Josephus for n elements and kth elimination

Recursion approach : Since recursion progression is linear, so we can use INDUCTION and SUBSTITUTION method.

Example : n=5, k=3;

HYPOTHESIS : j5 = josephus(n=5,k=3)

SUBSTITUTION : j4 = josephus(n=4,k=3)

INDUCTION: derive j5 using j4

$n=5;$ 0 1 ~~2~~ 3 4

3 4 0 1 (transformed into n=4)

$n=4;$ 0 ~~1~~ ~~2~~ ~~3~~

2 1 3

$j_{4;3}$

this relation helps deriving j5 using j4

$$\boxed{j_5 = (j_4 + k)\,\%\,n}$$

```java
public static int recursiveJosephus(int n, int k) {
    if(n==1) return 0;

    int j_nMinus1 = recursiveJosephus(n - 1, k);

    int j_n = (j_nMinus1 + k)%n;

    return  j_n;
}
```

Josephus solution for k = 2

observations:

2^1 2^2 2^3 2^4

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$f(n)$	1	1	3	1	3	5	7	1	3	5	7	9	11	13	15	1

Safe point

increasing odd sequence

Josephus safepoint for k=2 is an increasing odd sequence that restarts with 1 whenever the index n is a power of 2. Therefore, if we choose M and L so that n=2^M +L, then safepoint is Lth odd sequence point that is (2*L + 1)

Q. What is the 5th odd and 5th even number?

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

1st odd   2nd odd   3rd odd   5th odd

$$\boxed{n^{th} \text{ odd} = 2n - 1 \text{ or } 2n + 1}$$

$$\boxed{n^{th} \text{ even} = 2n}$$

Understanding m & L

Let's understand k = 2 & n = 13.

$m = 3$ $\uparrow$ $\lfloor \log_2 13 \rfloor$ $\leftarrow$ 8 $\to 2^3$, $\log_2 8 = 3$

8 ——— 13 ——— 16

16 $\to 2^4$, $\log_2 16 = 4$ $\to \lceil \log_2 13 \rceil \to m+1 = 4$

$n = 13$, $n = 2^m + L$

$= 2^3 + 5$

So, for $n = 13$, $m = 3$ & $L = 5$

for $n = 13$, $\boxed{n = 2^m + L}$

So, $\boxed{L = n - 2^m}$

Safe point is L^{th} odd sequence point

i.e $(2L+1)$

$\Rightarrow 2L+1 = 2(n-2^m) + 1$

$= 2(n - 2^{\lfloor \log_2 n \rfloor}) + 1$

$\Longrightarrow$

$\lfloor \log_2 n \rfloor = m$

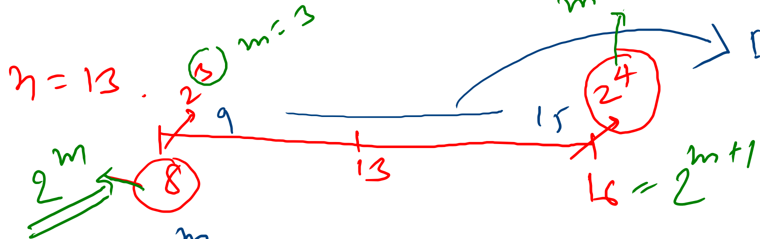
$\lfloor \log_2 13 \rfloor = 3$

So, $\lfloor \log_2 n \rfloor = m$

$\lceil \log_2 n \rceil = m+1$

Calculating m & m+1 using bitwise

Finding safe point using bitwise operation for k = 2.



