



Demystifying the **General Coding Assessment**

Presented by



Agenda

Introduction

Who are we, what we do?

What's GCA?

What's General Coding Assessment (GCA) and how does it work?

What does it take to implement the hardest question on GCA?

An interactive session of solving the 4th question on GCA

What's next?

Ready to take the test? Want to practice more? We'll show you how and answer any questions you all may have.

Introduction

What's GCA?

What's GCA?

- Industry standard assessment for early talent professionals
- Designed to help companies #GoBeyondResumes
- Measures general implementation and problem solving ability regardless of the coding language of choice



General Coding Framework

Duration: 1 hour, 10 minutes

Skills

implementation • data structures • problem solving

Roles

software developer • software engineer



LEARN MORE

SELECT

GCA Structure

Part 1

Part 1 – GCA

Question 1

✓ Can Include

- Tasks that require a combination of 2 to 3 basic concepts. For example:
 - Iterating over an array and taking into account some condition.
 - Splitting a string by some condition.
- Should usually be solvable using one loop.
- The task description should clearly state the implementation steps.

✗ Should Exclude

- Anything that requires noticing or proving a certain pattern.
- Anything that requires optimizing an algorithm.
- Anything that requires a knowledge of classic algorithms.
- Anything that requires good implementation skills.

Example:

Given an array of integers a . The task is to return another array b where $b[i] = a[i - 1] + a[i] + a[i + 1]$, (if an element does not exist it should be 0).

Part 1 - GCA

Question 2

✓ Can Include

- Tasks that require a combination of 3 to 5 basic concepts. For example:
 - Splitting a string into substrings, modifying each substring and comparing with other strings.
 - Iterating over an array to produce two new arrays given some conditions, modifying the second array and appending it to the beginning of the first array.
- Should usually be solvable using one to two nested loops.
- The task description should clearly state the implementation steps.

✗ Should Exclude

- Anything that requires noticing or proving a certain pattern.
- Anything that requires optimizing an algorithm.
- Anything that requires a knowledge of classic algorithms.

Example:

Given a list of words (consisting of lowercase English letters) and a complex word written in camelCase (consisting of English letters), check if the complex word consists of words from the given list.

Part 1 – GCA

Question 3

✓ Can Include

- Implementing a specific comparator for strings.
- Implementing a specific merge function for arrays.
- Other implementation challenges that clearly explain what needs to be done and require translating the instructions into code.

✗ Should Exclude

- Anything that requires noticing or proving a certain pattern.
- Anything that requires optimizing an algorithm with advanced data structures like binary indexed trees, etc.
- Anything from special topics like graphs, number theory, or dynamic programming.

Example:

Given two strings, merge them with the following merge function: instead of comparing the characters in the usual lexicographical order, compare them based on how many times they occur in their respective strings. Fewer occurrences mean the character is considered smaller; in case of equality, compare them lexicographically; in case of equality, take the character from the first string.

Part 1 – GCA

Question 4

✓ Can Include

- Tasks that require noticing an application of a certain algorithm, data-structure or technique.
- Optimizing some queries with the help of data structures like hashmaps or sets.
- Algorithms on trees like finding the longest path of a tree.

✗ Should Exclude

- Brain teasers or tasks that require specialized knowledge of certain libraries or advanced algorithms like Dijkstra, Kruskal, FFT, etc.
- Tasks that require very long implementation. Unlike the second task, implementation is not key here; problem-solving is key, thus this task should ideally avoid taking up lots of implementation time.

Example:

Create a data structure that allows the following operations.

insert x y – insert an object with key *x* and value *y*.

get x – return the value of an object with key *x*.

addToKey x – add *x* to all keys in map.

addToValue y – add *y* to all values in map.

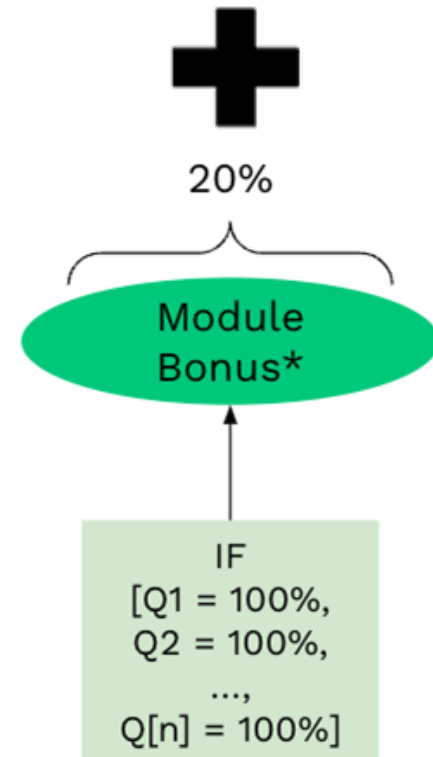
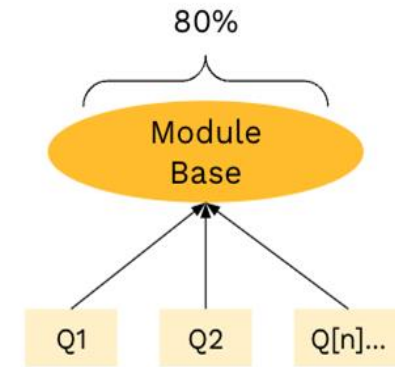
Time complexity of each operation should be $O(\log(N))$

