라이나 생명 간편고지 항암방사선 치료특약

In [1]:

```
import numpy as np
import pandas as pd
```

PV

In [2]:

```
class Calculation:
    def __init__(self, excel_file : str = './INFO.xlsx'):
        self.excel_file = excel_file
        self.sht_rate = pd.read_excel(self.excel_file, sheet_name="위험률", header = 1).f
illna(0)
        self.sht_rate = self.sht_rate[['Key', 'x', 'Male', 'Female']]

    def getQx(self, rKey : str, sex : int = None) -> np.array:
        if sex==None:sex = self.sex
        qx = np.zeros(120)
        df = self.sht_rate.loc[self.sht_rate['Key'] == rKey].copy(deep=True)
        if sex == 1:
            df = df[['x', 'Male']].values
        else:
            df = df[['x', 'Female']].values
        for row in df:
            age, rate = row
            qx[int(age)] = rate
        return qx[self.x:]

    def setArgs(self, x : int, sex : int, \
        n : int, m : int, re: int, hyung : int):
        #========== Set Arguments ==========#
        self.x = x
        self.sex = sex

        self.n = n  # 보험기간
        self.m = m  # 납입기간

        self.re = re  # 1 : 최초계약, 2 : 갱신계약
        self.hyung = hyung   # 형

        assert re in [1, 2]
        assert hyung in [1, 2]

        self.w = 108 if sex==1 else 110
        self.l0 = 100000
        self.i = 0.0225
        self.v = 1/(1+self.i)
        self.AMT = 1000000   # 가입금액

        self.beta5 = 0
        if self.re == 1:
            self.alpha1 = 0.35/1000
            self.alpha2 = 0.05*min(self.n, 20)
            if self.hyung == 1:self.S = 0.18
            else:self.S = 0.17
        else:
            self.alpha1 = 0.245/1000*min(self.n, 10)/10
            self.alpha2 = 0.035*min(self.n, 20)
            self.S = 0.01
        self.beta1 = 0.002/1000
        self.beta2 = 0.385 - self.beta5
        self.betaPrime = self.beta1/2
```

```
def main(self):
    # a
    if self.re==1:a = [3/4] + [1.]*(self.n)
    else:a = [1.]*(self.n+1)

    #=========== RiskRate ===========#
    # 나중에 위험율 긁어오는 코드 추가!!!!
    qx_t = self.getQx("Qt")
    qx_c = self.getQx("Qc")
    qx_ca = self.getQx("Qca")


    # lx
    lx = [self.l0]
    lx_c = [self.l0]
    lx_ca = [self.l0]
    for t in range(self.n+1):
        lx.append(lx[t]*(1-qx_t[t]*a[t]))
        lx_c.append(lx_c[t]*(1-qx_c[t]*a[t]))
        lx_ca.append(lx_ca[t]*(1-(qx_ca[t]-(1-a[t])*qx_c[t])))

    # Dx
    Dx = [lx[t]*self.v**t for t in range(self.n+1)]
    Dx_c = []
    for t in range(self.n+1):
        D = lx_c[t]*self.v**t
        Dx_c.append(D)


    # Nx
    Nx = [sum(Dx[t:]) for t in range(self.n+1)]
    Nx_c = [sum(Dx_c[t:]) for t in range(self.n+1)]
    Nshop = [max(Nx_c[t] - Nx_c[self.m], 0) for t in range(self.n+1)]
    Nstar = 12*((Nx_c[0] - Nx_c[self.m]) - 11/24*(Dx_c[0] - Dx_c[self.m]))

    # M#
    dx_t = [lx_ca[t]*a[t]*qx_t[t] for t in range(self.n+1)]
    Cx_t = []
    for t in range(self.n+1):
        if t==0 and self.re == 1:
            C = dx_t[t]*self.v**(t+5/8)
        else:
            C = dx_t[t]*self.v**(t+1/2)
        Cx_t.append(C)

    Mx_t = [sum(Cx_t[t:]) for t in range(self.n+1)]
    if self.re==1 and self.hyung==2:
        Mshop = []
        for t in range(self.n+1):
            if self.re==1 and t<2:
                Mshop.append(0.5*(Mx_t[t]-Mx_t[2])+(Mx_t[2] - Mx_t[self.n]))
            else:
                Mshop.append(Mx_t[t] - Mx_t[self.n])
    else:
        Mshop = [Mx_t[t] - Mx_t[self.n] for t in range(self.n+1)]


    #=========== Prenium ===========#

    # 월납 순보험료
    NP = Mshop[0] / Nstar
    # 기준 연납 순보험료
    NP_std = Mshop[0]/(Nx_c[0] - Nx_c[min(self.n, 20)])
    # 연납 베타순보험료
    NP_beta = Mshop[0]/(Nx_c[0] - Nx_c[self.m]) + \
        self.betaPrime*(Nx[0]-Nx[self.n]-Nshop[0])/Nshop[0]
    # 월납 영업보험료
    G = (NP+ (self.alpha1+self.alpha2*NP_std)*Dx[0]/Nstar+self.beta1/12 \
        +self.betaPrime*(Nx[0]-Nx[self.n]-Nstar/12)/Nstar)\
        /(1-self.beta2-self.beta5)
```

```
        tVx = []
        for t in range(self.n+1):
            if t==0:V=0
            elif t<self.m:V = (Mshop[t]+self.betaPrime*(Nx[t]-Nx[self.n]-Nshop[t])-NP_be
ta*Nshop[t])/Dx[t]
            else:V = (Mshop[t]+self.betaPrime*(Nx[t]-Nx[self.n]))/Dx[t]
            V = round(V,6)
            tVx.append(V)


        alpha_std = NP_std*0.05*min(self.n, 20)+10/1000*self.S
        alpha_apply = self.alpha1+self.alpha2*NP_std
        alphaPrime = min(alpha_std, alpha_apply)

        tWx = []
        for t in range(self.n+1):
            V = tVx[t]
            W = max(V - max(0, (1-t/min(7, self.m)))*alphaPrime, 0)
            tWx.append(W)


        #=========== Output ===========#

        return {'G' : round(self.AMT*G), \
            'NP_beta' : round(self.AMT*NP_beta), \
                'tVx' : [round(V*self.AMT) for V in tVx], \
                    'tWx' : [round(W*self.AMT) for W in tWx]}
```

