We look at capital insurance and denote:

$$b : \mathbb{N} \to \mathbb{R}, \xi \mapsto b(\xi)$$

The benefit due if person dies at age ξ. We assume that the policy is issues at age x and matures at age s. If the person suvives util s a benefit of $m(s)$ is due.

We use the normal notation regarding v and K and denote by Z the random present value of this insurance. Z can be calculated as follows:

$$Z = \chi_{K_x < s-x} \times v^{K_x+1} \times b(x + K_x) + \chi_{K_x \geq s-x} \times v^{s-x} \times m(s)$$

Looking at the expected value we get

$$\mathbb{E}[Z] = \sum_{k=0}^{s-x-1} {}_k p_x \, q_{x+k} \, v^{k+1} \times b(x + k) + {}_{s-x}p_x \times v^{s-x} \times m(s)$$

This can be calculated via commutation functions as follows

$$A_x^{gen} := \mathbb{E}[Z] = \frac{1}{D_x} \left(\sum_{k=0}^{s-x-1} C_{x+k} \times b(x + k) + D_s \times m(s) \right)$$

We can also calculate this recursively as follows:

$$\begin{aligned} A_s^{gen} &= m(s) \\ A_x^{gen} &= q_x \times v \times b(x) + p_x \times v \times A_{x+1}^{gen} \end{aligned}$$

We will use this in the following for our calculations and note that further details can be found in the script

In [46]:
```
import math
def Qx(gender,x,t,param =[]):
    # This is our default mortality
    if gender == 0:
        a =[2.34544649e+01,8.70547812e-02,7.50884047e-05,-1.6791793
5e-02]
    else:
        a =[2.66163571e+01,8.60317509e-02,2.56738012e-04,-1.9163267
5e-02]
    return(np.exp(a[0]+(a[1]+a[2]*x)*x+a[3]*t))

def QxNoReduction(gender,x,t,param =[]):
    # This is our default mortality
    t = 2015
    if gender == 0:
        a =[2.34544649e+01,8.70547812e-02,7.50884047e-05,-1.6791793
5e-02]
    else:
        a =[2.66163571e+01,8.60317509e-02,2.56738012e-04,-1.9163267
5e-02]
    return(np.exp(a[0]+(a[1]+a[2]*x)*x+a[3]*t))



def QxMedieval(gender,x,t,param =[]):
    a=-9.13275
    b= 8.09432e-2
    c= -1.1018e-5
    value=math.exp(a+(b+c*x)*x)
    alpha = 7.26502413
    beta = 0.01342065
    return(max(0,min(1,alpha*value+beta)))

def ConstantBenefit(x):
    return(1.)

def ZeroBenefit(x):
    return(0.)
```

In [40]:
```
import numpy as np
import matplotlib.pyplot as plt

class Capital:
    def __init__(self,Qx,i=0.02,omega=110,t0 = 2020):
        self.dIrate = i
        self.nOmega = omega
        self.psymB = ConstantBenefit
        self.psymM = ConstantBenefit
        self.psymQx = Qx
        self.dV = 1. /(1+self.dIrate)
        self.nT0 = t0
```

