

Kids with the Greatest Number of Candies

Given the array `candies` and the integer `extraCandies`, where `candies[i]` represents the number of candies that the *ith* kid has.

For each kid check if there is a way to distribute `extraCandies` among the kids such that he or she can have the **greatest** number of candies among them. Notice that multiple kids can have the **greatest** number of candies.

Example 1:

Input: `candies = [2,3,5,1,3]`, `extraCandies = 3`

Output: `[true,true,true,false,true]`

Explanation:

Kid 1 has 2 candies and if he or she receives all extra candies (3) will have 5 candies --- the greatest number of candies among the kids.

Kid 2 has 3 candies and if he or she receives at least 2 extra candies will have the greatest number of candies among the kids.

Kid 3 has 5 candies and this is already the greatest number of candies among the kids.

Kid 4 has 1 candy and even if he or she receives all extra candies will only have 4 candies.

Kid 5 has 3 candies and if he or she receives at least 2 extra candies will have the greatest number of candies among the kids

Example 2:

Input: `candies = [4,2,1,1,2]`, `extraCandies = 1`

Output: `[true,false,false,false,false]`

Explanation: There is only 1 extra candy, therefore only the kid 1 will have the greatest number of candies among the kids regardless of who takes the extra candy.

Example 3:

Input: `candies = [12,1,12]`, `extraCandies = 10`

Output: `[true,false,true]`

Constraints:

- `2 <= candies.length <= 100`
- `1 <= candies[i] <= 100`
- `1 <= extraCandies <= 50`

Solution:

```
class Solution{
    public List<Boolean> kidsWithCandies(int[] candies, int extraCandies) {
        List<Boolean> output = new ArrayList<>();
        int max = 0;
        for(int i=0; i < candies.length; i++) {
            if (candies[i] > max) {
                max = candies[i];
            }
        }

        for(int i = 0; i < candies.length; i++) {
            if(candies[i] + extraCandies >= max) {
                output.add(i,true);
            }
            else {
                output.add(i,false);
            }
        }
        return output;
    }
}
```

Success Details >

Runtime: 1 ms, faster than 56.56% of Java online submissions for Kids With the Greatest Number of Candies.

Memory Usage: 39.4 MB, less than 75.06% of Java online submissions for Kids With the Greatest Number of Candies.

Next challenges: [Triangle](#) [Find the Duplicate Number](#) [Fixed Point](#)

Show off your acceptance: [f](#) [t](#) [in](#)

Time Submitted	Status	Runtime	Memory	Language
a few seconds ago	Accepted	1 ms	39.4 MB	java

```
1 * class Solution{
2 *     public List<Boolean> kidsWithCandies(int[] candies, int extraCandies) {
3 *         List<Boolean> output = new ArrayList<>();
4 *         int max = 0;
5 *         for(int i=0; i < candies.length; i++) {
6 *             if (candies[i] > max) {
7 *                 max = candies[i];
8 *             }
9 *         }
10 *
11 *         for(int i = 0; i < candies.length; i++) {
12 *             if(candies[i] + extraCandies >= max) {
13 *                 output.add(i,true);
14 *             }
15 *             else {
16 *                 output.add(i,false);
17 *             }
18 *         }
19 *         return output;
20 *     }
21 * }
```

Testcase Run Code Result Debugger

Accepted Runtime: 0 ms

Your input: [2,3,5,1,3]
3

Output: [true,true,true,false,true]

Expected: [true,true,true,false,true]

Diff

Run Code Submit
