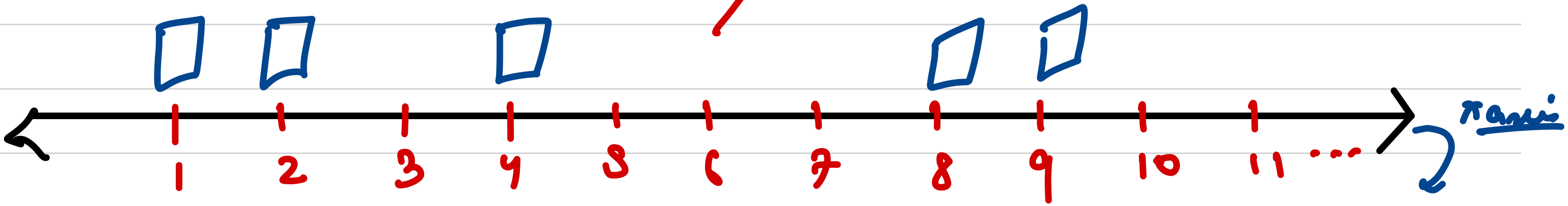
N=5 C=3

[1, 2, 8, 4, 9]

you cannot put a stall on pos 6,5 etc because there are no stalls.



1 stall → 1 cow

min dist

C_1-1 C_2-2, C_3-4 → 1

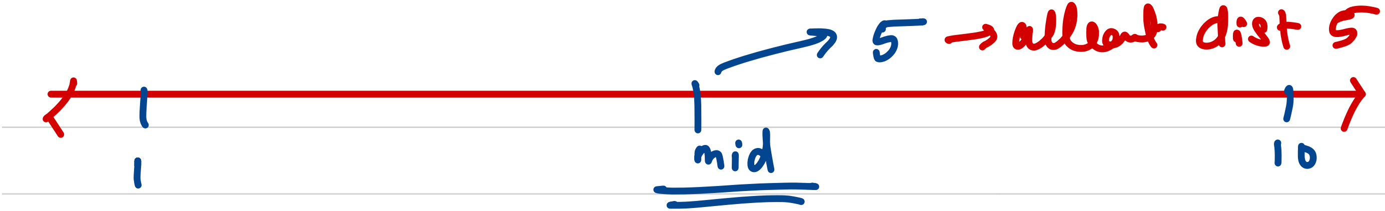
C_1-1 C_2-4, C_3-8 → 3

