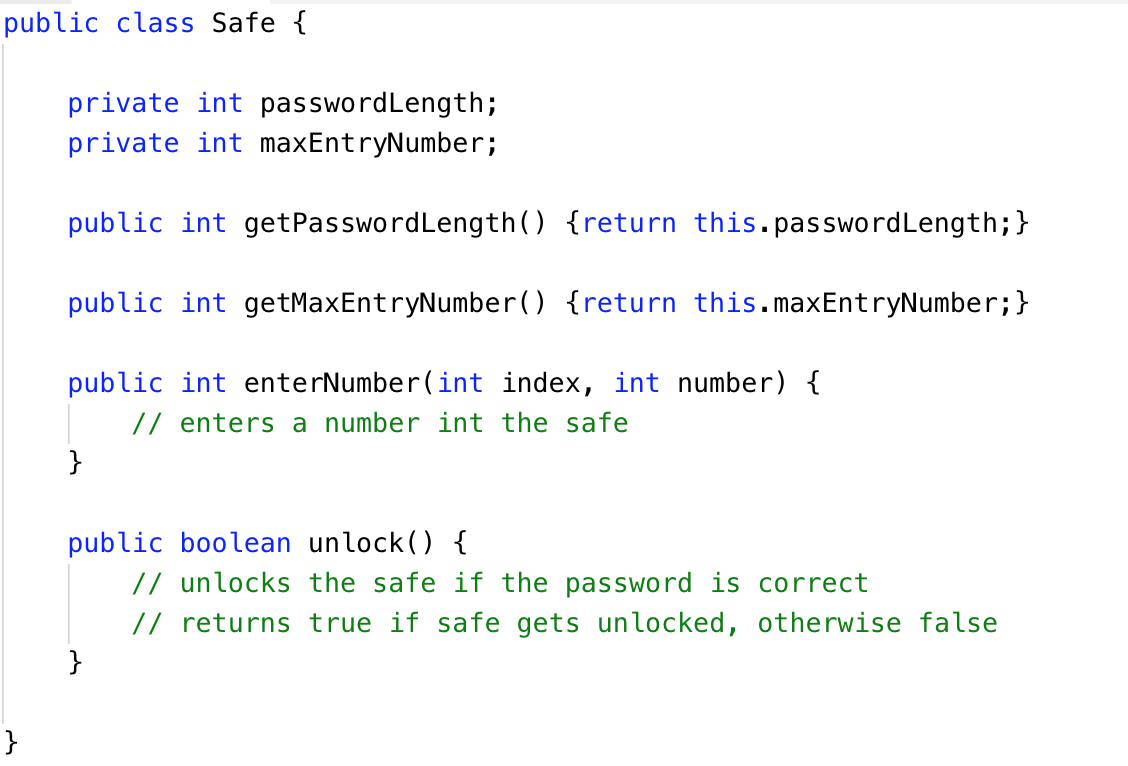
Recursive backtracking

You dropped out of UT computer science to become a professional thief. Unfortunately you do not know how to crack safes, but you do remember recursive backtracking techniques! Write a program to crack a safe combination of arbitrary length. You are given the following safe class:



The password length is the number of numbers required to open the safe. The max entry number is the largest number you can guess (so possible guesses are from 0 to maxEntryNumber). For example, the following combinations are valid (but might not be the correct combination) if passwordLength = 3 and maxEntryNumber = 60:

23-60-0

1-2-3

60-60-60

23-54-12

0-0-0

While the following are not valid

23-53 (need to be length 3, not 2)

-12-23-60 (no negative numbers)

4-23-61 (must all be less than 60)

The combination starts at 0-0-….-0 (passwordLength 0’s). With our same situation (passwordLength = 3 and maxEntryNumber = 60), we would start at 0-0-0. If we call enterNumber(0, 50) then our combination would be 50-0-0.

Complete the crackSafe method below. The password length will always be greater than 0. The max entry number will always be bigger than 0. crackSafe returns the combination as a list of ints where the first element is the first number in the combination and so on. So if our combination is 5-23-56 then crackSafe should return [5,23,56]. Remember to make a helper method with custom arguments.

public List<Integer> crackSafe(Safe safe) {