In [3]:

```
Cal = Calculation()
```

영업보험료 확인

In [4]:

```
sht_G = pd.read_excel("./INFO.xlsx", sheet_name="영업보험료 확인", header=1).fillna(0.)
sht_G.head()
```

Out[4]:

	Coverage	Sub1	Sub2	Sub3	Key	sex	x	n	m	G
0	1	1	1	0.0	1_1_1	1	40	10	10	45
1	1	1	1	0.0	1_1_1	1	50	10	10	105
2	1	1	1	0.0	1_1_1	1	60	10	10	217
3	1	1	1	0.0	1_1_1	2	40	10	10	123
4	1	1	1	0.0	1_1_1	2	50	10	10	153

In [5]:

```
cnt_error = 0
for i, row in enumerate(sht_G.values):
    _, hyung, re, _, _, sex, x, n, m, G = row
    Cal.setArgs(x, sex, n, m, re, hyung)
    result = Cal.main()
    if result['G']!=G:
        print(f"{i+1}th case - Cal : {result['G']} / G : {G}")
        cnt_error+=1
if cnt_error == 0:
    print(f"Pass")
```

```
23th case - Cal : 183 / G : 182
```

준비금

In [6]:

```
sht_V = pd.read_excel("./INFO.xlsx", sheet_name="준비금 확인", header=1).fillna(0.)
sht_V.head()
```

Out[6]:

	Coverage	Sub1	Sub2	Sub3	Key	sex	x	n	m	NP_beta	V(1)	V(3)	V(5)	V(7)	V(10)	W(1)	W(3)	W(5)	W(7)	W(10)
0	1	1	1	0.0	1_1_1	1	40	10	10	270	133	258	315	291	0	0	0	176	291	0
1	1	1	1	0.0	1_1_1	2	40	10	10	812	324	519	551	419	0	0	87	335	419	0
2	1	1	2	0.0	1_1_2	1	50	10	10	696	237	568	695	601	0	0	312	567	601	0
3	1	1	2	0.0	1_1_2	2	50	10	10	1046	31	69	84	84	0	0	0	0	84	0
4	1	2	1	0.0	1_2_1	1	40	10	10	251	185	380	404	346	0	0	108	268	346	0

In [7]:

```
cnt_error = 0
for i, row in enumerate(sht_V.values):
    _, hyung, re, _, _, sex, x, n, m, NP_beta, V1, V3, V5, V7, V10, W1, W3, W5, W7, W10
= row
    tVx = [V1, V3, V5, V7, V10]
    tWx = [W1, W3, W5, W7, W10]
    Cal.setArgs(x, sex, n, m, re, hyung)
    result = Cal.main()
    if result['NP_beta']!=NP_beta:
        print(f"{i+1}th case - Cal : {result['NP_beta']} / 연납베타순보험료 : {NP_beta}")
        cnt_error+=1
    if result['tVx'][1] != V1:
        print(f"{i+1}th case - Cal : {result['tVx'][1]} / V(1) : {V1}")
        cnt_error += 1
    if result['tVx'][3] != V3:
        print(f"{i+1}th case - Cal : {result['tVx'][3]} / V(3) : {V3}")
        cnt_error += 1
    if result['tVx'][5] != V5:
        print(f"{i+1}th case - Cal : {result['tVx'][5]} / V(5) : {V5}")
        cnt_error += 1
    if result['tVx'][7] != V7:
        print(f"{i+1}th case - Cal : {result['tVx'][7]} / V(7) : {V7}")
        cnt_error += 1
    if result['tVx'][10] != V10:
        print(f"{i+1}th case - Cal : {result['tVx'][10]} / V(10) : {V10}")
        cnt_error += 1

    if result['tWx'][1] != W1:
        print(f"{i+1}th case - Cal : {result['tWx'][1]} / W(1) : {W1}")
        cnt_error += 1
    if result['tWx'][3] != W3:
        print(f"{i+1}th case - Cal : {result['tWx'][3]} / W(3) : {W3}")
        cnt_error += 1
    if result['tWx'][5] != W5:
        print(f"{i+1}th case - Cal : {result['tWx'][5]} / W(5) : {W5}")
        cnt_error += 1
    if result['tWx'][7] != W7:
        print(f"{i+1}th case - Cal : {result['tWx'][7]} / W(7) : {W7}")
        cnt_error += 1
    if result['tWx'][10] != W10:
        print(f"{i+1}th case - Cal : {result['tWx'][10]} / W(10) : {W10}")
        cnt_error += 1
if cnt_error == 0:
    print(f"Pass")
```

```
Pass
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```