<https://programmers.co.kr/learn/courses/30/lessons/42586>

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | package programmers.c30.e42586;  import java.util.ArrayList;  import java.util.Arrays;  public class Main1 {  static class Solution {  public int[] solution(int[] progress, int[] speed) {  ArrayList<Integer> answer = new ArrayList<>();  // 진행중인 작업의 마감일과 작업 수  int 마감일 = (int)Math.ceil((100.0 - progress[0]) / speed[0]), 작업수 = 1;  for (int i = 1; i < progress.length; ++i) {  int 현재마감일 = (int)Math.ceil((100.0 - progress[i]) / speed[i]);  if (현재마감일 <= 마감일) ++작업수;  else {  answer.add(작업수);  마감일 = 현재마감일;  작업수 = 1;  }  }  answer.add(작업수);  return answer.stream().mapToInt(i -> i).toArray();  }  }  public static void main(String[] args) {  int[][] progress = {{93, 30, 55}, {95, 90, 99, 99, 80, 99}};  int[][] speed = {{1, 30, 5}, {1, 1, 1, 1, 1, 1}};  Solution sol = new Solution();  for (int i = 0; i < progress.length; ++i)  System.out.println(Arrays.toString(sol.solution(progress[i], speed[i])));  }  } |

|  |
| --- |
| ArrayList<Integer> answer = new ArrayList<>();  // 진행중인 작업의 마감일과 작업 수  // 첫 작업이 진행중이다.  // 마감일 계산식은 (100 - 진척도) / 속도 이고, 이 식의 결과값을 올림해야 한다.  // 올림계산: Math.ceil(값)  // 그런데 정수 나눗셈은 소수점 아래를 버려버리기 때문에, 실수 나눗셈을 해야 한다.  // 계산식의 숫자 하나가 실수이면, 실수 나눗셈을 하게 된다. 그래서 100.0 이다.  int 마감일 = (int)Math.ceil((100.0 - progress[0]) / speed[0]), 작업수 = 1;  for (int i = 1; i < progress.length; ++i) {  int 현재마감일 = (int)Math.ceil((100.0 - progress[i]) / speed[i]);  if (현재마감일 <= 마감일) ++작업수; // i 번째 작업 마감일이 그 이전 작업들의 마감일보다 빠르다면  // 그 이전 작업이 마감될 때까지 대기해야 한다  else { // i 번째 작업 마감일이 그 이전 작업들의 마감일보다 크다면  answer.add(작업수); // 그 이전 작업들이 먼저 종료되고,  마감일 = 현재마감일; // 현재 작업만 마감일까지 대기하게 된다  작업수 = 1;  }  }  answer.add(작업수);  // ArrayList<Integer> 객체로부터 int[] 배열을 만들어 리턴한다  return answer.stream().mapToInt(i -> i).toArray(); |