Daily Coding Problem #69

Problem

This problem was asked by Facebook.

Given a list of integers, return the largest product that can be made by multiplying any three integers.

For example, if the list is `[-10, -10, 5, 2]`, we should return `500`, since that's `-10 * -10 * 5`.

You can assume the list has at least three integers.

Solution

If all the integers were positive, then we would simply need to take the three largest numbers of the array. Then, we can just sort it and return the last three elements.

However, we need to account for negative numbers in the array. If the largest product that can be made includes a negative number, we would need to have two so as to cancel out the negatives. So, we can take the larger of:

- The three largest numbers
- The two smallest (most negative) numbers, and the largest number

```
def maximum_product_of_three(lst):
    lst.sort()
    third_largest, second_largest, first_largest = lst[-3], lst[-2], lst[-1]
    first_smallest, second_smallest = lst[0], lst[1]
    return max(third_largest * second_largest * first_largest,
    first_largest * first_smallest * second_smallest)
```

This runs in O(N log N) time since we have to sort the input array.

We could also do this in O(N) time by using select or looking for the largest elements manually.

```
from math import inf

def maximum_product_of_three(lst):
    max1, max2, max3, min1, min2 = -inf, -inf, -inf, inf, inf

    for x in lst:
        if x > max1:
            max3 = max2
            max2 = max1
            max1 = x
        elif x > max2:
            max3 = max2
            max2 = x
        elif x > max3:
            max3 = x

        if x < min1:
            min2 = min1

            min1 = x
        elif x < min2:
            min2 = x

    return max(max1 * max2 * max3, max1 * min1 * min2)
```

© Daily Coding Problem 2019

Privacy Policy

Terms of Service

Press