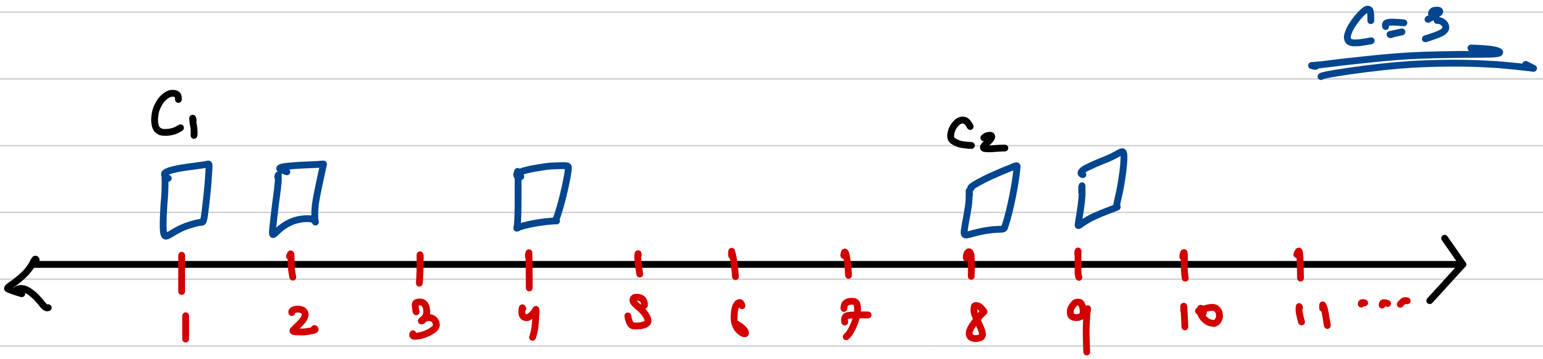
C_1-1 C_2-4, C_3-9 → 3

C_1-1 C_2-8, C_3-9 → 1

max

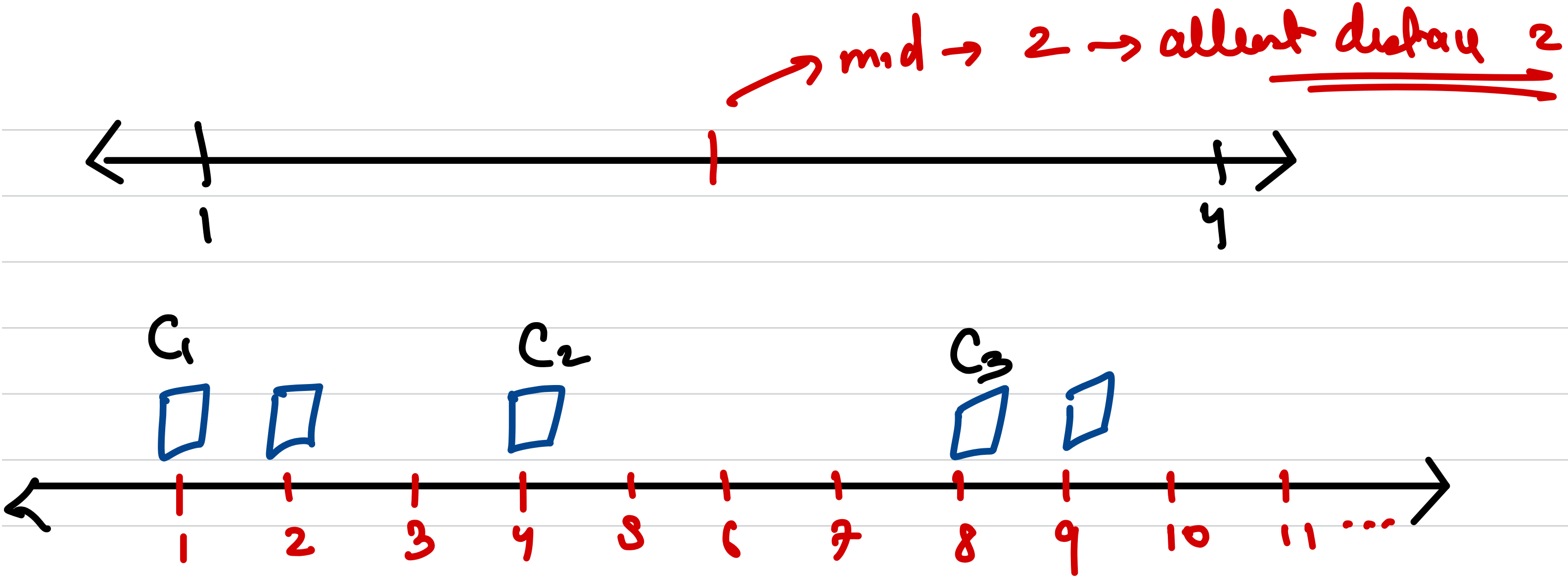
stalls of diff config



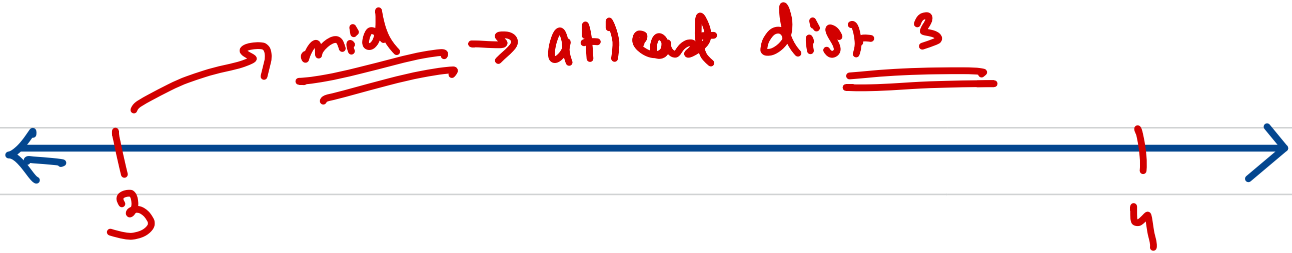


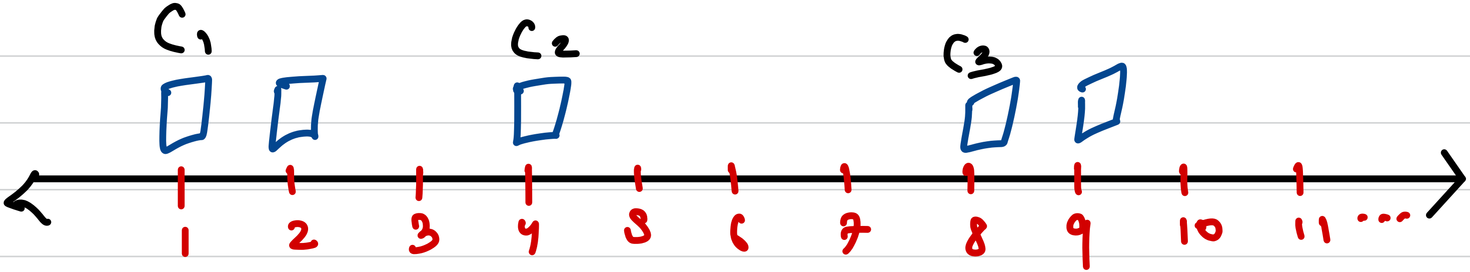
In this config we can't place cows.

