

# Report

프로그래머스 기능개발 문제(42586)



과목 : 코딩테스트지도

담당교수 : 이승진 교수님

학부 : IT융합자율학부

학번 : 202114136

이름 : 장준희

제출일 : 2025-03-30

작업 git url: <https://github.com/jjune960/coding-assignments/>

## 문제 설명

프로그래머스 팀에서는 기능 개선 작업을 수행 중입니다. 각 기능은 진도가 100%일 때 서비스에 반영할 수 있습니다.

또, 각 기능의 개발속도는 모두 다르기 때문에 뒤에 있는 기능이 앞에 있는 기능보다 먼저 개발될 수 있고, 이때 뒤에 있는 기능은 앞에 있는 기능이 배포될 때 함께 배포됩니다.

먼저 배포되어야 하는 순서대로 작업의 진도가 적힌 정수 배열 progresses와 각 작업의 개발 속도가 적힌 정수 배열 speeds가 주어질 때 각 배포마다 몇 개의 기능이 배포되는지를 return 하도록 solution 함수를 완성하세요.

제한 사항

- 작업의 개수(progresses, speeds배열의 길이)는 100개 이하입니다.
- 작업 진도는 100 미만의 자연수입니다.
- 작업 속도는 100 이하의 자연수입니다.
- 배포는 하루에 한 번만 할 수 있으며, 하루의 끝에 이루어진다고 가정합니다. 예를 들어 진도율이 95%인 작업의 개발 속도가 하루에 4%라면 배포는 2일 뒤에 이루어집니다.

## 요구 분석

각 기능은 진도가 100%일 때 배포된다. 단 뒤에 있는 기능은 앞에 있는 기능이 배포될 때 함께 배포된다.

각 배포마다 몇 개의 기능이 배포되는지를 return

## 1차 시도

먼저 수작업을 통해서 몇 개의 기능이 배포되는지를 확인해 보았다. (스캔 파일 참조)

그 결과 앞의 값부터 먼저 검사하고 인덱스를 이동하며 검사한다는 것을 확인하였다.

## 1. 모든 원소를 speeds 만큼 더하기

먼저 모든 원소를 더하는 것이 필요하다.

코드 및 실행결과:

