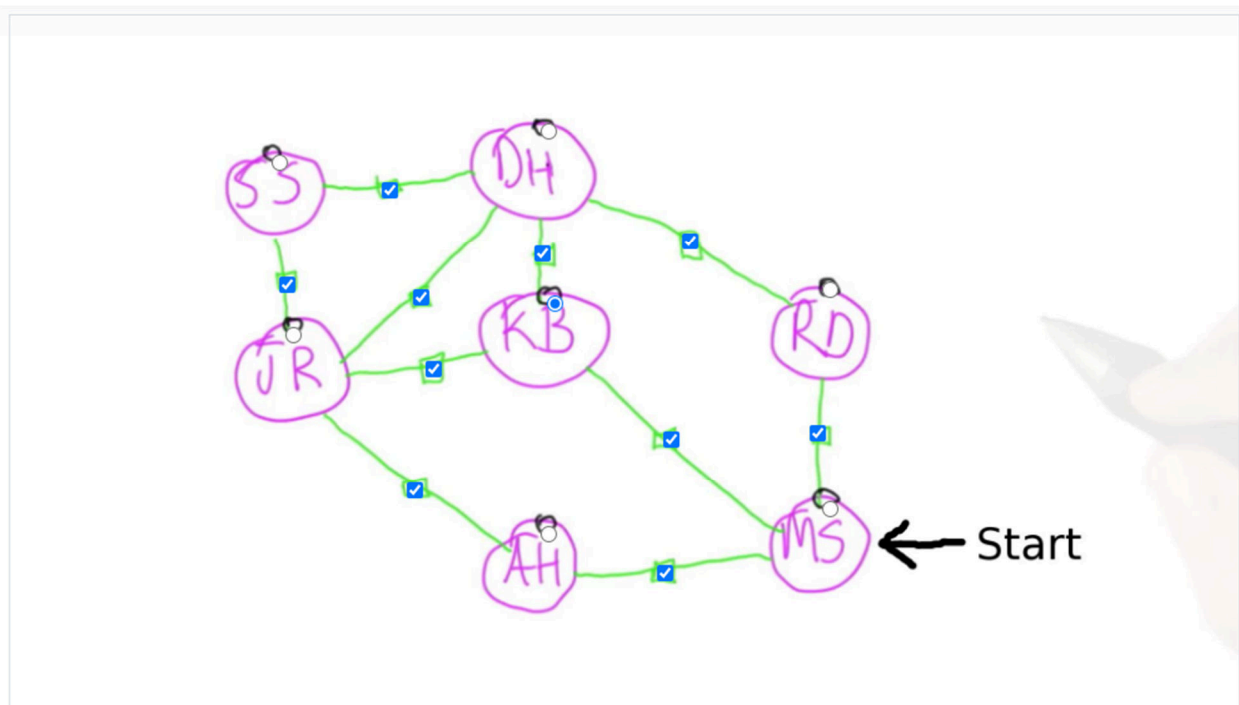


## Magic Trick



## Eulerian Path

Can a graph with only even-degree nodes have an Eulerian path?

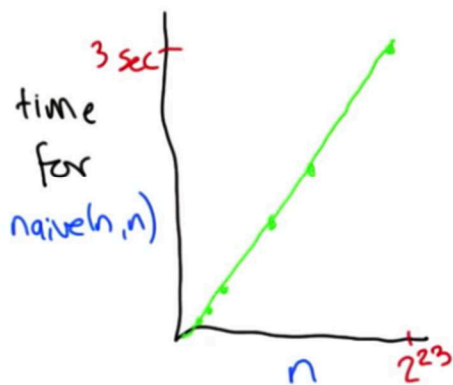
- No, the start and end nodes must have odd degree
- Yes, it depends on the graph, though
- Yes, all such graphs do.
- Yes, all graphs do.

## Case Study

What does  
 $\text{naive}(a,b)$   
compute as a  
function of  $a$  &  $b$ ?

- ☐  $\max(a,b)$
- ☐  $a-b$
- ☐  $b-a$
- ☐  $a+b$
- ☒  $a \times b$

## Running Time



How does running time  $t$   
relate to input  $n$ ?

- ☐ roughly constant  $t \approx c$
- ☐ roughly logarithmic  $t \approx \log n$
- ☒ roughly linear  $t \approx cn$
- ☐ roughly exponential  $t \approx c^n$

## Russian Peasant Algorithm

Bit Shifts


What is  $17 \gg 1$  ?

- ☐ 171
- ☐ 9
- ☒ 8
- ☐ 8.5
- ☐ 34




## How Many Additions

How many additions for  $\text{russian}(20, 7)$



## Measuring Time

```
1 # How many units of time will this python code take to run?
2
3 s = 0
4 for i in range(10):
5     s = s + i
6     print s
7
8
9
10
11
12
```



## Counting Steps

```
1 import math
2
3 def time(n):
4     """ Return the number of steps
5     necessary to calculate
6     `print countdown(n)` """
7     steps = 0
8     steps = 3 + 2 * math.ceil(n/5.0)
9     return steps
10
11 def countdown(x):
12     y = 0
13     while x > 0:
14         x = x - 5
15         y = y + 1
16
17     return math.ceil(y*2)+3
18
19 print countdown(50)
20
```

## Steps for Naïve

```
1 # counting steps in naive as a function of a
2
3 def naive(a, b):
4     x = a
5     y = b
6     z = 0
7     while x > 0:
8         z = z + y
9         x = x - 1
10    return z
11
12 def time(a):
13     # The number of steps it takes to execute naive(a, b)
14     # as a function of a
15     steps = 0
16     steps = 2*a + 3
17     return steps
18
19 print (time(5))
```

## Halving

How many times can you divide a number  $x$  in half (rounding down) before it hits zero?

$x$	1	2	3	4	5	6	7	8	9	10	11
#halvings	1	2	2	3	3	3	3	4	4	4	4

- $x$
- $x/2$
- $\log_2 x$
- $\lfloor \log_2 x \rfloor + 1$

How many times can you divide a number  $x$  in half (rounding down) before it hits zero?

$x$	1	2	3	4	5	6	7	8	9	10	11
#halvings	1	2	2	3	3	3	3	4	4	4	4

- $x$
- $x/2$
- $\log_2 x$
- $\lfloor \log_2 x \rfloor + 1$

- How many times can you divide a number  $x$  in half (rounding down) before it hits zero?
- | $x$       | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|-----------|---|---|---|---|---|---|---|---|---|----|----|
| #halvings | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 4 | 4 | 4  | 4  |
- $x$
  - $x/2$
  - $\log_2 x$
  - $\lfloor \log_2 x \rfloor + 1$

## Recurrence Relation

$$T(n) = \begin{cases} 1 & \text{if } n=0, \\ 3 + T(n/2) & \text{if } n \text{ is even,} \\ 3 + T((n-1)/2) & \text{else} \end{cases}$$

○  $T(n) = \lfloor \log_2 n \rfloor + 1$

○  $T(n) = 3 \lfloor \log_2 n \rfloor + 3$

○  $T(n) = 3 \lfloor \log_2 n \rfloor + 1$

○  $T(n) = 3 \lfloor \log_2 n \rfloor + 4$