If we can't maintain min dist 5, then anything greater than 5 is also not possible. → Move Left



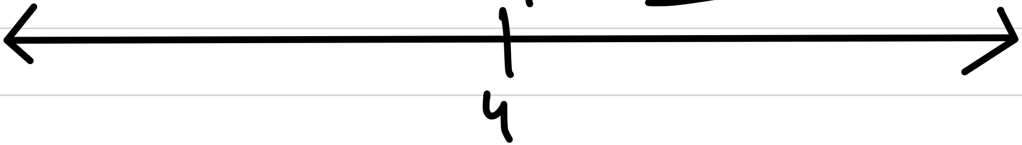
So with atleast dist 2 between any 2 cows we can place all the cows. So we can place it in less than 2 also. → Move Right.

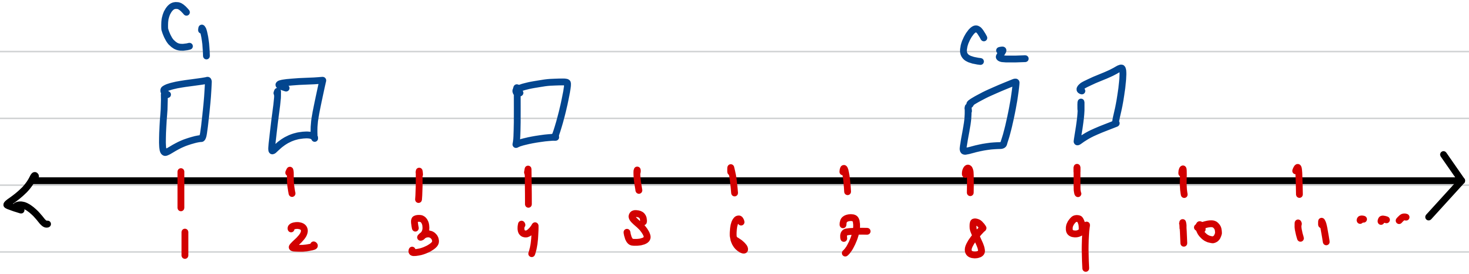




We are able to do in 3, so → move right.

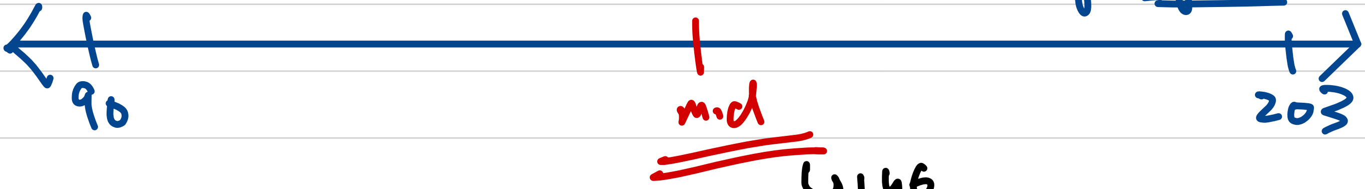
mid → atleast dist 4.