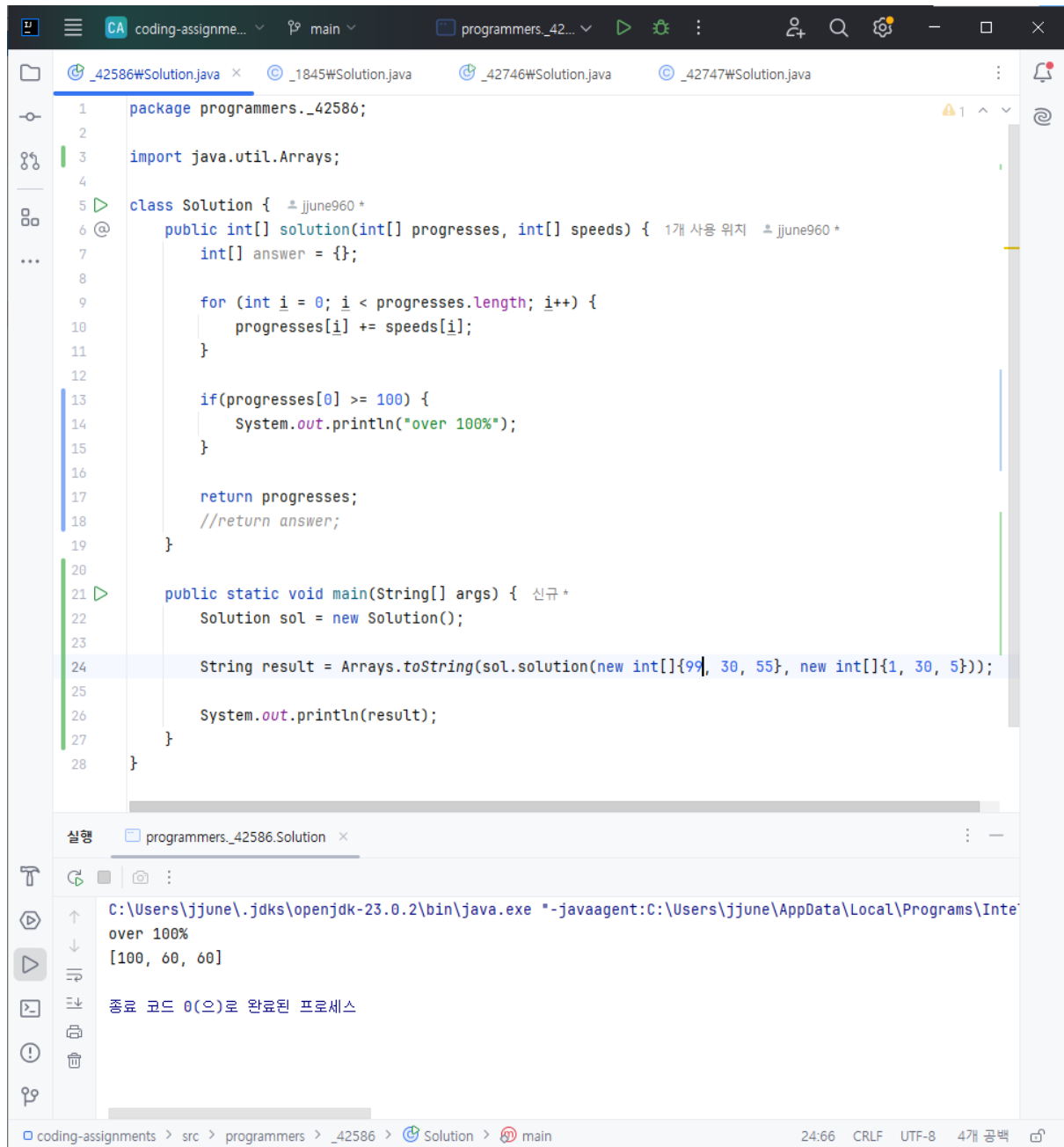
Main.java	Output
<pre>1 import java.util.Arrays; 2 3 class Main { 4 5     public int[] solution(int[]         progresses, int[] speeds) { 6         for (int i = 0; i &lt;             progresses.length; i++) 7             progresses[i] +=                 speeds[i]; 8         } 9         return progresses; 10    } 11 12    public static void main(String[]         args) { 13        Main sol = new Main(); 14 15        String result = Arrays             .toString(sol.solution                 (new int[]{93, 30, 55},                 new int[]{1, 30, 5})); 16 17        System.out.println(result); 18    } 19 }</pre>	<pre>[94, 60, 60]  === Code Execution Successful ===</pre>

이와 같이 모든 원소가 더해진 것을 확인할 수 있다.

## 2. 첫번째 원소 검사하기

첫번째 원소를 검사하여 기능개발이 완료되었는지 확인한다.

실행결과



The screenshot shows an IDE with a Java file named `_42586#Solution.java`. The code defines a `Solution` class with a `solution` method that increments the first element of the `progresses` array by the first element of the `speeds` array. It also includes a `main` method that calls `solution` with specific arrays and prints the result. The execution output at the bottom shows the command used, the output `over 100%` and `[100, 60, 60]`, and a message indicating successful completion.

```
package programmers._42586;

import java.util.Arrays;

class Solution {
    public int[] solution(int[] progresses, int[] speeds) {
        int[] answer = {};

        for (int i = 0; i < progresses.length; i++) {
            progresses[i] += speeds[i];
        }

        if(progresses[0] >= 100) {
            System.out.println("over 100%");
        }

        return progresses;
        //return answer;
    }

    public static void main(String[] args) {
        Solution sol = new Solution();

        String result = Arrays.toString(sol.solution(new int[]{99, 30, 55}, new int[]{1, 30, 5}));

        System.out.println(result);
    }
}
```

