Interview Cake

I have an array of $n + 1$ numbers. Every number in the range $1..n$ appears once except for one number that appears twice.

Write a method for finding the number that appears twice.

Gotchas

We can do this with $O(1)$ additional memory.

Breakdown

To avoid using up extra memory space, lets use some math!

Solution

First, we sum all numbers $1..n$. We can do this using the equation:

$$\frac{n^2+n}{2}$$

because the numbers in $1..n$ are a triangular series.

Second, we sum all numbers in our input array, which should be the same as our other sum but with our repeat number added in twice. So the difference between these two sums is the repeated number!

Java ▾

```java
public static int findRepeat(int[] numbers) {

    if (numbers.length < 2) {
        throw new IllegalArgumentException("Finding duplicate requires at least two numbers");
    }

    int n = numbers.length - 1;

    int sumWithoutDuplicate = (n * n + n) / 2;

    int actualSum = 0;
    for (int number : numbers) {
        actualSum += number;
    }

    return actualSum - sumWithoutDuplicate;
}
```

Complexity

$O(n)$ time. We can sum all the numbers $1..n$ in $O(1)$ time using the fancy formula, but it still takes $O(n)$ time to sum all the numbers in our input array.

$O(1)$ additional space—the only additional space we use is for numbers to hold the sums with and without the repeated value.

Bonus

If our array contains huge numbers or is really long, our sum might be so big it causes an integer overflow. What are some ways to protect against this?

Ready for more?

Check out our full course ➞