HOMEWORK 1

COMP3121 - ALGORITHM DESIGN

QUESTION 3

PUNEETH KAMBHAMPATI

z5164647

Brute Force:

The most straightforward solution would be to find the heaviest apple first which would require 1023 weighings. Then find the second heaviest apple by weighing all the rest of the apples again which will need 1022 weighings. This would require 2045 comparisons and it would definitely exceed the limit of 1032 from the question.

In the Brute force,

Heaviest Apple	- 1023 weighings
Second Heaviest Apple	- 1022 weighings

Binary Tree Solution:

To provide an overview, by introducing a binary tree to keep track of comparisons, we can avoid the 1022 weighings for the second heaviest apple and narrow it down to 9 weighings. Reaching a total of,

$$1023 + 9 = 1032 \text{ weighings}$$

The first step would be to weigh all apples in designated pairs. Each apple will be a leaf node in the binary tree. At the leaf layer, we will have

$$\text{Number of weighings} = \text{number of total apples} / 2 = 512$$

All the apples that were heavier, will move to the next layer. And this process will repeat till we have just one apple left, which will be out **ROOT**.

The apple at the ROOT will be the heaviest apple. To reach this step we will have made,

$$512 + 256 + 128 + 64 \ldots + 2 + 1 = 1023 \text{ weighings}$$

Now, to narrow down our search for the second heaviest apple. We know that the second heaviest apple would only stop moving to the next layer when compared to the biggest apple. So, we only have to look at the apples that lost the weighing competition to the heaviest apple.

So, we look at the apple faced by the heaviest apple in every layer. For a tree with 1024 apples, we would have 10 layers including the ROOT layer. So the number of comparisons that we need to look at are 10-1 = 9.

Therefore, we can find the next heaviest apple by keeping track of the comparisons through a binary tree.