실행 결과:

```
C:\Users\jjune\.jdk\openjdk-23.0.2\bin\java.exe "-javaagent:C:\Users\jjune\AppData\Local\Programs\IntelliJ IDEA\bin\idea-agent.jar" -jar C:\Users\jjune\AppData\Local\Programs\IntelliJ IDEA\bin\idea.jar
over 100%
[100, 60, 60]
종료 코드 0(으)로 완료된 프로세스
```

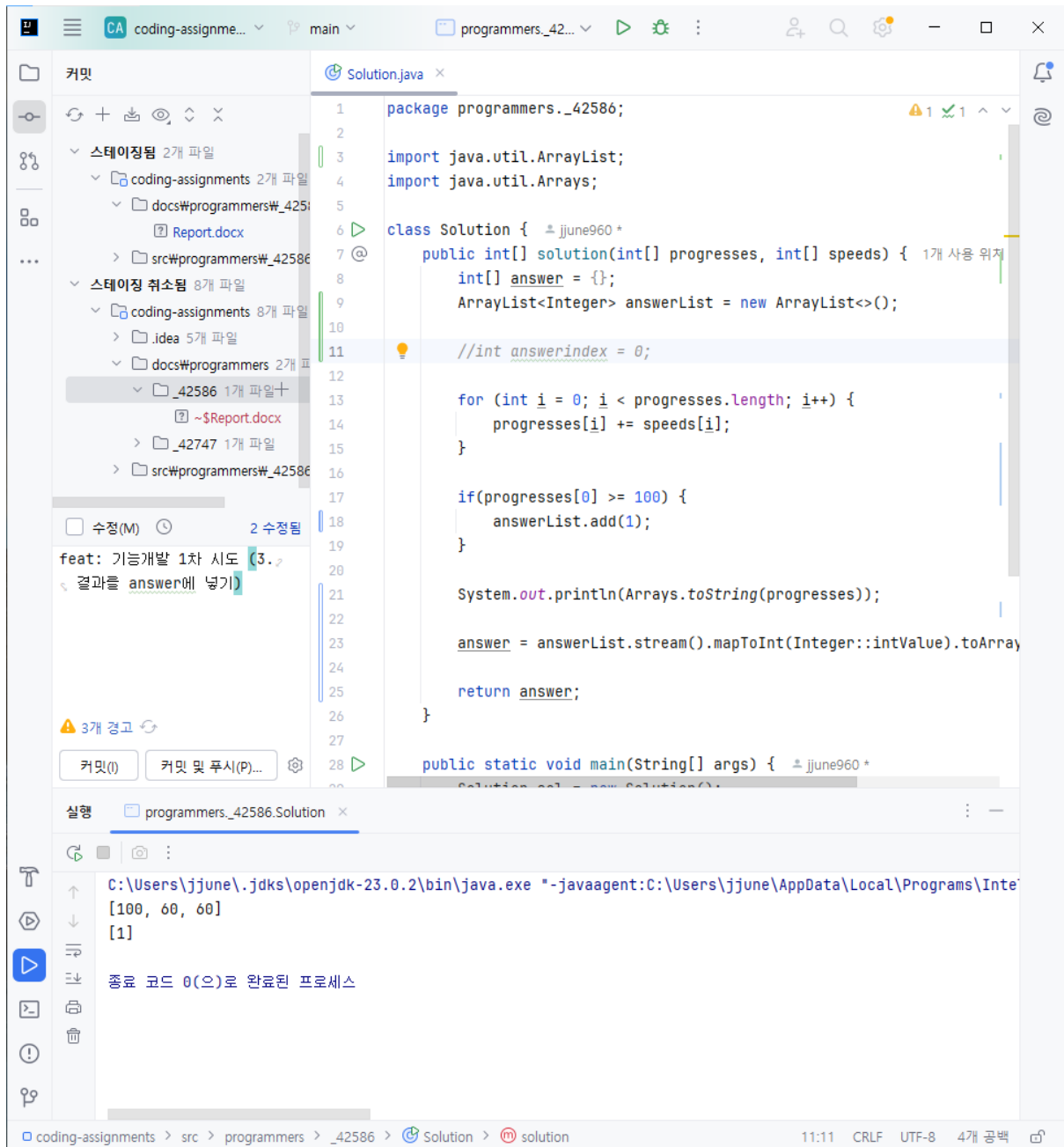
100% 이상일때의 처리를 하였다.

3. 결과를 answer에 넣기

작업에 대한 내역을 answer에 넣어보겠다.

arrayList를 통해서 넣을 것이다. (배열이 가변적이기 때문)

실행결과



The screenshot shows an IDE window with a project named 'coding-assignment...'. The left sidebar shows a file explorer with a directory structure including 'src#programmers#\_42586'. The main editor displays a Java file named 'Solution.java' with the following code:

```
1 package programmers._42586;
2
3 import java.util.ArrayList;
4 import java.util.Arrays;
5
6 class Solution {
7     public int[] solution(int[] progresses, int[] speeds) {
8         int[] answer = {};
9         ArrayList<Integer> answerList = new ArrayList<>();
10
11         //int answerindex = 0;
12
13         for (int i = 0; i < progresses.length; i++) {
14             progresses[i] += speeds[i];
15         }
16
17         if(progresses[0] >= 100) {
18             answerList.add(1);
19         }
20
21         System.out.println(Arrays.toString(progresses));
22
23         answer = answerList.stream().mapToInt(Integer::intValue).toArray();
24
25         return answer;
26     }
27
28     public static void main(String[] args) {
29         Solution sol = new Solution();
30         int[] progresses = {100, 60, 60};
31         int[] speeds = {1, 2, 2};
32         int[] answer = sol.solution(progresses, speeds);
33         System.out.println(Arrays.toString(answer));
34     }
35 }
```

The bottom panel shows the execution output for the program 'programmers\_42586.Solution':

```
C:\Users\jjune\jdk\openjdk-23.0.2\bin\java.exe "-javaagent:C:\Users\jjune\AppData\Local\Programs\IntelliJ IDEA\lib\idea_rt.jar=1111:C:\Users\jjune\AppData\Local\Programs\IntelliJ IDEA\bin" C:\Users\jjune\IdeaProjects\coding-assignment\src\programmers\_42586\Solution.java
[100, 60, 60]
[1]
```

The output shows the array [100, 60, 60] and the result [1], indicating that the first element is confirmed to be 100 or more, and the count is 1.

첫번째 원소를 확인하여 개수를 answer에 카운트 하였다.

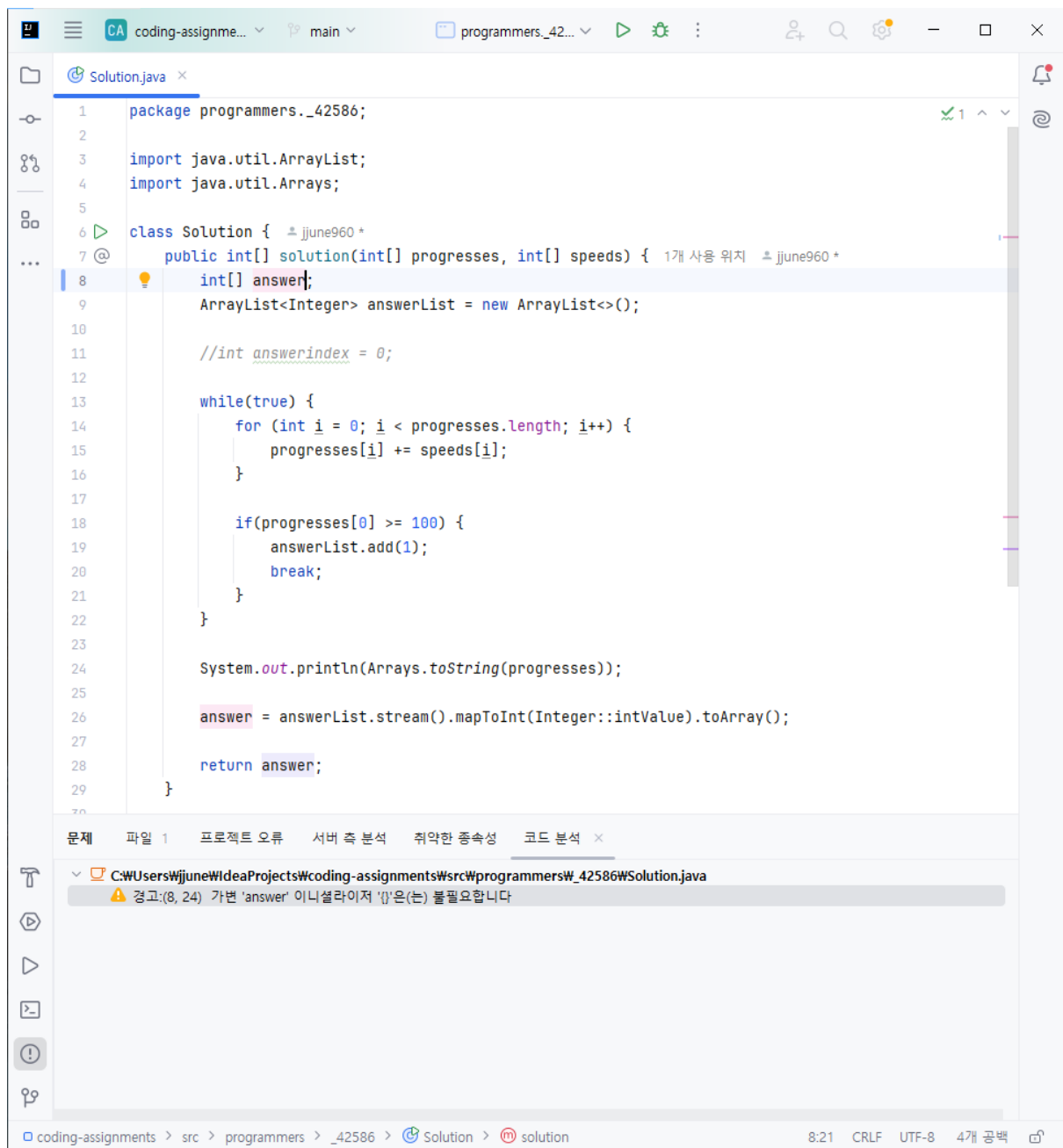
#### 4. 반복해서 더하기

이제 이러한 검사를 모든 원소에 반복해서 해야 한다.

그렇다면 본질적으로 이 문제가 무엇을 요구하는 지를 파악해야 한다.

이 문제는 모든 원소를 계속 speeds 만큼 더한 뒤 조건을 따져야 하는 문제이다.

While 문을 이용하면 계속 더할 것이다.



```
1 package programmers._42586;
2
3 import java.util.ArrayList;
4 import java.util.Arrays;
5
6 class Solution {
7     public int[] solution(int[] progresses, int[] speeds) {
8         int[] answer;
9         ArrayList<Integer> answerList = new ArrayList<>();
10
11         //int answerindex = 0;
12
13         while(true) {
14             for (int i = 0; i < progresses.length; i++) {
15                 progresses[i] += speeds[i];
16             }
17
18             if(progresses[0] >= 100) {
19                 answerList.add(1);
20                 break;
21             }
22         }
23
24         System.out.println(Arrays.toString(progresses));
25
26         answer = answerList.stream().mapToInt(Integer::intValue).toArray();
27
28         return answer;
29     }
30 }
```

문제 파일 1 프로젝트 오류 서버 측 분석 취약한 종속성 코드 분석 ×

C:\Users\Wjjune\IdeaProjects\coding-assignments\src\programmers\42586\Solution.java

