Example of computation of worst case (Binary Search)

Result : worst - case running time

$T(n) = \lfloor \lg_2 n \rfloor + 1$.

Binary search, ordered.

Find an item x in an ordered array I based only of comparisons of x to elements of I.

size (I) - number of elements to be searched (= n).

$I[0] \le I[1] \le I[2] \le \ldots \le I[i-1] \le$
$I[i] \le \ldots \le I[n-1]$
$\wedge$

midpoint $i = \left\lfloor \frac{n-1}{2} \right\rfloor$

Formulas we will use :

For every $m \in \mathbb{Z}$,

$m = \left\lfloor \frac{m}{2} \right\rfloor + \left\lceil \frac{m}{2} \right\rceil$ $\left(\text{same as } m - \left\lfloor \frac{m}{2} \right\rfloor = \left\lceil \frac{m}{2} \right\rceil\right)$

$\left\lceil \frac{m}{2} \right\rceil = \left\lfloor \frac{m+1}{2} \right\rfloor$ $\left(\text{hence } m - \left\lfloor \frac{m}{2} \right\rfloor = \left\lfloor \frac{m+1}{2} \right\rfloor\right)$

Exercise : prove both (should be easy).

Note : $\left\lfloor \frac{n}{2} \right\rfloor$ is performed by a binary operation "shift one position to the right".

Unsucessful search for $x > I[n-1]$ yields an example of worst - case number of comparisons.

If I contains no duplicates then sucessful search for $x = I[n-1]$ yields an example of worst - case number of comparisons as well.

Optional exercise for smart students
Prove the above two statements.

We will prove that this worst - case number of comparisons is exactly equal to the number of significant bits in the binary representation of n, that is, to $\lfloor \lg_2 n \rfloor + 1$.

We will show that while searching for an element larger than any element of the array A of m elements, after one comparison, $\left\lfloor \frac{m}{2} \right\rfloor$ elements (those after the midpoint) still need to be searched.

Since the binary representation of $\left\lfloor \frac{m}{2} \right\rfloor$ has one less bit than m has, this will show that in order to reduce the size of the set of elements still under search to 0, the number of comparisons must be equal to the number of significant bits in the binary representation of n.

Example. If $n = 10$, or 1010 in binary,

after first comparison ($x > I[4]$) there are still 5, or 101 in binary, elements to search,

after second comparison ($x > I[7]$) there are still 2, or 10 in binary, elements to search,

after third comparison ($x > I[8]$) there is still 1, also 1 in binary, element to search, and

after fourth comparison ($x > I[9]$) there are no elements to search.

Thus the search was completed in 4 comparisons, and that is also the number of significabt bits in the binary representation of 10.

The midpoint is $\left\lfloor \frac{m-1}{2} \right\rfloor$, and $x > A\left[\left\lfloor \frac{m-1}{2} \right\rfloor\right]$.

The number of elements of A after the midpoint is

$$(m-1)\,(\text{*the greatest index*}) - \left(\left\lfloor \frac{m-1}{2} \right\rfloor + 1\right)(\text{*the least index*}) + 1 =$$

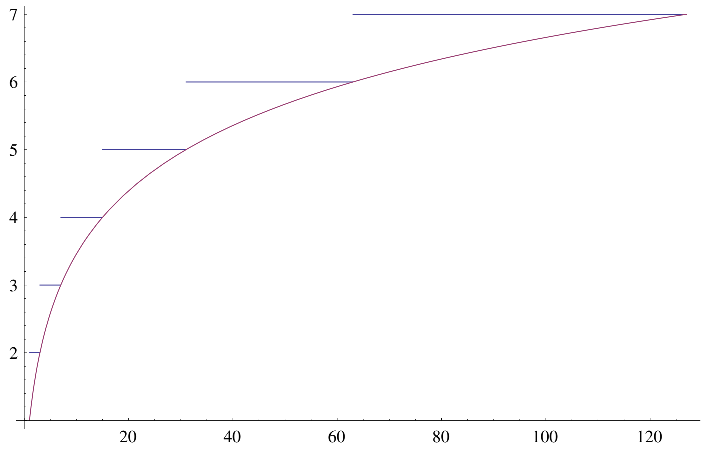
$$= m - 1 - \left\lfloor \frac{m-1}{2} \right\rfloor = \left\lceil \frac{m-1}{2} \right\rceil = \left\lfloor \frac{m}{2} \right\rfloor$$

So, what is the number of significant bits in the binary representation of n?

Answer :

$\lfloor \lg_2 n \rfloor + 1$

```
Plot[{⌊Log2[n]⌋ + 1, Log2[n + 1]}, {n, 1, 127}]
```



7
6
5
4
3
2
20
40
60
80
100
120