Problem 1: Probability of Never Losing Consecutive Games in a Season

For this problem, we tried two different approaches. Both yielded the same answer: with independent games and an 80% chance of winning each game, the probability of never losing consecutive games in a season is about 5.881686%.

Terms

From now on, we'll define these variables as such:

- $W = \text{probability of winning a single game}$
- $L = 1 - W$
- $P = \text{probability of never losing consecutive games in an } 82 \text{ game season}$

First Approach: Divide-and-Conquer

For our first approach, we tried to find P by breaking down the set of all seasons with consecutive losses into several smaller disjoint subsets, and then finding the probability for each of those. In our case, we found disjoint subsets with probabilities that summed to 1 - P, and then subtracted our summation from 1 to get P.

With an 82-game season, we split our set into 81 disjoint subsets, numbered 0 to 80. We'll define the ith subset as containing all possible seasons where the Warriors lose consecutively at least once, and their first consecutive losses are games i + 1 and i + 2. So the 0th subset contains all seasons where the Warriors lose their first and second game.

Now that we have these disjoint subsets, we have to solve the easier problem of finding the probability of each one.

The first one is easy, since it's just the probability that the Warriors lose games 1 and 2, which is L_2. The second one is easy as well: WL_2. After this, it gets a bit more complicated.

We'll start by further breaking down each subset based on the number of losses that take place in the first i games. Let's call this quantity j. Since none of these j losses can be consecutive, we know that each loss must be followed by a win. So each loss is really a two-game pattern of LW. Thus, the number of ways these first i games can play out is $\binom{(i-j)}{(j)}$, which accounts for each loss being followed by a win.

Then, since we know that there must be j losses and $i - j$ wins, the probability of each of these subsets occurring is $W_{i-j} * L_j * L_2$ (the L^2 is for the consecutive losses).

If we multiply the number of ways those first i games with j losses can happen by the probability of each individual event, then we get the total probability of the Warriors' losing their first back-to-back games after i games. So all we need to do is sum each of these subsets with j going from 0 to i and i going from 0 to 80 to give us the total probability of the Warriors losing back-to-back games at all.

Then we subtract from 1 to get the probability that the Warriors don't lose any consecutive games.

All together, it looks like this:

$$1 - L_2 \sum_{i=0}^{80} \sum_{j=0}^{i} \binom{i-j}{j} * W_{i-j} * L_j$$

Our Python code, which computes the same thing, looks like this:

```python
# begin choose implementation

import operator as op
def ncr(n, r):
    if r > n:
        return 0
    r = min(r, n - r)
    if r == 0:
        return 1
    numer = reduce(op.mul, xrange(n, n - r, -1))
    denom = reduce(op.mul, xrange(1, r + 1))
    return numer // denom

# end choose implementation


games = 82
win = 0.8
loss = 1 - win

total = 0
for i in range(games - 1):
    for j in range(i + 1):
        total += ncr(i - j, j) * win ** (i - j) * loss ** (j)

print 1 - (total * (loss ** 2))
```

Second Approach: Recurrence

We also came up with a second approach.
For this one, we adopt a dynamic programming approach.

Definition:

$\$dp[i][j] = \text{the number of ways to create win-loss sequences of length i with j wins}$

Base Case:

$dp[1][0] = |L|$
$dp[1][1] = |W|$
$dp[2][1] = |WL, LW|$

$dp[2][2] = |WW|$

Recurrence Relation

$dp[i][j+1] = dp[i-1][j] + dp[i-2][j]$

Finally, at the end, we sum up the probability of all of these possibilities occurring by noting that the probability of each particular sequence occurring is equal to $.8_{\text{\# of wins}} * .2_{\text{\# of losses}}$.

That code looks like this:

```
ans = {k: defaultdict(int) for k in range(1, 82 + 1)}
ans[1][0], ans[1][1], ans[2][1], ans[2][2] = 1, 1, 2, 1
for i in range(3, 82 + 1):
    for j in range(0, 82):
        ans[i][j + 1] += ans[i - 1][j]
        ans[i][j + 1] += ans[i - 2][j]


result = 0
for k, v in ans[82].items():
    result += v * (win_prob**k) * ((1 - win_prob)**(82 - k))
print(result)
```