If we multiply any number from this range will be greater than $\underline{2^{m+1}} = 2^4 = 16$

$n = 2^m + L$; $n = 8$ to $n = 15$ can be represented as $\underline{2^3 + L}$, using $L = 0$ to 7

$8 = 2^3 + 0$

$9 = 2^3 + 1$

$10 = 2^3 + 2$

$11 = 2^3 + 3$

$\vdots$

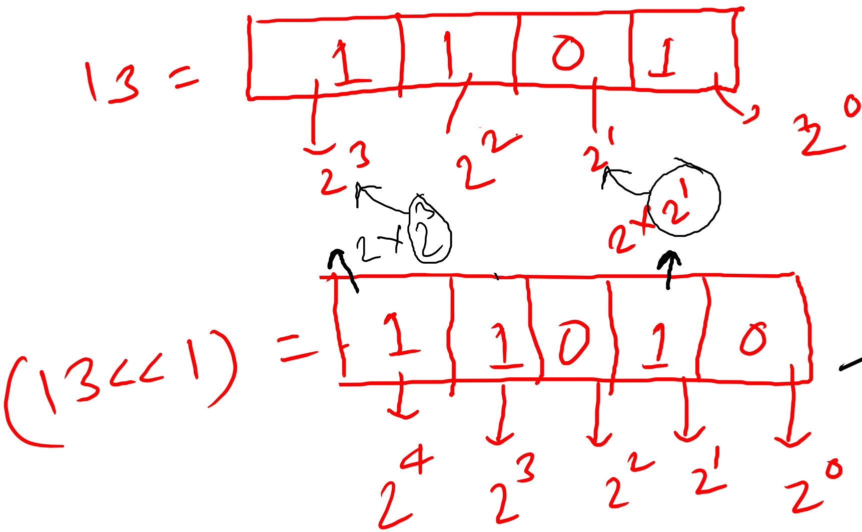
$15 = 2^3 + 7$

$2^m =$ Integer.highestOneBit(n)

$2 \times 2^m = \underline{2^{m+1}} =$ Integer.highestOneBit(2×n)

Multiply by 2 using bitwise operator

$$(n<<1) == 2\times n$$

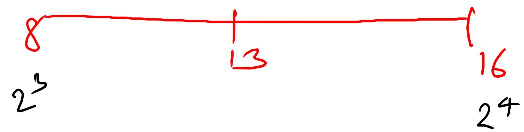


→ effectively we are multiplying each bit position by 2.

Divide by 2 using bitwise operation

$(n>>1)$ → it divides each bit position by 2.

$n = 13, \; k = 2,$



$n = 2^m + L$

$2n = 2 \times 2^m + 2L$ — eq 1

Safe point $= 2L + 1$

In eq 1, if we add 1 and remove $\underline{2 \times 2^m}$ term will give Safe point.

Strategy to get Safepoint using bitwise operation:

$n = 2^m + L$

Step1 (i) multiply the 'n' by 2

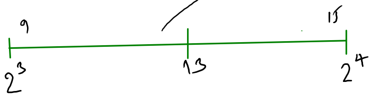
$2n = 2 \times 2^m + 2L$

Step2 (ii) add 1 to the output of step I

$2n + 1 = 2 \times 2^m + \underbrace{(2L + 1)}_{\text{safe point}}$

Step(iii) make 2×2^m term as zero in step ii

$n = 13, \; K = 2$



If we multiply any number by 2 from this range will be greater than or equal to 2^4 i.e $\underline{2^{m+1}}$

Step1 = multiply the n by 2

$$2n = 2 \times 2^m + 2L$$

bitwise multiply by 2

$$(n << 1)$$

$$n << 1 = (2^m << 1) + (L << 1)$$

Step2 add 1 to step1

$$((n << 1) \mid 1)$$

$$(n << 1) \mid 1 = 2^m << 1 + ((L << 1) \mid 1)$$

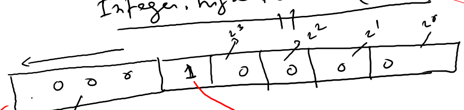
Step3 remove $2^m << 1$ term.

$2^m << 1 = 2^{m+1}$ → We will create an 'and mask' having $(m+1)^{th}$ bit position as zero and other bits as 1.

Integer.highestOneBit($2 \times n$) $= 2^{m+1} = 2^{3+1}$

13

$2 \times n = 2 \times 13 = 26$ greater than $2^{3+1} = 2^4$



all zero in remaining 27 bits

2^{m+1}

If we take complement of this will give desired &mask containing all 1's except at $(m+1)^{th}$ bit position.