COP 3502

```
int sum = 0;
while (n > 0) {
    Sum += n%2;
    n /= 2;
}
```

How many times does loop run?

$n, \frac{n}{2}, \frac{n}{4}, \frac{n}{8}, \dots \frac{n}{2^k} = 1$

Value of k for which

$n = 2^k$

by def $k = \log_2 n = O(\lg n)$

$b > 1$

$\log_b n = \frac{\log_c n}{\log_c b}$, log rule for any $c > 1$.

$\log_c b \rightarrow$ CONST.

log tends to appear when repeated division or multiplication embedded in the analysis.

bits in binary representation of n is $O(\lg n)$.

$S = 1 + \frac{1}{2} + \frac{1}{4} + \dots = \frac{a_1}{1-r} = \frac{1}{1-\frac{1}{2}} = \boxed{2}$

$a_1 \quad r = \frac{1}{2}$

What is the average case run time of binary search on an array size n, $n = 2^k - 1$ for an integer k, assuming that we are searching for a randomly chosen item in the array.

$n = 7$

2	6	8	9	20	25	40
3	2	3	1	3	2	3

$$= \frac{3 \times 4 + 2 \times 2 + 1 \times 1}{7}$$

Def avg run time $\sum_{x \in \text{Inputs}} \underbrace{P_x}_{\text{prob of input } x} \cdot \underbrace{f(x)}_{\text{run time for input } x}$

# Searches	Prob
1	$\frac{1}{n}$
2	$\frac{2}{n}$
3	$\frac{4}{n} \to 2^{3-1}$
4	$\frac{8}{n} \to 2^{4-1}$
⋮	
K	$\frac{2^{k-1}}{n}$

$$\sum_{i=0}^{k-1} \cancel{(i+1)} \frac{2^i}{n} \times (i+1)$$

$$S = \frac{1}{n} \times 1 + \frac{2}{n} \times 2 + \frac{4}{n} \times 3 + \frac{8}{n} \times 4 + \cdots \frac{2^{k-1}}{n} \times k$$

$$S = \frac{1}{n}\times 1 + \frac{2}{n}\times 2 + \frac{4}{n}\times 3 + \frac{8}{n}\times 4 + \cdots \frac{2^{k-1}}{n}\times k$$

$$-2S = \frac{2}{n}\times 1 + \frac{4}{n}\times 2 + \frac{8}{n}\times 3 + \quad \frac{2^{k-1}}{n}\times(k-1) + \frac{2^k}{n}\times k$$

$$-S = \frac{1}{n} + \frac{2}{n} + \frac{4}{n} + \frac{8}{n} + \cdots \frac{2^{k-1}}{n} - \frac{k\times 2^k}{n}$$

$$-S = \frac{1}{n}\left(2^k - 1 - k\times 2^k\right)$$

$$S = \frac{1}{n}\left(k\times 2^k - 2^k + 1\right)$$

$$S = \frac{1}{n}\left(2^k(k-1)+1\right)$$

$n = 2^k - 1$

$2^k = n+1$

$k = \log_2(n+1)$

$$S = \frac{1}{n}\left((n+1)(k-1)+1\right)$$

$$S = \frac{1}{n}\left((n+1)(\log_2(n+1)-1)+1\right)$$

$$O\left(\frac{n}{n}\lg n\right) = O(\lg n)$$

Old Foundation Exam Qs

Section C Question 1

Fall 2025
Summer 2025
Fall 2023
Spring 2022
Sum 2020

Fall 2025

```
lo = 0
hi = 2^k

while (lo < hi) {

    mid = (lo + hi)/2;
    O(n)

    if (     )
        lo = mid+1
    else
        hi = mid;
}
```

for loop simple loop body

$hi - lo$ gets divided by 2 each time, and it starts at $2^k - 0 = \boxed{2^k}$

How many times do I have to divide 2^k by 2 to get it down to 1?

k] loop runs at most $O(k)$ times

So total run time is $O(kn)$.

Sum 2025

n # bits 0 to 2^k-1

0000
0001
0010
0011
0100
0101
:
0111
1000 $1 \rightarrow$ time

Neat idea is to look @ 1 bit.

lsb toggles every time! (2^k-1 times)

2^{nd} lsb = every other time ($2^{k-1}-1$ time

last bit = (2^1-1 times)

$$\sum_{i=1}^{k}(2^i-1) = \sum_{i=1}^{k} 2^i - \sum_{i=1}^{k} 1$$

$$= \left(\sum_{i=0}^{k} 2^i\right) - 2^0 - k$$

$$= 2^{k+1} - 1 - 1 - k$$

$$= 2^{k+1} - k - 2$$

$$\underbrace{\boxed{1+2+1+3+1+2+1}}_{T(3)} + 4 + \underbrace{\boxed{}}_{T(3)}$$

Let $T(n)$ be answer for n, then

$$T(n) = n + 2T(n-1), \quad T(1) = 1$$

$$S = 2^{k-1}\times 1 + 2^{k-2}\times 2 + 2^{k-3}\times 3 + \cdots \; 2^0 \times k$$

Fall 2023

$n=4$ $k=2$

AAAA ×

AAAB ×

⋮

BBBB ✓

$2 \times 2 \times 2 \times 2 = 2^4$

\# of strings in list

2^n strings

amt of time on each string

$O(n)$

Ans $= O(n2^n)$.