Scoring

Part 2

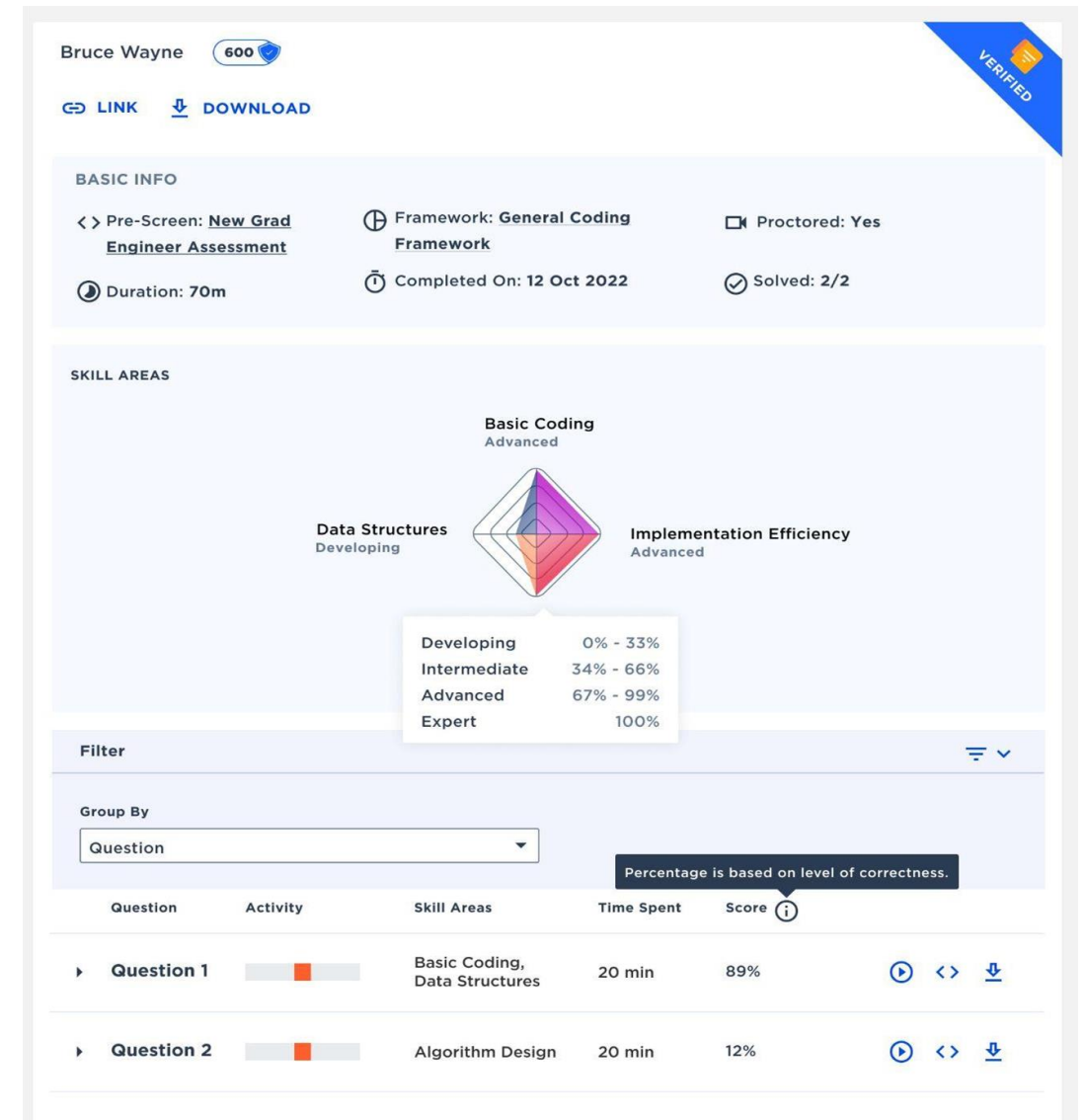
Coding Score

- Coding Score is the main high-level output of a General Coding Assessment
- Made up of a two-tiered scoring system that provides an overall score and a detailed breakdown of skill proficiencies
- 1st tier (80%): base points gained by successfully completing questions
- 2nd tier (20%): bonus points awarded based on performance of skill areas, only if question is solved 100% successfully
- Functional correctness is the main contributing factor to your score
- Minimum threshold varies greatly depending on the company



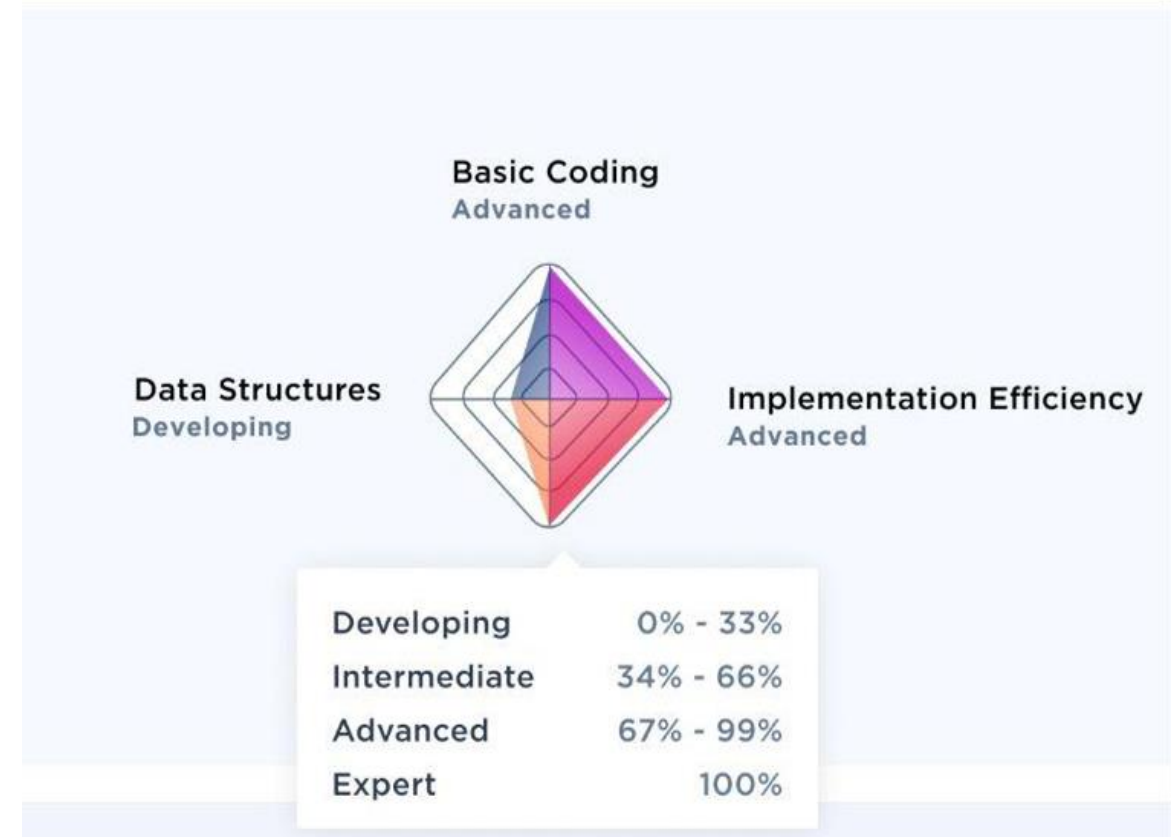
Coding Score - cont.

- While you can skip questions, going in order is usually a sound strategy
- Only 15% of test takers are able to get into the 530+ range
- If, for whatever reason, you hit a wall you wouldn't be penalized for it



Skill Areas

- Additionally, GCA results offers a visual representation and qualitative feedback on the individual skills measured in the assessment
- The skill areas measured in the GCA are
 - Basic Coding
 - Data Manipulation
 - Ease of Implementation
 - Problem-Solving
- Each skill area will be defined as developing, intermediate, advanced, or expert



The Flow

The Flow

- You can choose your programming language before starting the test
- You may look up language documentation and syntax online during the test, but other materials are prohibited

General Coding Assessment

**1 hour, 10 minutes duration**

You cannot pause the test after starting.

**Submit your task when finished**

It's the green button on the bottom right.

**Complete 4 tasks**

4 coding.

**Answer questions in any order**

Solve tasks in any order, switch any time.

NEXT

The Flow – contd.

- Automated proctoring adds an extra level of security to ensure the integrity of the test
- Please make sure you have sufficient time and space and read the pre-requisite requirements before you begin



1. Grab your photo ID

Have your government-issued photo ID ready to show (like driver's license or passport).



2. Prepare your computer

Make sure your computer is equipped with the following:

- **Chrome** or **Firefox**
- A reliable internet connection
- Webcam and microphone



3. Make time and space

Reserve 1 hour and 10 minutes to take the test somewhere quiet and undisturbed.



4. Share your webcam and desktop

- When prompted, share your entire desktop screen as well as your webcam.
- Do not end video or screensharing until after you have finished the test.

[TAKE IT LATER](#)

[LET'S GO!](#)

The Flow – contd.



- Watch the timer! 70 minutes from the moment you click start
- You'll have 1 minute of EXTRA time to submit any code, if you haven't submitted it yet
- Running code doesn't affect your score, so run it as much as you need

General Coding Assessment

Duration: **70m** Questions: **4** Submitted: **0**

1:09:51

Remaining Time

Question	Type	Score	Submitted
Question 1	Single-Function	0/300	 Unsubmitted VIEW
Question 2	Single-Function	0/300	 Unsubmitted VIEW
Question 3	Single-Function	0/300	 Unsubmitted VIEW
Question 4	Single-Function	0/300	 Unsubmitted VIEW

Implementing GCA Q4

Q4 - sumDivisibleByK

You are given an array of integers a and an integer k . Your task is to calculate the number of ways to pick two different indices $i < j$, such that $a[i] + a[j]$ is divisible by k .

$a =$

1	2	3	4	5
---	---	---	---	---

 $k = 3$

Example

For $a = [1, 2, 3, 4, 5]$ and $k = 3$, the output should be $\text{sumsDivisibleByK}(a, k) = 4$.

There are 4 pairs of numbers that sum to a multiple of $k = 3$:

- $a[0] + a[1] = 1 + 2 = 3$
- $a[0] + a[4] = 1 + 5 = 6$
- $a[1] + a[3] = 2 + 4 = 6$
- $a[3] + a[4] = 4 + 5 = 9$

$$a[0] + a[1] = 1 + 2 = 3 \quad \text{Yes}$$

$$a[0] + a[2] = 1 + 3 = 4$$

$$a[0] + a[3] = 1 + 4 = 5$$

$$a[0] + a[4] = 1 + 5 = 6 \quad \text{Yes}$$

$$a[1] + a[2] = 2 + 3 = 5$$

$$a[1] + a[3] = 2 + 4 = 6 \quad \text{Yes}$$

$$a[1] + a[4] = 2 + 5 = 7$$

$$a[2] + a[3] = 3 + 4 = 7$$

$$a[2] + a[4] = 3 + 5 = 8$$

$$a[3] + a[4] = 4 + 5 = 9 \quad \text{Yes}$$

Q4 - sumDivisibleByK - Brute Force Approach

main.py3

```
1 def sumDivisibleByK(a, k):
2     retval = 0
3     for i in range(len(a)):
4         for j in range(i+1, len(a)):
5             if (a[i] + a[j]) % k == 0:
6                 retval = retval + 1
7     return retval
```

TESTS

CUSTOM TESTS

RUN TESTS

⚠ Tests passed: 6/11. Execution time limit exceeded: Program exceeded the execution time limit. Make sure that it completes execution in a few seconds for any possible input.

Test 1

Input: a: [1, 2, 3, 4, 5]
k: 3

Output: 4

Expected Output: 4

Console Output: Empty

Error Output: Empty

Test 2

Test 3

Test 4

Test 5

Test 6

Test 7

Test 8

Test 9

Test 10

Test 11

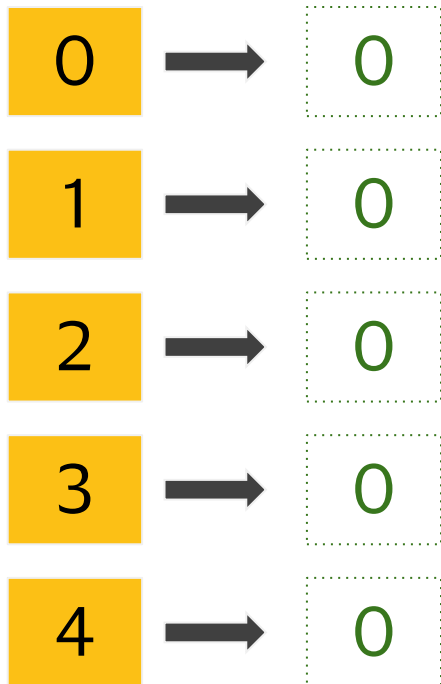
Q4 - sumDivisibleByK - Different Way to Solve Problem?

a =

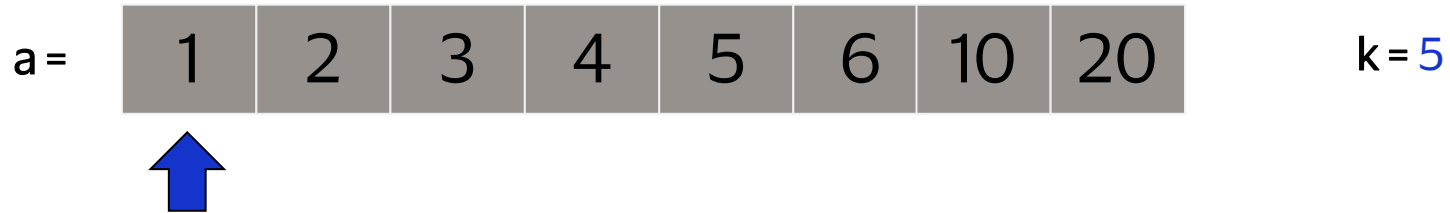
1	2	3	4	5	6	10	20
---	---	---	---	---	---	----	----

 k = 5

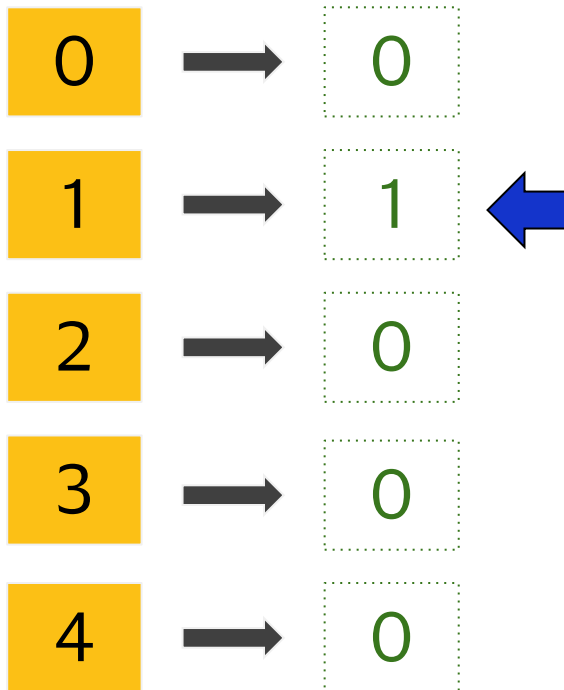
Keeping track of remainders?



Q4 - sumDivisibleByK - Different Way to Solve Problem?



Keeping track of remainders?

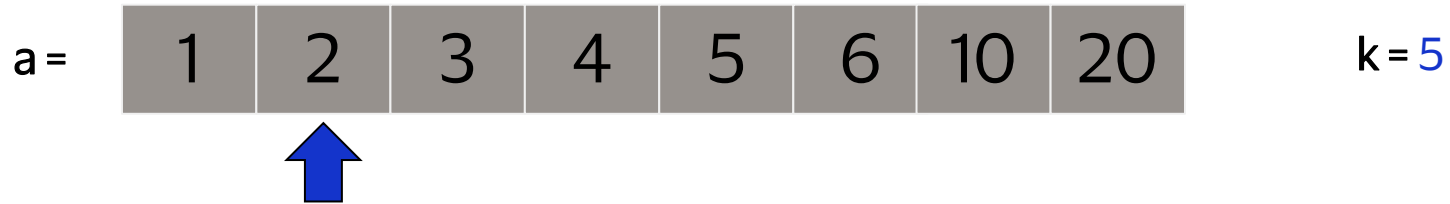


The current element: $a[0] = 1$

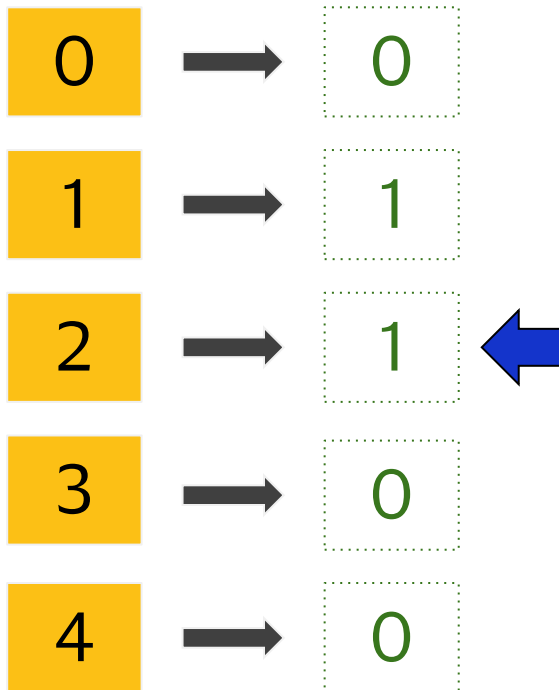
Find reminder to k: $1 \% 5 = 1$

Increase the count for reminder 1

Q4 - sumDivisibleByK - Different Way to Solve Problem?



Keeping track of remainders?



The current element: $a[1] = 2$

Find remainder to k: $2 \% 5 = 2$

Increase the count for remainder 2

Q4 - sumDivisibleByK - Different Way to Solve Problem?

a =

1	2	3	4	5	6	10	20
---	---	---	---	---	---	----	----

 k = 5



Keeping track of remainders?

0	→	0
1	→	1
2	→	1
3	→	1
4	→	0

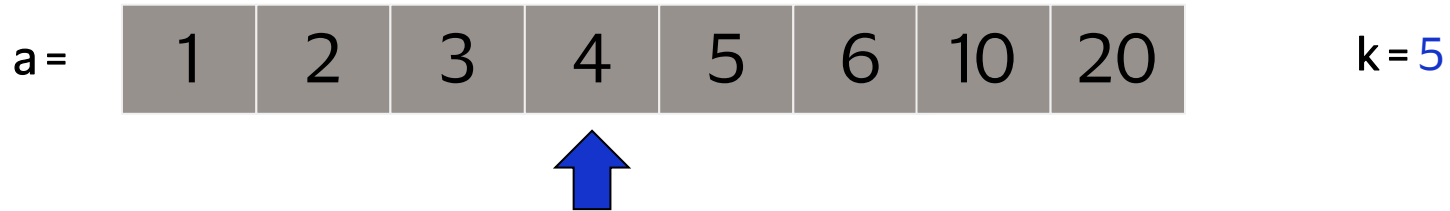


The current element: $a[2] = 3$

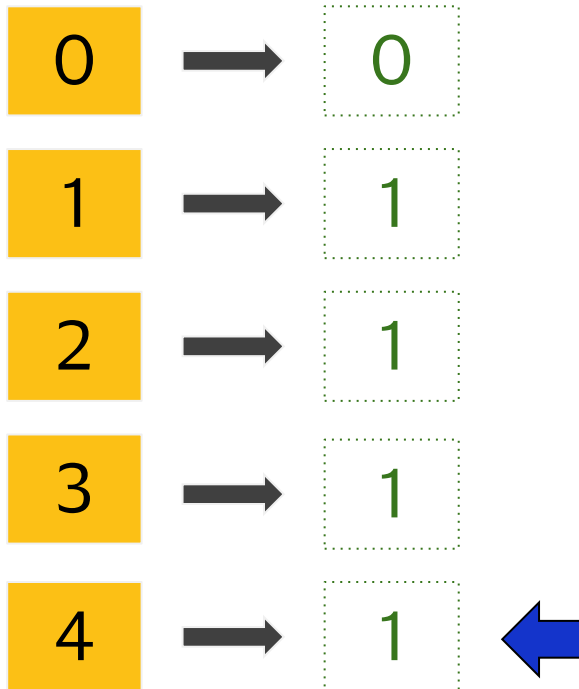
Find reminder to k: $3 \% 5 = 3$

Increase the count for reminder 3

Q4 - sumDivisibleByK - Different Way to Solve Problem?



Keeping track of remainders?

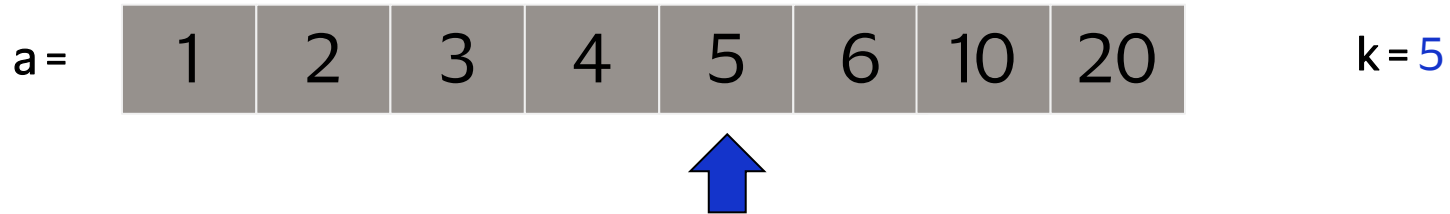


The current element: $a[3] = 4$

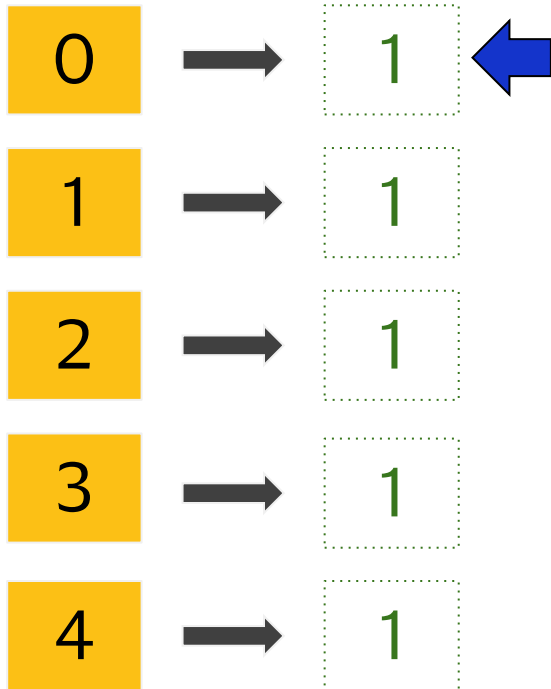
Find remainder to k: $4 \% 5 = 4$

Increase the count for remainder 4

Q4 - sumDivisibleByK - Different Way to Solve Problem?



Keeping track of remainders?

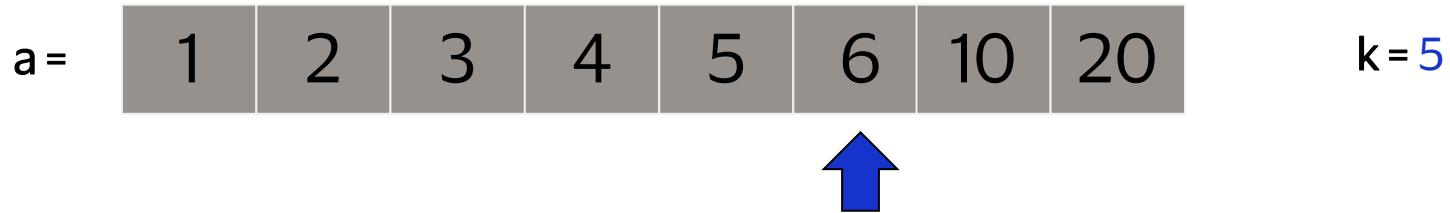


The current element: $a[4] = 5$

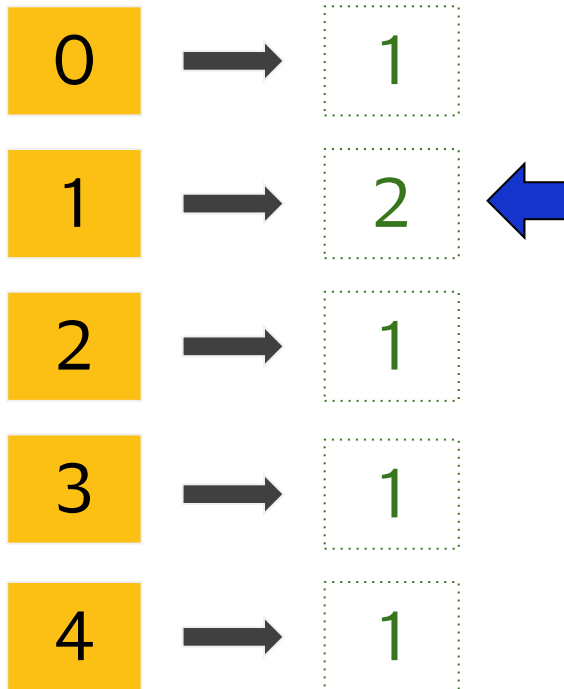
Find remainder to k: $5 \% 5 = 0$

Increase the count for remainder 0

Q4 - sumDivisibleByK - Different Way to Solve Problem?



Keeping track of remainders?

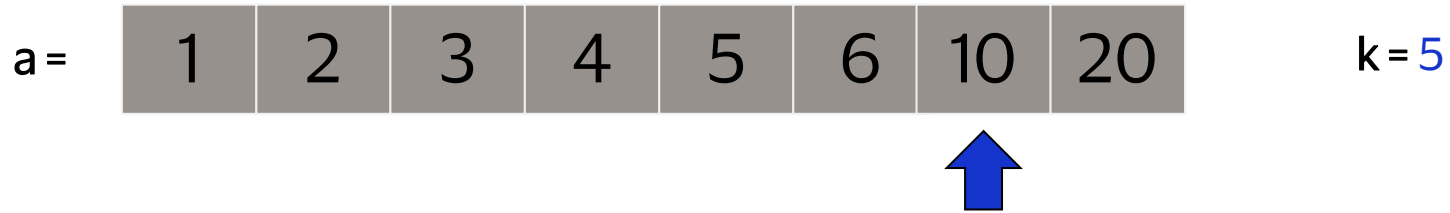


The current element: $a[5] = 6$

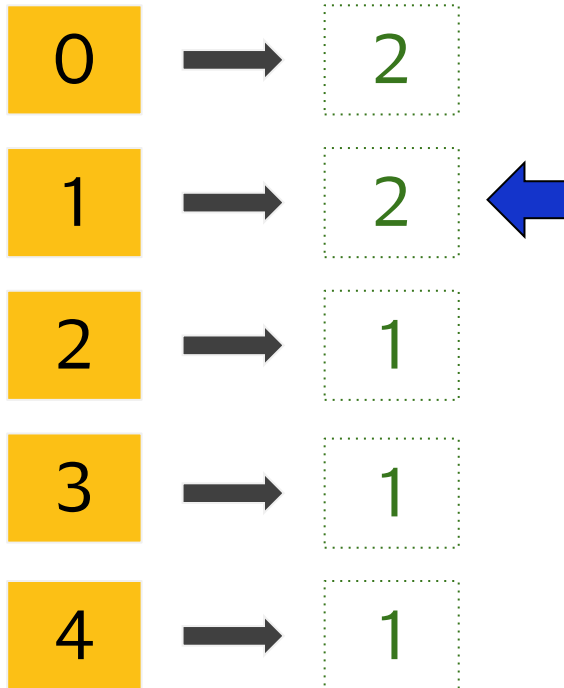
Find reminder to k: $6 \% 5 = 1$

Increase the count for reminder 1

Q4 - sumDivisibleByK - Different Way to Solve Problem?



Keeping track of remainders?

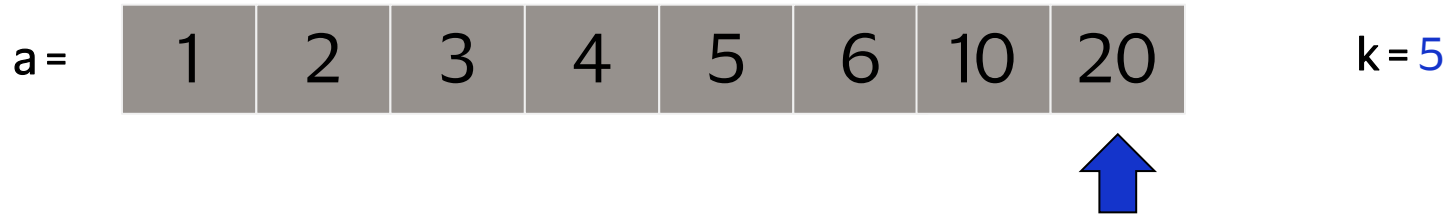


The current element: $a[6] = 10$

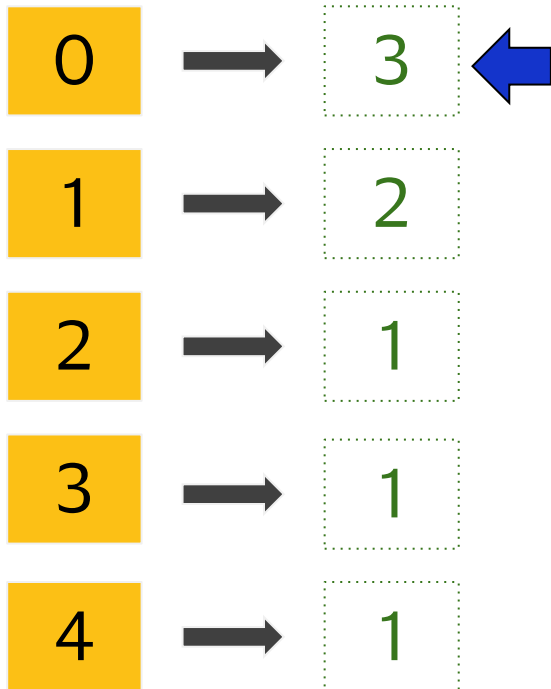
Find reminder to k: $10 \% 5 = 0$

Increase the count for reminder 0

Q4 - sumDivisibleByK - Different Way to Solve Problem?



Keeping track of remainders?



The current element: $a[7] = 20$

Find reminder to k: $20 \% 5 = 0$

Increase the count for reminder 0

Q4 - sumDivisibleByK - Different Way to Solve Problem?

a =

1	2	3	4	5	6	10	20
---	---	---	---	---	---	----	----

 k = 5

Keeping track of remainders?

0	→	3
1	→	2
2	→	1
3	→	1
4	→	1

Sums divisible by k AND product of their remainder count

0	+	0
1	+	4
2	+	3
3	+	2
4	+	1

Remainder count

$$3 * 2 = 6$$

$$2 * 1 = 2$$

$$1 * 1 = 1$$

$$1 * 1 = 1$$

$$1 * 2 = 2$$

Remove double counting...

$$12 / 2 = 6$$

Q4 - sumDivisibleByK - Optimized Approach

main.py3


Saved

```
1 def sumsDivisibleByK(a, k):
2     remainders = {}
3     for num in a:
4         remainders[num % k] = remainders.get(num % k, 0) + 1
5     ans = 0
6     for rem in remainders.keys():
7         if rem == 0 or rem == k - rem:
8             ans += remainders[rem] * (remainders[rem] - 1) // 2
9         else:
10            ans += remainders[rem] * remainders[k - rem]
11    return int(ans / 2)
```

TESTS

CUSTOM TESTS

 RUN TESTS

 Tests passed: 11/11. Click Submit to run the full test suite and save your solution.

▶ Test 1



▶ Test 2



▶ Test 3



▶ Test 4



▶ Test 5



▶ Test 6



▶ Test 7



▶ Test 8



▶ Test 9



▶ Test 10



▶ Test 11



Want to take a practice test?

Scan the QR code for a link to the
Practice Coding Assessment



Q

VISA

&

A