Analyzing the two options: how much is each option worth today?

Option 1: $1,000,000 today

Assume interest rate of p% \( p > 0 \)

Option 2: \( S0k \) year 0 \( S0k \) year 1 \( S0k \) year 2 \( S0k \) year 3 \( \ldots \) \( S0k \) year \( n \)

Present value = \( S0k + \frac{S0k}{(1+p)} + \frac{S0k}{(1+p)^2} + \frac{S0k}{(1+p)^3} + \ldots + \frac{S0k}{(1+p)^n} \)

= \( S0k \left[ 1 + \frac{1}{(1+p)} + \frac{1}{(1+p)^2} + \ldots + \frac{1}{(1+p)^n} \right] \)

Let \( r = \frac{1}{1+p} \)

= \( S0k \left[ 1 + r + r^2 + r^3 + \ldots + r^n \right] \) geometric series \( r < 1 \)

Summation: \( \sum_{i=0}^{n} r^i \)
Analyzing the two options: how much is each option worth today?

\[ \sum_{i=0}^{n} r^i = S = 1 + r + r^2 + \ldots + r^{n-1} + r^n \]

multiply by \(-r\)
then add two equations

\[ -rS = -r - r^2 - r^3 + \ldots - r^n - r^{n+1} \]

\[ S - rS = 1 - r^{n+1} \]

\[ S = \frac{1 - r^{n+1}}{1 - r} = \sum_{i=0}^{n} r^i \]

take $1M$!

let \( p = 8\% \)
\( n = 100 \)

option 2: \( 50k \left[ \frac{1 - r^{101}}{1 - r} \right] \geq 675k \), \( r \approx \frac{1}{1.08} \approx \frac{1}{1 + 0.08} \)

what if \( n \) is huge?
\( r^{n+1} \to 0 \)
we have \( \frac{1}{1-r} \)
Counting the number of operations in matrix multiplication.

In general, we care about arithmetic (+, -, *, /) and comparison operators (>, <).

```python
def multiply_matrix_vector(A: list[list[float]], x: list[float]) -> list[float]:
    """Computes the matrix-vector multiplication A \times x = b""
    m = len(A) # number of rows, also num. entries in b
    n = len(x) # number of columns of A, also num. entries in x
    b = [0 for _ in range(m)] # initialize b to zero
    for i in range(m):
        assert len(A[i]) == n
        for j in range(n):
            b[i] = b[i] + A[i][j] * x[j]
    return b
```

\[
\text{# operations} = \sum_{i=1}^{m} \sum_{j=1}^{n} 2 = \sum_{i=1}^{m} 2n = 2n \sum_{i=1}^{m} 2 = 2mn
\]

when \(m,n\) get very big, we might not care about \(1\)
Counting the number of operations in a graph algorithm.

In general, we care about arithmetic (+, -, *, /) and comparison operators (>, <).

def count_edges(adj_matrix: list[list[int]]) -> int:
    """Counts the number of edges in a graph (V, E) given its adjacency matrix."""
    n = len(adj_matrix)  # this is |V|
    s = 0
    for i in range(n):
        for j in range(i):
            s = s + adj_matrix[i][j]
    return s
Exercise 1:

Compute a closed-form expression for: \[\sum_{i=1}^{n} \sum_{j=1}^{n} (i + j) = \frac{n}{2} \sum_{i=1}^{n} i + \frac{n}{2} \sum_{j=1}^{n} j\]

Hint: \(\sum_{i=1}^{n} i = \frac{n(n+1)}{2}\).

1. \[\sum_{i=1}^{n} \sum_{j=1}^{n} i = \sum_{i=1}^{n} \left(i + i + i + \cdots + i\right) = \sum_{i=1}^{n} i \cdot i = n \sum_{i=1}^{n} i = n \frac{n(n+1)}{2}\]

2. \[\sum_{i=1}^{n} \sum_{j=1}^{n} j = \sum_{i=1}^{n} \frac{n(n+1)}{2} = \frac{n(n+1)}{2} \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \cdot \frac{n(n+1)}{2}\]

\(1 + 2 = n^2(n+1)\)