SNS College of Engineering Coimbatore - 641107

NON-RECURSIVE ALGORITHM

AP/IT

General Plan for Analysis of non – recursive algorithm

Decide on parameter *n* indicating *input size*

Identify algorithm's *basic operation*

Determine *worst*, *average*, and *best* cases for input of size *n*

Set up a sum for the number of times the basic operation is executed

Simplify the sum using standard formulas and rules

Useful summation formulas and rules

$\sum_{l \le i \le n} 1 = 1+1+\ldots+1 = n \in \Theta(n)$

$\sum_{1 \le i \le n} i = 1+2+\ldots+n = n(n+1)/2 \in \Theta(n^2)$

$\sum_{1 \le i \le n} i^2 = 1^2+2^2+\ldots+n^2 = n(n+1)(2n+1)/6 \approx n^3/3 \in \Theta(n^3)$

$\sum_{0 \le i \le n} a^i = 1 + a + \ldots + a^n = (a^{n+1} - 1)/(a - 1)$ for any $a \neq 1$

In particular, $\sum_{0 \le i \le n} 2^i = 2^0 + 2^1 + \ldots + 2^n = 2^{n+1} - 1 \in \Theta(2^n)$

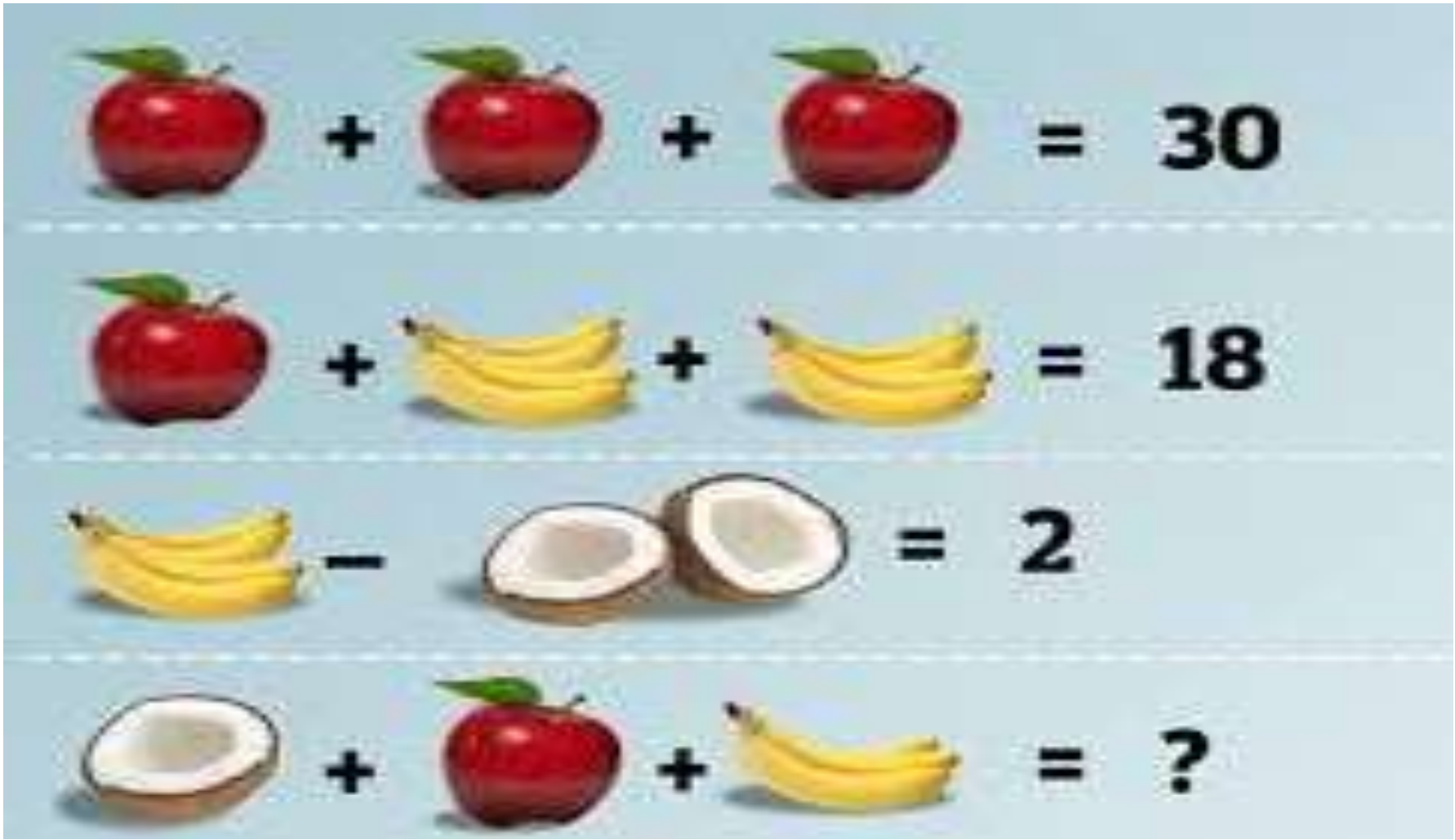
Example 1: Maximum element

ALGORITHM *MaxElement*$(A[0..n-1])$

```
//Determines the value of the largest element in a given array
//Input: An array A[0..n − 1] of real numbers
//Output: The value of the largest element in A
maxval ← A[0]
for i ← 1 to n − 1 do
    if A[i] > maxval
        maxval ← A[i]
return maxval
```

T(n) = $\Sigma_{1 \le i \le n-1} 1 = n-1 = \Theta(n)$ comparisons

BREAK



- Decide
- Identify
- Determine
- Set up
- Solve