Calculation of expected maturity value for *Futures Market* probabilistic game.

Stock values are integers $0 \leq v \leq 10$;
Probabilities p of v's rising or falling are multiples of 0.1 ;
The value v is capped at c, which itself has probability *cFalls* of falling in each period.

State declarations, constructors and destructors.

In[1]:=
```
maxV = 10
numV = maxV + 1
pSteps = 10
numP = pSteps + 1
pQuantum = 1.0 / pSteps
numC = numV
numStates = numV * numP * numC
```

In[8]:=
```
getV[state_] := Quotient[state - 1, numP * numC]
getP[state_] := pQuantum * Quotient[Mod[state - 1, numP * numC], numC]
getC[state_] := Mod[state - 1, numC]

makeState[v_, p_, c_] := 1 + (v * numP + Round[p / pQuantum]) * numC + c
```

Assorted transition parameters and functions.

In[12]:=
```
pUpProbs[v_ /; v < maxV / 2] := 2 / 3
pUpProbs[v_ /; v == maxV / 2] := 1 / 2
pUpProbs[v_ /; v > maxV / 2] := 1 / 3

cFalls = 0.5

upV[v_, c_] := Min[v + 1, c]
downV[v_] := Max[v - 1, 0]
upP[p_] := Min[p + pQuantum, 1]
downP[p_] := Max[p - pQuantum, 0]
downC[c_] := Max[c - 1, 0]

minExp[exp1_, exp2_] := Table[Min[exp1[[s]], exp2[[s]]], {s, 1, numStates}]
maxExp[exp1_, exp2_] := Table[Max[exp1[[s]], exp2[[s]]], {s, 1, numStates}]
```

Next-time modal operator, function of post-expectation and state.

In[23]:=
```
monthPoint[  exp_,      v_, p_, c_   ] := (
        p       * pUpProbs[v]           * (1-cFalls)
                * exp[[makeState[      upV[v,c], upP[p], c                    ]]]

    +   p       * pUpProbs[v]           * cFalls
                * exp[[makeState[      upV[v,c], upP[p], downC[c]     ]]]

    +   p       * (1-pUpProbs[v])       * (1-cFalls)
                * exp[[makeState[      upV[v,c], downP[p], c                  ]]]

    +   p       * (1-pUpProbs[v])       * cFalls
                * exp[[makeState[      upV[v,c], downP[p], downC[c] ]]]

    +   (1-p)   * pUpProbs[v]           * (1-cFalls)
                * exp[[makeState[      downV[v], upP[p], c                    ]]]

    +   (1-p)   * pUpProbs[v]           * cFalls
                * exp[[makeState[      downV[v], upP[p], downC[c]             ]]]

    +   (1-p)   * (1-pUpProbs[v])       * (1-cFalls)
                * exp[[makeState[      downV[v], downP[p], c                  ]]]

    +   (1-p)   * (1-pUpProbs[v])       * cFalls
                * exp[[makeState[      downV[v], downP[p], downC[c]           ]]]
    )
```

In[24]:=
```
monthExp[exp_] := Table[monthPoint[exp, getV[s], getP[s], getC[s]], {s, 1, numStates}]
```

Constant expectations, and the example iterator functions *f0,f1* etc:

function *f0* applies the demonic choice late;
function *f1* applies the demonic choice early;
function *f2* applies a fixed investor strategy;
function *f3* calculates probability;
function *f4* removes the *barring* feature;
function *f5* calculates probability using a strategy.

In[79]:=
```
vExp = Table[getV[s], {s, 1, numStates}]
constExp[x_] := Table[x, {s, 1, numStates}]
zeroExp = constExp[0]
strategy1[s_] := getC[s] ≤ (getV[s] + 1)
strategy2[s_] := (getV[s] ≥ 5) && (getP[s] ≥ 0.5)
condExp[b_, exp1_, exp2_] := Table[If[ b[s], exp1[[s]], exp2[[s]]], {s, 1, numStates}]
mvExp = monthExp[vExp]
vAtLeast[v_] := Table[If[getV[s] ≥ v, 1.0, 0.0], {s, 1, numStates}]
mv6Exp = monthExp[vAtLeast[6]]
f0[exp_] := maxExp [mvExp, monthExp[ minExp[exp, monthExp[exp]] ] ]
f1[exp_] := maxExp [mvExp, minExp[monthExp[exp], monthExp[monthExp[exp]] ] ]
f2[exp_] := condExp[strategy1, mvExp, monthExp[ minExp[exp, monthExp[exp]] ] ]
f3[exp_] := maxExp [mv6Exp, monthExp[ minExp[exp, monthExp[exp]] ] ]
f4[exp_] := maxExp [mvExp, monthExp[exp]]
f5[exp_] := condExp[strategy2, mv6Exp, monthExp[ minExp[exp, monthExp[exp]] ] ]
```

Calculation of various fixed points.

In[96]:=
```
fp5 = FixedPoint[f5, zeroExp]
```

Fixed-point applied to all initial states $0 \leqslant v \leqslant maxV$, with c at the maximum and initial share increase probability shown.

In[38]:=
```
initialIncreaseProb = 0.5
```

In[97]:=
```
Do[Print[
    "Initial share value",
    PaddedForm[v, 2],
    " gives highest expected maturity value",
    PaddedForm[fp5[[makeState[v, initialIncreaseProb, maxV]]], {4, 3}],
    "."
  ], {v, 0, maxV}]
```

```
Do[Print[
    "Initial share value",
    PaddedForm[v, 2],
    " gives expected maturity value",
    PaddedForm[mvExp[[makeState[v, initialIncreaseProb, maxV]]], {3, 2}],
    "."
  ], {v, 0, maxV}]
```

```
reserveNow = Table[mvExp[[s]] >= fp0[[s]], {s, 1, numStates}]
fixedCap = 8
Print["Cap is ", fixedCap, "."]
Do[
  Print[
    "For (v,p) at (",
    PaddedForm[v, 1], ",", PaddedForm[p, 2],
    ") reserve now if ",
    reserveNow[[makeState[v, p, fixedCap]]],
    "."
  ],
  {v, 0, maxV}, {p, 0, 1, pQuantum}
]
```