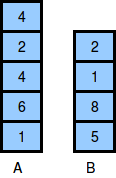
SI: Stacks

Alexa has two stacks of **non-negative integers**, stack A and stack B. Alexa challenges Nick to play the following game:

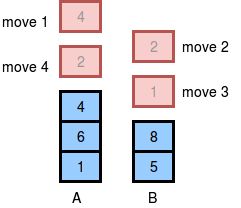
* In each move, Nick can remove one integer from the top of either stack.
* Nick keeps a running sum of the integers he removes from the two stacks.
* Nick is disqualified from the game if, at any point, his running sum becomes greater than some integer given at the beginning of the game, represented as x.
* Nick's *final score* is the total number of integers he has removed from the two stacks.

Given A, B, x, find the maximum possible score Nick can achieve (i.e., the maximum number of integers he can remove without being disqualified).

**Example:** x = 10 and the two stacks initially look like this:



The image below depicts the integers Nick should choose to remove from the stacks. We print as our answer, because that is the maximum number of integers that can be removed from the two stacks without the sum exceeding x. The output should be 4 in this case since he can remove a maximum of 4 blocks before exceeding 10.



You can do this using recursive backtracking but try to do it iteratively (i.e. using for loops). The only data structures you can use are two stacks (two in addition to the ones you already have).

**Hint:** Don’t just pick the lowest block on the top of each stack each time, that doesn’t work every time.

public int bestScore(Stack<Integer> s1, Stack<Integer> s2, int limit) {