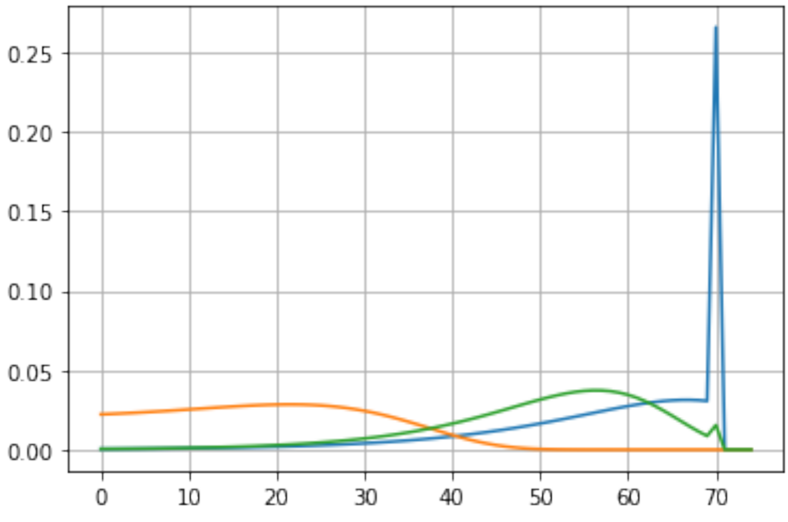
```
    def CommFu(self,gender,x,symbol):
        # Allowable symbnols l, D, N, S, C, M, R
        if symbol == "l":
            l = 100000.
            px = 1.
            for i in range(x):
                t = self.nT0 + i
                qx = self.psymQx(gender,i,t)
                px *= (1-qx)
            return(l * px)
        if symbol == "D":
            return(self.CommFu(gender,x,"l")*self.dV**x)
        if symbol == "C":
            return((self.CommFu(gender,x,"l")-self.CommFu(gender,x+
1,"l"))*self.dV**(x+1))
        if symbol == "N":
            sum = 0
            for i in range(x,self.nOmega):
                sum += self.CommFu(gender,i,"D")
            return(sum)
        if symbol == "S":
            sum = 0
            for i in range(x,self.nOmega):
                sum += self.CommFu(gender,i,"N")
            return(sum)
        if symbol == "M":
            sum = 0
            for i in range(x,self.nOmega):
                sum += self.CommFu(gender,i,"C")
            return(sum)
        if symbol == "R":
            sum = 0
            for i in range(x,self.nOmega):
                sum += self.CommFu(gender,i,"M")
            return(sum)
        return("Error")

    def CalcPV(self,gender,x,s):
        PV = self.psymM(s)
        n  = s-x
        for i in range(s-1,x-1,-1):
            t = self.nT0 + i - x
            qx = self.psymQx(gender,i,t)
            px = 1. - qx
            PV = qx * self.dV * self.psymB(i) + px * self.dV * PV
        return(PV)

    def CalcCF(self,gender,x,s,periods = 50):
        CF = np.zeros(periods)
        px = 1
        for i in range(x,s):
            t = self.nT0 + i - x
            qx = self.psymQx(gender,i,t)
            n = i - x
```

```
            if n >= periods:
                break
            CF[n] = px * qx * self.psymB(i)
            px *= (1-qx)
            #print(i, px, qx)

        n = s - x
        if n < periods:
            CF[n] = px * self.psymM(s)

        return(CF)
```

In [47]:
```
Ax = Capital(Qx)
AxMed = Capital(QxMedieval)
AxNR = Capital(QxNoReduction)
print(Ax.CalcPV(0,30,100), Ax.CalcPV(1,30,100),AxMed.CalcPV(0,30,10
0))

print("Example Ax")
AxNRPV = AxNR.CalcPV(0,30,110)
AxNRPVCF = AxNR.CommFu(0,30,"M")/AxNR.CommFu(0,30,"D")
print(AxNRPV,AxNRPVCF,AxNRPV-AxNRPVCF )

print("Example Ax:n")
x = 30
s = 65
n =  s - x
print("n = ", n)
AxNRPV = AxNR.CalcPV(0,30,65)
AxNRPVCF = (AxNR.CommFu(0,30,"M")-AxNR.CommFu(0,65,"M")+AxNR.CommFu
(0,65,"D"))/AxNR.CommFu(0,30,"D")
print(AxNRPV,AxNRPVCF,AxNRPV-AxNRPVCF )

print("Example Ax:n^1")
x = 30
s = 65
n =  s - x
AxNR.psymM = ZeroBenefit
print("n = ", n)
AxNRPV = AxNR.CalcPV(0,30,65)
AxNRPVCF = (AxNR.CommFu(0,30,"M")-AxNR.CommFu(0,65,"M"))/AxNR.CommF
u(0,30,"D")
print(AxNRPV,AxNRPVCF,AxNRPV-AxNRPVCF )
```