↳ invalid

atmost a student can get mid no. of pages



↳ 146

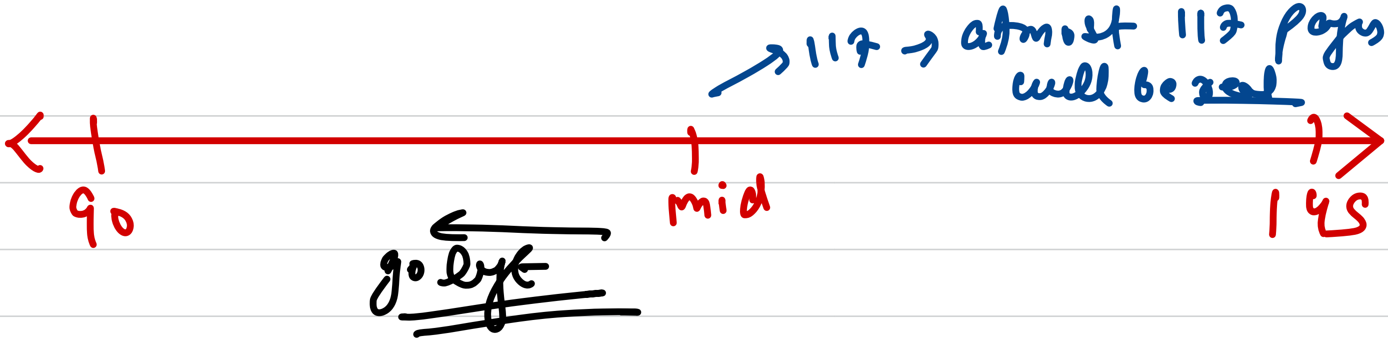
12, 34, 67, 90

↑ S_2

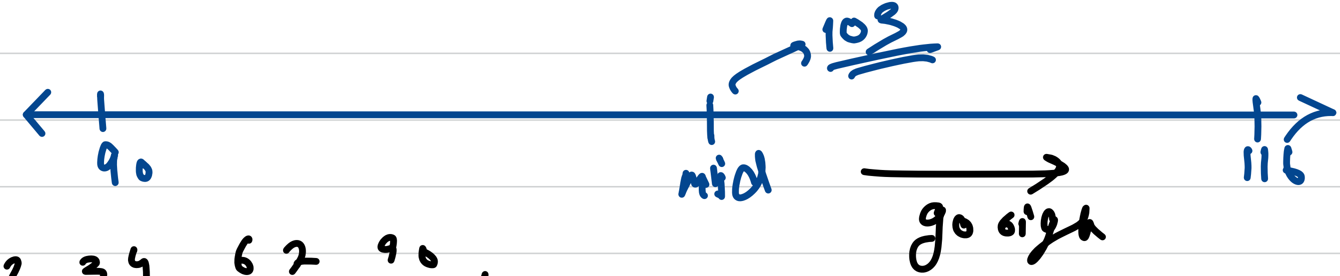
$S_1 = \underbrace{12 + 34 + 67}$

$S_2 = 90$

if we can do in 146, we want even less value

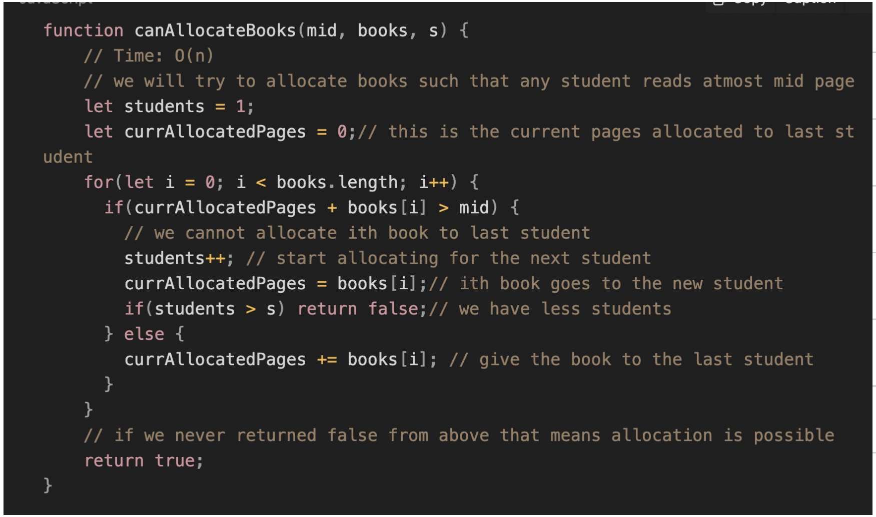


$$\underbrace{12, 34, 67}_{S_1}, \underbrace{90}_{S_2}$$



$$\underbrace{12\ 34}_{S_1}, \underbrace{67, 90}_{S_2}$$

mid = 119



```
function canAllocateBooks(mid, books, s) {
    // Time: O(n)
    // we will try to allocate books such that any student reads atmost mid page
    let students = 1;
    let currAllocatedPages = 0;// this is the current pages allocated to last student
    for(let i = 0; i < books.length; i++) {
      if(currAllocatedPages + books[i] > mid) {
        // we cannot allocate ith book to last student
        students++; // start allocating for the next student
        currAllocatedPages = books[i];// ith book goes to the new student
        if(students > s) return false;// we have less students
      } else {
        currAllocatedPages += books[i]; // give the book to the last student
      }
    }
    // if we never returned false from above that means allocation is possible
    return true;
}
```

Students = ~~1~~ 2

currAllocated Page = 90

i = ~~0~~ ~~1~~ ~~2~~ ~~3~~ 4

12, 34, 67, 90

↑
i