P3. Carriers

[[홍원표]]

Important Facts:

- M → Maximum Weight of Shopping Bag
- A carrier can ONLY CARRY 2 bags AT MOST
- Sum of the 2 Bags <= M ← → Maximum Possible Sum of 2 Bags = M
- Undefined # of Bags

Implementation:

- To get the maximum sum of 2 bags (which is M), if one bag has weight of N, the other must be M-N. → Let us call these corresponding bags as A and B.
  A + B = N + (M - N) = M
- As Inputs of Bags come in, check whether the corresponding bags already exist.
  If it doesn't exist, save the bag.
  If exists, instead of saving the bag, get rid of the corresponding bag that already existed.
  This means a maximum-weight-carrying-man has been born! Let's add +1 to # of carriers.
- After inputs are over, the # of bag pairs of A and B will either be {x, 0} or {0, x} ←(x is an arbitrary number)
  Check from M to M/2. If # of bags of weight y is 0, # of bags of weight M-y is x. ← (y is arbitrary number)
  Add that x as a value that holds Lower Half.
  If # of bags of weight y is x, # of bags of weight M-y is 0.
  Check the value of Lower Half. if x > # Lower Half, make # Lower Half as 0, and add x to total # of carrier.
  if x < # Lower Half, decrement x from # Lower Half, add x to total # of carrier, then x as 0.
  [Key Fact: As we are checking from bigger weight to smaller weight, the corresponding bags will be checked from smaller to bigger weight. This lets # Lower Half to be simply added without problem.]
- When the Checking is finished, Add the remaining # Lower Half by (# Lower Half / 2)
  [Key Fact: Any Addition of two Lower Half Weight < M.
  So, simply dividing # Lower Half by 2 and add the remainder, if exists.

0	1	...	M-1	M
	4			
	3			

Assume that # of Bag of weight 1 was already saved as 4.
If the next input is M-1, instead of adding the input to M-1, decrement # of Bag of weight 1. resulting 3.

This Algorithm can be applied to Coin to Dollar Machine, assuming Coin # is Undefined.

# of Bags → X. Max Weight → M
Time Complexity: O(X + M/2)
Space Complexity: O(M)