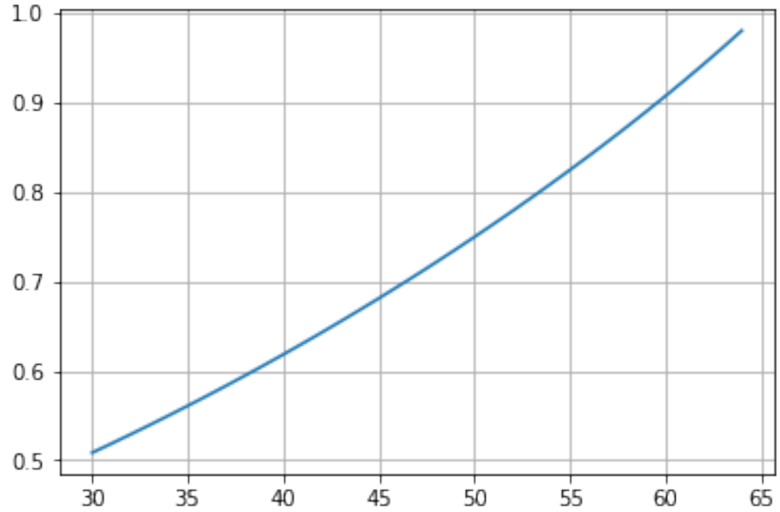
```
0.3249739178301026 0.2889270919233321 0.6828173613075574
Example Ax
0.37464976341934786 0.3746497634357318 -1.6383949752452054e-11
Example Ax:n
n =  35
0.511759031333641 0.5117590313336409 1.1102230246251565e-16
Example Ax:n^1
n =  35
0.0673058681223249 0.06730586812232482 8.326672684688674e-17
```

In [38]:
```
CF1 = Ax.CalcCF(0,30,100,periods=75)
CF2 = AxMed.CalcCF(0,30,100,periods=75)
CF3 = AxNR.CalcCF(0,30,100,periods=75)
plt.figure(1)
plt.plot(range(len(CF1)),CF1,range(len(CF2)),CF2,range(len(CF3)),CF
3)
plt.grid(True)
```



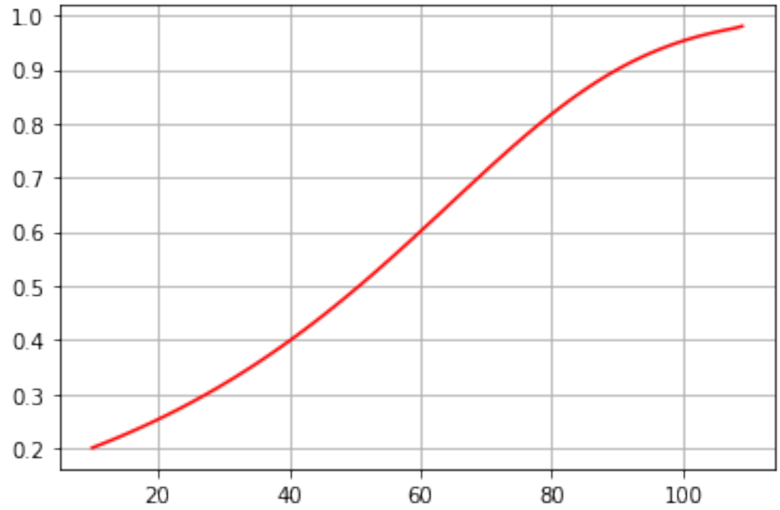
In [48]:
```
print("How does Expected Value move for changing ages, Example Ax:n
")

x=range(30,65)
y = []
for i in x:
    y.append(Ax.CalcPV(0,i,65))

plt.figure(2)
plt.plot(x,y)
plt.grid(True)
```

```
How does Expected Value move for changing ages, Example Ax:n
```



In [50]:
```
print("How does Expected Value move for changing ages, Example Ax")

x=range(10,110)
y = []
for i in x:
    y.append(Ax.CalcPV(0,i,110))

plt.figure(3)
plt.plot(x,y,"r")
plt.grid(True)
```

```
How does Expected Value move for changing ages, Example Ax
```



In [ ]: