Basics of Programming

L11: Recursion-II

Apr 2020

Dr. Ram P Rustagi
Professor, CSE Dept
KRP, KSGI
rprustagi@ksit.edu.in

Recursion

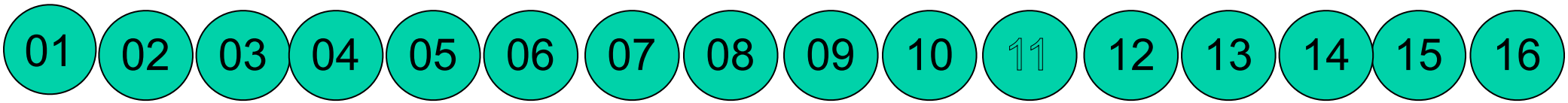
- Recursive approach
  - Break the given problem into smaller problems
  - Find the smallest size problem that you can solve
  - Merge the solution from smaller problem solve bigger problems

Review

- Hanoi's tower
- Tic-Tac-Toe program
- Combine(N,K)

Find Defective (light) Ball

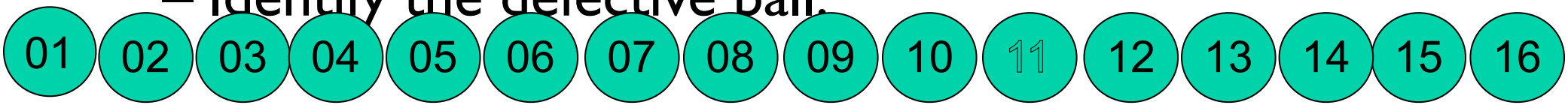
- Given 16 balls with one defective (say lighter)
  – Identify the defective ball.



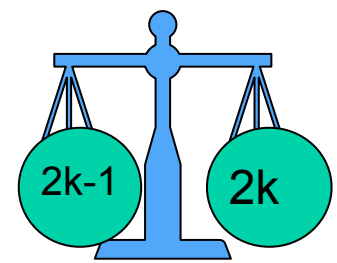
- Solution 1:
  - Compare 1 with 2
  - Compare 1 with 3
  - :
  - Compare 1 with 16
- Time taken:
  - 15 comparisons (worst case)

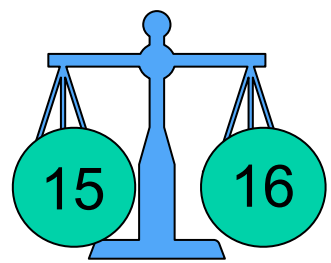
Find Defective (light) Ball

- Given 16 balls with one defective (say lighter)
  - Identify the defective ball.



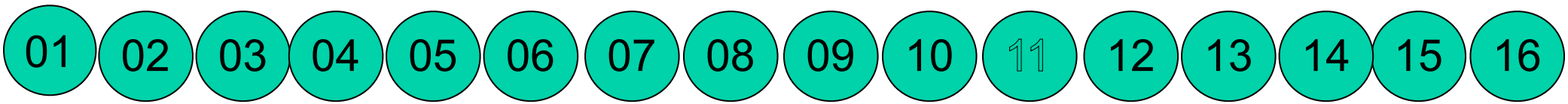
- Solution 2:
  - Compare 1 with 2
  - Compare 3 with 4
  - :
  - Compare 15 with 16
- Time taken:
  - 8 comparisons (worst case)



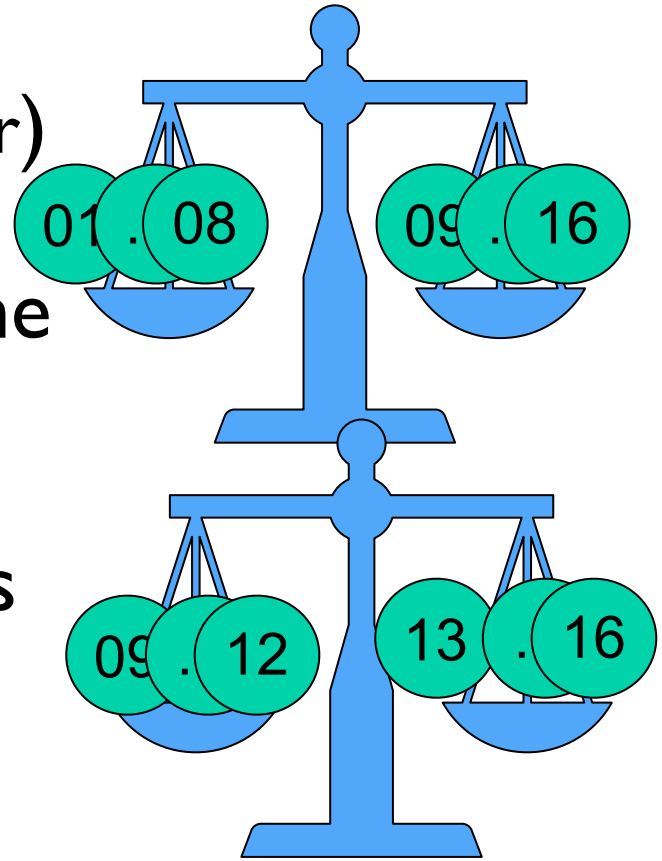


Find Defective (light) Ball

- Given 16 balls with one defective (say lighter)
  – Identify the defective ball.

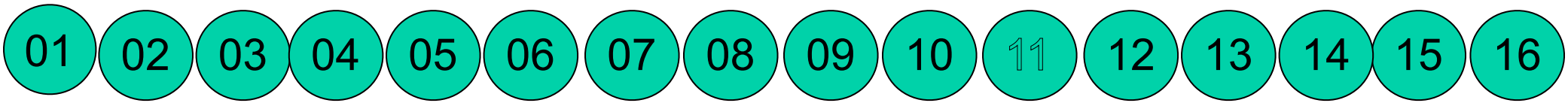


- Solution3: Tail Recursion (Divide & Conquer)
  - Divide into 2 sets, each of 8 balls
  - Compare 1-8 with 9-16, and divide the lighter set into two parts each of 4.
    :
  - Continue the process till lighter ball is found
- Time taken: 4 comparisons ($\log_2 16$)



Find Defective (Light) Ball

- Given 16 balls with one defective (say lighter)
  – Identify the defective ball.



- Solution3: Total number of comparisons

```
O(log N)
```

Binary Search

- General logic of Binary search
  - Given N elements in sorted order in a list,
  - Find if value K exists

```
def Binsearch(key, A, low, high):
    if low >= high:
        return -1
    mid = (low + high) // 2
    if A[mid] == key:
        return mid
    if A[mid] > key:
        return Binsearch(key, A, low, mid]
    else:
        return Binsearch(key, A, mid+1, high)
```

Reverse of a list

- Given a list of N elements, reverse the elements using recursion
- Approach:
  - Decrease the problem size by 2.
    - Solve the problem of size N-2
    - Swap the two end elements (Merge operation)

```
def reverse(L,low,high) # index high excluded
   if low + 1 >= high:
      return
   L[low],L[high-1] = L[high-1], L[low]
   return reverse(L, low+1, high-1)
```

Digit Check in a Number

- Problem: check if digit `d` is present a +ve integer `n`
- Approach:
  - if `n` consists of only 1 digit
    - return `True` if `n==d` else return `False`
  - if the last digit is `d`, return `True`
  - Else repeat the process for `n/10`

```
def check(n,d):
   if (n==0):
      return False
   elif (n%10 == d):
      return True
   else:
      return check(n//10, d)
```

Exercises using Recursion

- Ex 01: Check if a given string is a palindrome
- Ex02: List all the prime factors of a number N
  - e.g.if N is 24, then it should print, 2 2 2 3
- Ex03: Converting a number in its binary form
- Ex04: Print all permutations of numbers
  - e.g. Given list of 4 elements [a,b,c,d]
    - print all `16` permutations

Questions