Halloween Lights[1]

UCD Computer Science Halloween Programming Competition 2023

Donald Knuth has prepared his home for the visit of the Halloween trick or treaters, with a big box of sweets inside the door and ghoulishly scary lights hanging outside in the garden. He gets bored waiting, so starts to play with the lights to pass the time. He has *n* lights in the garden and a switchboard in his living room with *n* switches. Each light can be either on or off, and pressing the *i*-th switch toggles the state of light *i*.

Donald makes up this game. Initially all lights are off. He then makes a sequence of moves, where each move involves selecting a (possibly empty) set of switches and pressing them, thus inverting the states of the corresponding lights. The goal of the game is to make exactly *m* moves thereby achieving a final state where the first *v* lights are on and the other lights off. There is one restriction however: the same set of switches cannot be pressed in two different moves.

There are many ways to win this game, and Donald, being Donald, wants to find out how many ways of winning there are. He takes out his laptop, ignores the trick or treaters coming to his door, and codes up a solution. This is your task as well!

(Note that two games are considered the same if, after a reordering of the moves in one of them, at every step the same set of switches is pressed in both of them. For example, if n = 4, m = 3, and v = 2, one possible winning game is obtained by pressing switches 1, 2 and 4 in the first move, 1 and 3 in the second one, and 1, 3 and 4 in the last one. This is considered equivalent to, say, first pressing 1 and 3; then 1, 2, 4; and then 1, 3, 4.)

Input

Your program should read three integer inputs from the command line:

n: the total number of lights (1..100)
m: the number of moves (1..100)
v: the number of lights that should be on at the end of the game (0..*n*)

Output

A single integer representing the number of ways the game can be won. Mod the result with 10,567,201 in case it gets too big. Here are two cases to use for testing:

Sample Input	Corresponding Output
3 3 1	7
6 4 0	10416

Submission details: You'll be sent a link to a Google form

Judging: A correct solution, coded cleanly. If no one gets a correct solution, the judges will decide which works best, so if you're stuck just implement something relevant (like an animation using asterixes that shows the game being played), and you never know!

Appeals: Appeals written in blood on a pumpkin to be delivered by midnight.

---

[1] Thanks to David O. for the problem.