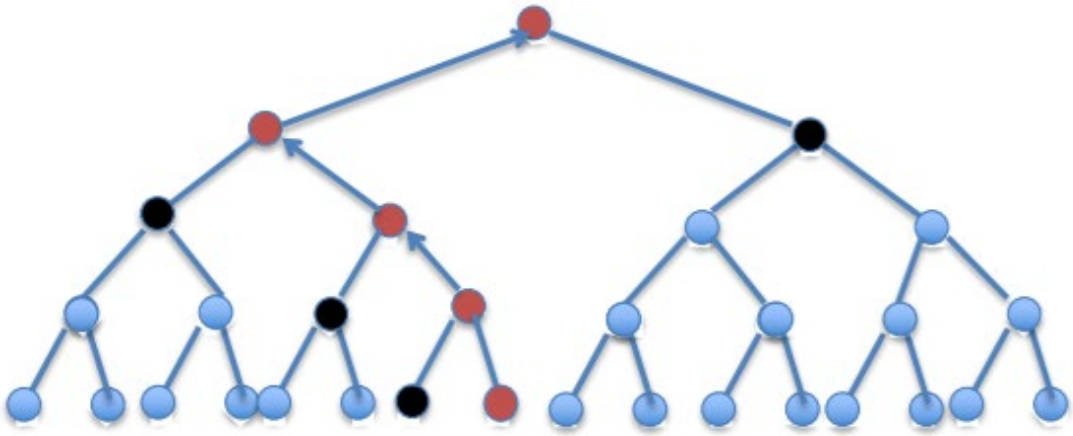
Dheeraj Satya Sushant Viswanadham
(z5204820)

COMP3121 Assignment 1 – Question 3

3) If we think about this conceptually, then we will reach the below conclusion:

We have 1024 distinct apples. We can only compare 1 apple against 1 other. We can hence create 512 pairs of apples and compare them against each other. We then take the heavier apple of the pair and compare it against the heavier apple of another pair. This goes on until we reach the heaviest apple and would be as follows: 512 -> 256 -> 128 -> 64 -> 32 -> 16 -> 8 -> 4 -> 2 -> 1 and in total would be 1023 weightings. Then, to get the second-heaviest apple, we know that at some point of the weightings it had gone up against the heaviest apple (otherwise it would not have 'lost'). Hence, we need to keep track of which of the 10 apples that the heaviest apple had gone up against and weigh them against each other which would take a total of 9 weightings. This is determined by the following logic - to determine the heaviest of these remaining 10 we follow a similar method as above where we would have 5 pairs of apples, except this time we have the fifth pair of apples to be compared against the final heaviest pair of the initial 4 pairs as 10 is not a result of 2^x whereas 2^3 = 8 so we first compare those 8 balls against each other. Summing it all together, we would have 1023 weightings + 9 weightings = **1032 total weightings**.

See given diagram as an overall visual example: (note this diagram was given in the H/W hints)



Example Pseudocode:

```
class App {

        # … initialise …

}

class Balance {

        # … initialise …

        # weigh function e.g. weight(left, right) and returns (left, right) if left is heavier than right otherwise vice-versa

}
```

Dheeraj Satya Sushant Viswanadham
(z5204820)

initialise new variables for our balance and apples making sure that we have 1024 apples e.g. Bal = new Balance();

Apply mergesort on the apples until we have our last pair of apples

    # in the mergesort make sure to only track the previous weighs of the heavier apple when weighing previous weighs to keep track of our second-heaviest apple – we can disregard the previous weighs of the lighter apple as its not needed.

To get our second-heaviest apple we can then simply weigh the previous apples that the heaviest apple had gone up against e.g.

for apple in heaviest.trackPrev do:

    left = app[x];

    right = app[x – 1];

    # and then weigh them on the balance against each other until we reach our heaviest of the remaining 10 apples that our overall heaviest apple had gone up against.

End