경고:(8, 24) 가변 'answer' 이니셜라이저 '값'은(는) 불필요합니다

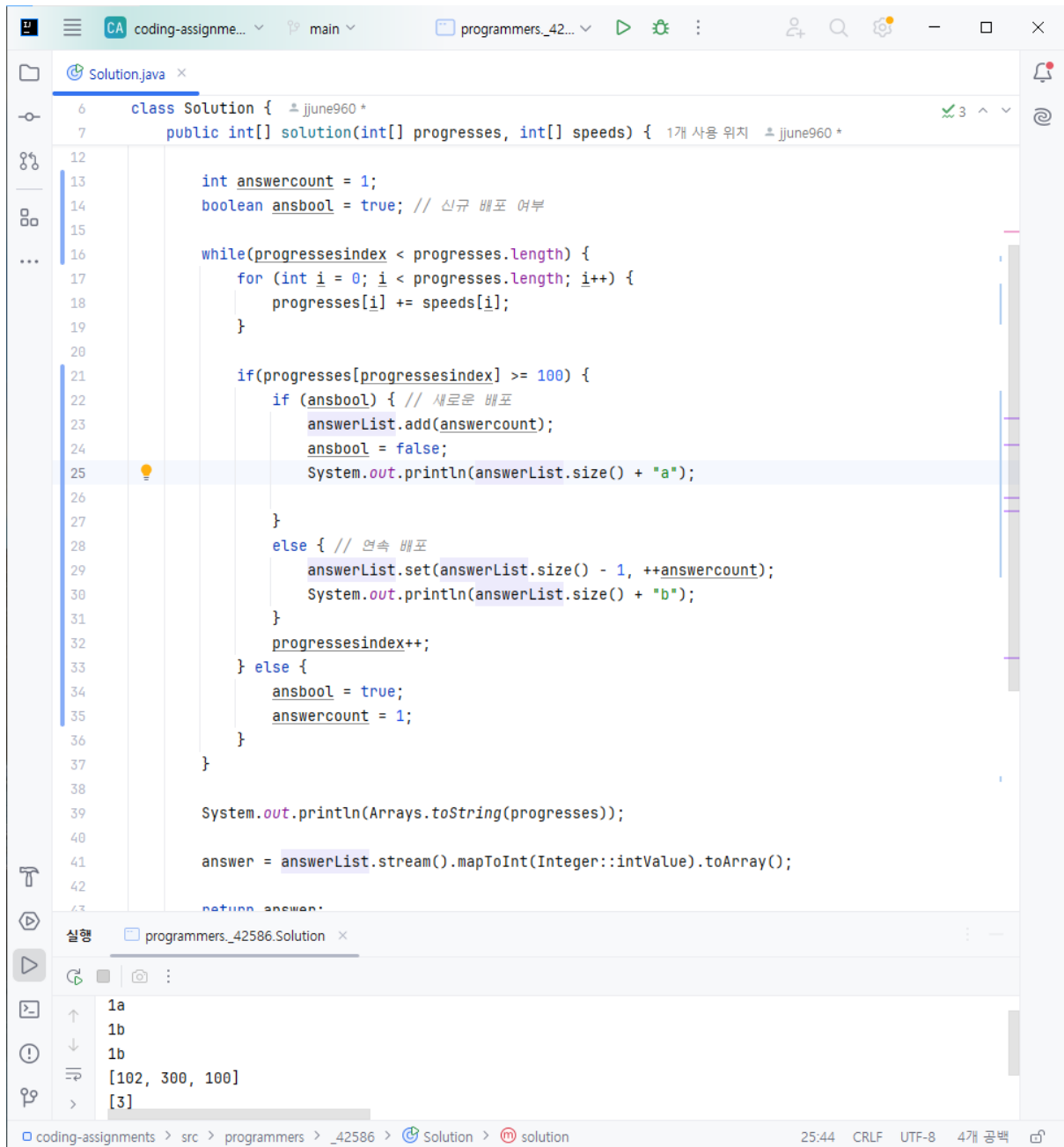
coding-assignments > src > programmers > \_42586 > Solution > solution 8:21 CRLF UTF-8 4개 공백

break문으로 첫번째 문이 감지되면 멈추게 구현하였다.

## 5. 반복해서 검사하기 (마지막엔 끝내기) (실패)

반복해서 검사하기 위해서는 현재 검사중인 index를 확인하는 것이 필요하다.

0번부터 시작해서 마지막 index까지 검사하면 된다.



```
class Solution {
    public int[] solution(int[] progresses, int[] speeds) {

        int answercount = 1;
        boolean ansbool = true; // 신규 배포 여부

        while(progressesindex < progresses.length) {
            for (int i = 0; i < progresses.length; i++) {
                progresses[i] += speeds[i];
            }

            if(progresses[progressesindex] >= 100) {
                if (ansbool) { // 새로운 배포
                    answerList.add(answercount);
                    ansbool = false;
                    System.out.println(answerList.size() + "a");
                }
                else { // 연속 배포
                    answerList.set(answerList.size() - 1, ++answercount);
                    System.out.println(answerList.size() + "b");
                }
                progressesindex++;
            } else {
                ansbool = true;
                answercount = 1;
            }
        }

        System.out.println(Arrays.toString(progresses));

        answer = answerList.stream().mapToInt(Integer::intValue).toArray();

        return answer;
    }
}
```

실행 결과:

```
1a
1b
1b
[102, 300, 100]
[3]
```

이 코드는 문제가 있다.

배포의 여부가 잘못 계산된다. (새로운 배포와 연속 배포의 순서 오류)

## 실패 원인 찾기

Progresses = [93, 30, 55], speeds = [1, 30, 5]에서의 문제를 분석하겠다.

이 로직에서는 첫번째 원소때만 새로운 배포가 되고 다음부터는 연속배포가 되는 문제를 발견하였다.

즉 answerList의 첫번째 원소에만 3이 들어간 것이다. (의도는 [2,1])

문제의 원인을 생각한 결과 다음이 추정된다.

1. 원소 증가 코드를 배포 계산 중간에도 더하는 문제

수정한 코드

```
if(ansbool) {  
    for (int i = 0; i < progresses.length; i++) {  
        progresses[i] += speeds[i];  
    }  
}
```

결과 (Progresses = [93, 30, 55], speeds = [1, 30, 5])

[2, 1]

제대로 작동한다.

## 배운 점









처음에는 배포의 카운트에만 집중했다.

하지만 반복을 하는 동안에 또 다른 연산이 실행되는 부분을 간과한 것 같다.

앞으로는 모든 연산이 미치는 영향을 면밀히 검토하여 다양한 특면을 고려한 코드를 작성할 것이다.



github commit hash

Commits on Mar 29, 2025		
docs: 기능개발 배운 점	jjune960 committed 1 minute ago	98d9beb  <>
feat: 기능개발 1차 시도 (실패 원인 찾기 & 수정) (성공)	jjune960 committed 9 minutes ago	c7312dc  <>
feat: 기능개발 1차 시도 (5. 반복해서 검사하기 (마지막엔 끝내기) (실패))	jjune960 committed 48 minutes ago	856b567  <>
feat: 기능개발 1차 시도 (4. 반복해서 더하기)	jjune960 committed 2 hours ago	6aad2cf  <>
feat: 기능개발 1차 시도 (3. 결과를 answer에 넣기)	jjune960 committed 2 hours ago	aeb5f7b  <>
Commits on Mar 28, 2025		
feat: 기능개발 1차 시도 (2. 첫번째 원소 검사하기)	jjune960 committed yesterday	6045bad  <>
feat: 기능개발 1차 시도 (1. 모든 원소를 speeds 만큼 더하기)	jjune960 committed yesterday	2209a30  <>
Commits on Mar 27, 2025		
Create _42586(Function development) library and docs folder and file	jjune960 committed 2 days ago	8bffbf8  <>