House Robber

Think of it as "for each house, pick it or skip it" while never taking neighbors.

How to model it:

Let dp[i] be the max money from the first i houses (i.e. considering nums[0..i-1]).

Decision at house i-1:

- Skip it → value dp[i-1].
- Rob it → you can't take the previous one, so value dp[i-2] + nums[i-1].

Recurrence: dp[i] = max(dp[i-1], dp[i-2] + nums[i-1]).

Base: dp[0] = 0

dp[1] = nums[0]

Answer: dp[n] where n = nums.size().

Space optimization (recommended):

You only need the last two states:

- Keep two variables: prev2 = dp[i-2], prev1 = dp[i-1].
- For each house x, new = max(prev1, prev2 + x), then shift prev2 = prev1, prev1 = new.