Java

```java
class Solution {
    public List<List<Integer>> findRLEArray(int[][] encoded1, int[][] encoded2) {

    }
}
```

JavaScript

```javascript
/**
 * @param {number[][]} encoded1
 * @param {number[][]} encoded2
 * @return {number[][]}
 */
var findRLEArray = function(encoded1, encoded2) {

};
```

TypeScript

```typescript
function findRLEArray(encoded1: number[][], encoded2: number[][]): number[][] {

};
```

C++

```cpp
class Solution {
```

```
public:
    vector<vector<int>> findRLEArray(vector<vector<int>>& encoded1, vector<vector<int>>& encoded2) {

    }
};
```

C#

```
public class Solution {
    public IList<IList<int>> FindRLEArray(int[][] encoded1, int[][] encoded2) {

    }
}
```

Kotlin

```
class Solution {
    fun findRLEArray(encoded1: Array<IntArray>, encoded2: Array<IntArray>): List<List<Int>> {

    }
}
```
