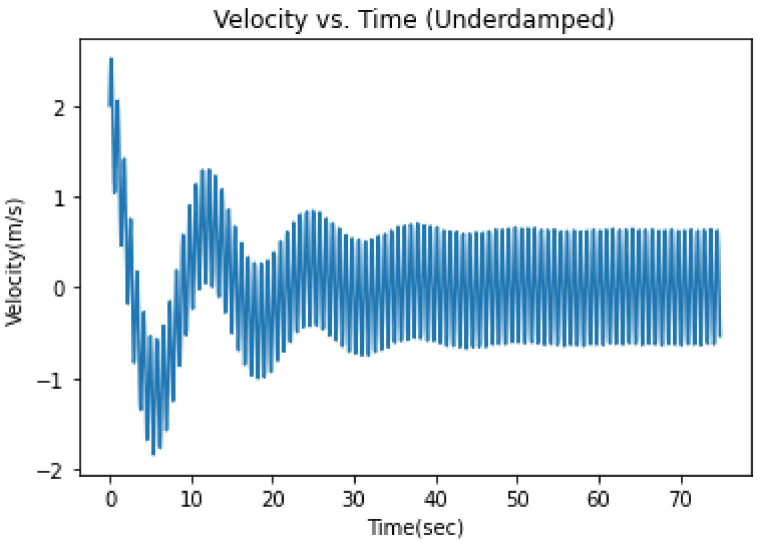
F/m= 5cos(omega5pi*t) omega=0.5

In [40]:

```
f_m = lambda t: 5*cos(omega*5*pi*t)
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 0.2
omega =0.5                          #initial time
tf = 75.0                           #final time
n = 1000                            #segments
dt = (tf-ti)/n                      #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                          #initial position
v_iv = 2.0                          #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
```

```
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Underdamped)","Dv
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Underdamped)","VvT13.pn
```



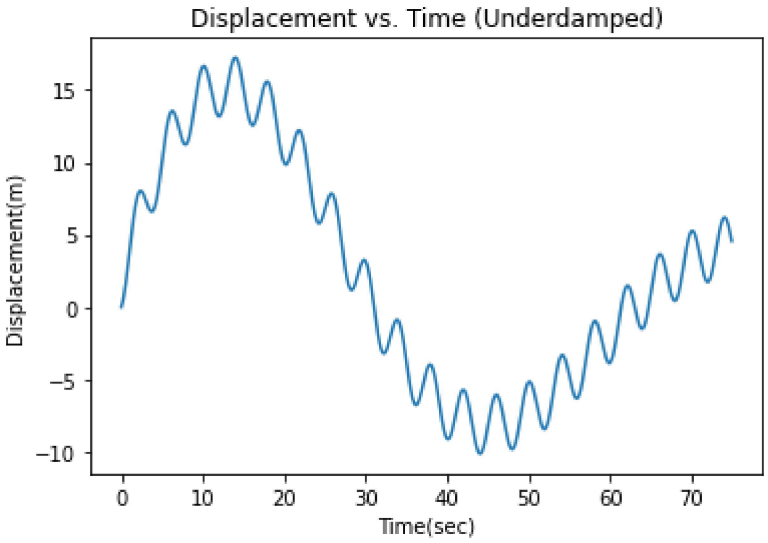


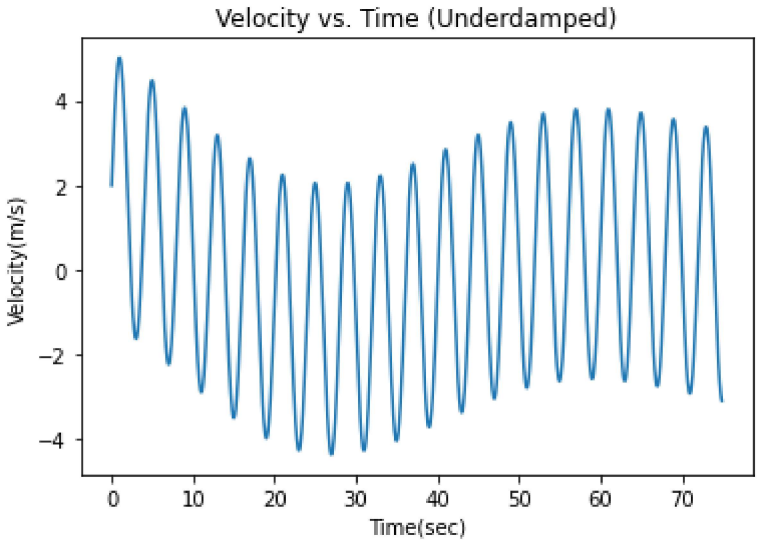
F/m= 5cos(omega5pi*t) omega=0.1

In [43]:

```
f_m = lambda t: 5*cos(omega*5*pi*t)
fxy_x = lambda t,x,v: v
fxy_v = lambda t,x,v: f_m(t)-2*zeta*omega*v-((omega**2)*x)

def rk4_x(ti,xi,vi,dt):
    K1 = fxy_x(ti,xi,vi)
    K2 = fxy_x(ti+a2*dt,xi+b21*K1*dt,vi)
    K3 = fxy_x(ti+a3*dt,xi+b31*K1*dt+b32*K2*dt,vi)
    K4 = fxy_x(ti+a4*dt,xi+b41*K1*dt+b42*K2*dt+b43*K3*dt,vi)
    xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return xip1

def rk4_v(ti,xi,vi,dt):
```

```
    K1 = fxy_v(ti,xi,vi)
    K2 = fxy_v(ti+a2*dt,xi,vi+b21*K1*dt)
    K3 = fxy_v(ti+a3*dt,xi,vi+b31*K1*dt+b32*K2*dt)
    K4 = fxy_v(ti+a4*dt,xi,vi+b41*K1*dt+b42*K2*dt+b43*K3*dt)
    vip1 = vi+(c1*K1+c2*K2+c3*K3+c4*K4)*dt
    return vip1


zeta = 0.2
omega =0.1#initial time
tf = 75.0                          #final time
n = 1000                           #segments
dt = (tf-ti)/n                     #step size
t = np.zeros(n)
x = np.zeros(n)
v = np.zeros(n)
x_iv = 0.0                         #initial position
v_iv = 2.0                         #initial velocity
t[0] = ti
x[0] = x_iv
v[0] = v_iv



for i in range(1,n):
    t[i] = ti+i*dt #increases time
    x[i] = rk4_x(t[i-1], x[i-1], v[i-1],dt)
    v[i] = rk4_v(t[i-1], x[i-1], v[i-1],dt)


LinePlot111(t,x,"Time(sec)","Displacement(m)","Displacement vs. Time (Underdamped)","Dv
LinePlot111(t,v,"Time(sec)","Velocity(m/s)","Velocity vs. Time (Underdamped)","VvT14.pn
```




Velocity vs. Time (Underdamped)
Velocity(m/s)
Time(sec)
4
2
0
−2
−4
0
10
20
30
40
50
60
70

In [ ]: