Lab 03 Assignment

public static int getLargest(int arr[], int sz) {

    int iterate1 = 0;

    int iterate2 = 0;

    int largest = 0;

    while (iterate1 < sz - 1) {

        iterate2++;

        if (iterate2 == sz) {

            iterate1++;

            iterate2 = iterate1;

            continue;

        }

        int product = arr[iterate1] \* arr[iterate2];

        if (product > largest)

            largest = product;

    }

    return largest;

}

What does it do?

* The algorithm computes the largest product of any two numbers in the array.
* iterate1 runs from 0 to sz-1.
* For each iterate1, iterate2 loops from iterate1+1 to sz.

Number of operations = about n × n = n².

Drop constants & lower terms

* The time complexity is O(n²).
* Space complexity is just a few variables (iterate1, iterate2, largest, product) → O(1).

The Big O runtime complexity is O(n²), and the space complexity is O(1).

Given that:

You have 20 bags.

19 bags contain pieces of 1.0 g each.

1 bag contains pieces of 1.1 g each.

You can only use the scale once.

First label the bags from 1 through and take a different number of candies from each bagf

Label the bags 1 through 20. Take a different number of candies from each bag like 1 candy from Bag 1, 2 candies from Bag 2, 5 candies from Bag 5, 6 candies from Bag 6, and so on…. Put them all together and weigh them.

Now calculate expected vs actual If all bags were normal (1.0 g each), the total weight would be:

1. + 2 + 3 + … + 20 = \frac {20 \times 21}{2} = 210 \text{grams}

If one bag has heavy pieces (1.1 g each), the total weight will be more than 210 g. The extra weight tells you which bag is heavy.

Example: If the weight = 210.3 g → Bag 3 is heavy. If the weight = 210.15 g → Bag 15 is heavy.