Recall from lecture[1] that Sam is either fit or unfit

\[ S = \{ \text{fit, unfit} \} \]

and has to decide whether to exercise or relax

\[ A = \{ \text{exercise, relax} \} \]

on the basis of the following (probability, reward)-matrices \((p(s, a, s'), r(s, a, s'))\) for row \(s\), column \(s'\) in table with corner \(a\)

<table>
<thead>
<tr>
<th></th>
<th>fit</th>
<th>unfit</th>
<th></th>
<th>fit</th>
<th>unfit</th>
</tr>
</thead>
<tbody>
<tr>
<td>exercise</td>
<td>.99, 8</td>
<td>.01, 8</td>
<td>relax</td>
<td>.7, 10</td>
<td>.3, 10</td>
</tr>
<tr>
<td>unfit</td>
<td>.2, 0</td>
<td>.8, 0</td>
<td>unfit</td>
<td>0, 5</td>
<td>1, 5</td>
</tr>
</tbody>
</table>

The \(\gamma\)-discounted value of \((s, a)\) is

\[
\lim_{n \to \infty} q_n(s, a)
\]

where

\[
q_0(s, a) := p(s, a, \text{fit})r(s, a, \text{fit}) + p(s, a, \text{unfit})r(s, a, \text{unfit})
\]

\[
V_n(s) := \max(q_n(s, \text{exercise}), q_n(s, \text{relax}))
\]

\[
q_{n+1}(s, a) := q_0(s, a) + \gamma (p(s, a, \text{fit})V_n(\text{fit}) + p(s, a, \text{unfit})V_n(\text{unfit})).
\]

In particular, \(\gamma = 0.9\) leads to the following \(q_n(s, a)\) for \(n = 0, 1, 2\)

<table>
<thead>
<tr>
<th></th>
<th>exercise</th>
<th>relax</th>
<th>(\pi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>fit</td>
<td>8, 16.955, 23.812</td>
<td>10, 17.65, 23.685</td>
<td>relax, relax, exercise</td>
</tr>
<tr>
<td>unfit</td>
<td>0, 5.4, 10.017</td>
<td>5, 9.5, 13.55</td>
<td>relax, relax, relax</td>
</tr>
</tbody>
</table>

For variety, let us add a state to \(S\), dead, for the new state set

\[ S' = \{ \text{fit, unfit, dead} \} \]

and revise the functions \(p\) and \(r\) to \(p'\) and \(r'\) as follows. Let us introduce a chance \(\frac{1}{10}\) of death from exercise

\[
p'(s, \text{exercise}, \text{dead}) = \frac{1}{10} \quad \text{for } s \in S
\]

\[
p'(s, \text{exercise}, s') = \frac{9p(s, \text{exercise}, s')}{10} \quad \text{for } s, s' \in S
\]

\[1\]It may help to read Poole & Mackworth, 9.5 Decision Processes.
and a chance \( \frac{1}{100} \) of death from relaxing

\[
p'(s, \text{relax, dead}) = \frac{1}{100} \quad \text{for } s \in S
\]

\[
p'(s, \text{relax, } s') = \frac{99p(s, \text{relax, } s')}{100} \quad \text{for } s, s' \in S
\]

and treat death as a sink

\[
p'(\text{dead, } a, \text{ dead}) = 1 \quad \text{for } a \in A
\]

\[
r'(s, a, \text{ dead}) = 0 \quad \text{for } s \in S', a \in A.
\]

Your task is to write a program that given

a positive integer \( n \), a \( \gamma \)-setting \( G \) \( (0 < G < 1) \), and a state \( s \in S' \)

returns the values

\[ q_n(s, \text{ exercise}) \] and \[ q_n(s, \text{ relax}) \]

for \( \gamma = G \) and the revised functions \( p' \) and \( r' \). You may use Python or if you prefer, Prolog.
Sample runs

| ?- show(10,fit,0.5).
n=0 exer:7.2 relax:9.9
n=1 exer:11.632725 relax:14.065425000000001
n=2 exer:13.499449625000001 relax:15.8726068875
n=3 exer:14.3100542525625 relax:16.67890763393752
n=4 exer:14.671913874445043 relax:17.04744552616906
n=5 exer:14.837435044819113 relax:17.219275950873875
n=6 exer:14.91464777255155 relax:17.300660474625666
n=7 exer:14.951231961810453 relax:17.339673626493486
n=8 exer:14.968774522183384 relax:17.3585432373766
n=9 exer:14.977261707407493 relax:17.37227837661356
n=10 exer:14.981394817674818 relax:17.37227837661356

| ?- show(8,unfit,0.8).
n=0 exer:0 relax:4.95
n=1 exer:4.276800000000001 relax:8.8704
n=2 exer:7.49466432 relax:11.9753568
n=3 exer:9.949318967808 relax:14.434482585600001
n=4 exer:11.841350674242356 relax:16.3821102077952
n=5 exer:13.310980618683727 relax:17.9246312845738
n=6 exer:14.458928012466197 relax:19.14630797738245
n=7 exer:15.35923221322587 relax:20.1138759180869
n=8 exer:16.067355537319195 relax:20.880189727124826

| ?- show(10,dead,0.99).
n=0 exer:0 relax:0
n=1 exer:0 relax:0
n=2 exer:0 relax:0
n=3 exer:0 relax:0
n=4 exer:0 relax:0
n=5 exer:0 relax:0
n=6 exer:0 relax:0
n=7 exer:0 relax:0
n=8 exer:0 relax:0
n=9 exer:0 relax:0
n=10 exer:0 relax:0